



Major Project Final Report

COM553 & COM559

Bsc Hons Interactive Multimedia Design

Student: Michael Mckeever (B00580235)

Supervisor: Stephen Hagan

<https://moodmemo.co>

Acknowledgements	5
1.0 - Introduction	6
1.1 - Background	6
1.2 - Aims & Objectives	8
1.2.1 - Aims	8
1.2.2 - Objectives	8
1.3 - Overview of work undertaken	9
1.4 - Overview of Report	11
2.0 - Concept Definition & Testing	12
2.1 - Idea Generation	12
2.2 - Requirements specification	14
2.2.3 - Further Non-functional Requirements	14
2.2.4 - Dependencies	18
2.3 - Paper Prototyping	19
2.3.2 - 6-Ups	21
2.3.3 - High fidelity Mockup	23
2.3.4 - Styletiles	24
2.3.5 - User Personas	26
2.4 - Feasibility Testing	27
2.5 - Methodology Testing	28
2.5.1 - Waterfall Methodology	28
2.5.2 - Prototyping Methodology	29
2.5.3 - Chosen Methodology	30
3.0 - Design	32
3.1 - Branding	32
3.2 - Typography	33
3.3 - Voice & Tone	34
3.4 - Colour	34

3.5 - Empty States & Onboarding	35
3.5.1 - Timeline iterations	35
3.5.2 - Statistics	36
3.5.3 - Map	37
3.6 - Page Errors	37
3.7 - Online Styleguide	38
4.0 - System Design & Architecture	40
4.1 - Refined Site Map	40
4.2 - User Flow	41
4.3 - Platform Architecture	43
4.4 - Use Case Diagram	44
4.5 - Logical Design	45
4.6 - Database Design	47
4.7 - Technology Selection	49
4.7.5 - APIs	54
5.0 - Implementation	56
5.1 - Tool Selection	56
5.1.1 - Google Maps API	56
5.1.2 - Skeleton Framework	57
5.1.3 - Highcharts API	57
5.2 - Technologies & Tools Used	60
5.2.1 - jQuery radio buttons	60
5.2.2 - Sweet Alert	61
5.2.3 - Favouriting a Moodmemo with AJAX	62
5.2.4 - View password on click/tap	63
5.2.5 - User Profile Photo	65
5.2.6 - Encrypting & decrypting a Moodmemo	66
5.3 - Notable Challenges	67
5.4 - Notable Achievements	68

6.0 - Testing	70
6.1 - Testing Approach	70
6.1.1 - White Box Testing	70
6.1.2 - Black Box Testing	70
6.1.3 - User Testing	71
6.2 - Test Process	71
6.2.1 - White Box Testing	71
6.2.2 - Black Box Testing	71
6.2.3 - User Testing	72
6.3 - Test Results	73
6.3.1 - Browser Testing	73
6.3.2 - User Testing	74
6.3.2 - Device & Responsive Testing	75
7.0 - Evaluations	77
7.1 - Project Outcomes Evaluation	77
7.2 - Methodology Evaluation	78
7.3 - Plan Evaluation	78
8.0 - Conclusion	79
8.1 - Summary of Report	79
8.2 - Reflection	79
8.3 - Reflection on my role	80
8.4 - Suggestions for further work	81
9.0 - References	83

Acknowledgements

I would like to, firstly, thank my mentor, Stephen Hagan, for all his support, wisdom and knowledge throughout the duration of the project, of which I am eternally grateful for.

Secondly, I would like to thank my peers and fellow classmates, Sean Murray and Christopher Rodgers for all of their support and encouragement when things got tough. Together we all kept each other on the right path.

I would also like to give thanks to Matthew Snoddy and David Turner for their help in overcoming particular technical challenges and answering any questions I had, no matter how naive they may have been.

Finally, I would like to thank Peter Nicholl, the course director, for working hard to make sure the Interactive Multimedia Design course runs smoothly and efficiently and also to all of my lecturers for their inspiring work ethic in lecturing all the final year student.

Thank you & enjoy

1.0 - Introduction

This report outlines all aspects of the Major Project and its development cycle to discuss each stage of the project and justify any choices that were made. It is structured in a way to represent the chosen Methodology. This first section introduces the report and looks in detail at the background, purpose of this project, and the outlined aims and objectives for the project.

1.1 - Background

Mental health has a large stigma associated with it. It is largely misunderstood and mistreated. 1 in 4 of us will experience some kind of mental health problem in the course of a year [*Mentalhealth.org.uk, (2015)*]. Mental health affects many of us but the attached stigma associated with it dissuades people from confronting and discussing it. Mental health within the tech industry, in particular, has become a big concern and talking point. It has only been within the last year that there has been a noticeable surge in the discussion of the topic more openly. Notable speakers within the industry such as Ed Finkler, [*Finkler, E. (2013)*] Christopher Murphy [*Murphy, C. (2014)*] and others have been doing a great deal in raising awareness and getting a conversation going. A tech conference called Prompt [*Prompt.engineyard.com, (2015)*] has even been started that centres around mental health within the industry and last year saw the introduction of Geek Mental Help Week [*Geekmentalhelp.com, (2014)*], a week-long “series of articles, blog posts, conversations, podcasts and events across the web about mental health issues, how to help people who suffer, and those who care for us.” Things are progressing but there is still a long way to go.

This is why, after investigating several potential projects to undertake for the basis of the Major Project that it was decided to develop a tool, Moodmemo, that would enable users to keep track of their moods, privately, wherever they are. The mission statement of Moodmemo is not to pursue the mental health issue but rather, provide

people with an outlet where they can write down their thoughts and keep track of their moods.

Geek Mental Help week ran from 27th October - 2nd November and featured a week long series of articles, blog posts, conversations and podcasts related to mental health issues. *[Clarke, A. (2014)]*

A particularly interesting read was an article by Brandon Scott discussing his experiences with anxiety. Brandon talks about his battle with anxiety disorder from an early age and how he believes he has finally overcome it through a mixture of therapy and medication. Brandon says:

“In the early days I used a tool called OhLife which emailed you everyday, and you could reply back with whatever you liked. It was great, I could put my thoughts somewhere, and didn't have to keep them all to myself.” [My thoughts, opinions and other notes, (2014)]

OhLife was a web app, like Moodmemo, that allowed you to keep track of your moods and jot down feelings. Unfortunately, it shut down in 2013 due to lack of funding. *[Ohlife.com, (2013)]* However, it was great to hear that Brandon had been seeking therapy in an online tool, a tool to jot down his daily thoughts as a means of catharsis, as this is precisely the ethos of Moodmemo, to give someone a place where they can jot down feelings and not feel like their data is being shared with advertisers or analytical purposes.

Research undertaken did provide tools that allow you to track your mood, such as MoodPanda for instance, *[MoodPanda.com, (2015)]*. However MoodPanda is different

in that it is more of a social network where users can interact with each other than an online diary where you can keep track of your moods. The main difference between MoodPanda and Moodmemo, is that MoodPanda is not private by default. Whatever is posted is displayed on a public feed, where it can be read and interacted with by other members. There is an option to turn on 'Privacy Mode', meaning no one can see the posts that user creates, however, this is disabled by default. Moodmemo is always private, users can only read their own Moodmemos.

1.2 - Aims & Objectives

1.2.1 - Aims

The main aim for this project, which has remain unchanged since the conception of the project is to create a web app that will allow users to keep track of their moods by providing them with the means to create posts, known as 'Moodmemos', that will be displayed within a timeline to which only they have access to. The final objectives for this project that have evolved over the course of the project have been outlined below:

1.2.2 - Objectives

1. A web application with a user registration system that allows a user to create an account by providing their email address, first name and a password.
2. A web application with user profile personalisation features such as the ability to change their email address, first name, password and profile avatar.
3. A web application that is easy to use and provides the ability to create a Moodmemo from within any area of the application.
4. A web application that provides a timeline where users can view all of their created Moodmemos.

5. Further interaction features from within the timeline will allow a user to favourite or delete any Moodmemo.
6. A web application that provides a list where a user can view all of their favourited Moodmemos.
7. A web application that features statistics and analytics of a user's mood over time based on the Moodmemos they have created.
8. A web application that asks for a user's permission to track their location when creating a Moodmemo in order to pinpoint on a map where that Moodmemo was created.
9. A web application that gives the user the option to delete their account and all stored data related to their account from the application database.

1.3 - Overview of work undertaken

This section details the work undertaken to complete the objectives in order to create the product that was outlined.

Before undertaking the development of the product a framework and system had to first be established, as standard practice for any web product being developed. The framework consisted of a customised version of the Skeleton boilerplate, a lightweight, barebones, mobile-first boilerplate. [*Getskeleton.com, (2014)*] This provided a suitable starting ground to develop the framework but was also lightweight enough that it allowed the framework to be reworked to suit the needs of the project. For styling, a modular approach was undertaken using multiple stylesheet modules developed using SCSS/Sass. Using Sass helped to ensure the stylesheet structure was developed modularly and the code was clean, semantic and scalable.

A login and registration system was developed to allow a user the ability to interact with the product. This was carried out using the UserCake registration system, a fully

open source user management system. [Cassels, J. (n.d.)] User information is stored within a MySQL database, connections to the database are made via MySQLi. Passwords are encrypted using a 32 character salt and SHA1 hashing.

A form was used to develop the Moodmemo creation process. Radio buttons are used to present to the user the possible mood outcomes they can select. jQuery is used to determine the character count of the textarea and limit it to 220 characters. If the character count is exceeded the post button is disabled. The mood selected, entered text, user ID and datetime is then posted using PHP and stored in the database.

The text within a Moodmemo is encrypted before being posted to the database using the PHP mdecrypt function and a 32 character SALT. The Moodmemo is then decrypted using the SALT and mcrypt function when it needs to be displayed in plain text within the web application.

A user's timeline is created by querying the Moodmemo data from the database based on the user's id and formatting this into readable data that can be displayed within the timeline.

AJAX was used throughout the project for functions that required posting or getting from the database such as when a user wishes to delete or favourite a moodmemo.

The Google Maps and Geolocation API was used to determine a user's latitude and longitude and post these to the database along with Moodmemo data when a user is creating a Moodmemo. The latitude and longitude is then used to display each Moodmemo in the form of a marker on a Google Map located in the Maps page. If a user does not give consent to share their location when creating a Moodmemo then a marker is not displayed.

jQuery is used extensively within the web application for various functions and implementations. For instance, if no Moodmemos have been posted then a div is appended featuring content welcoming the user and explaining how to create a Moodmemo.

1.4 - Overview of Report

In the report for the Moodmemo Major Project each section will describe the work undertaken and examine in detail the progression that has been made from conception to final product. The initial planning stage lays the foundation for what needs are to be met in order to categorise the project as a success. The development stage will outline the creation of various features and implementations and how difficult and cumbersome tasks are managed and what skills were required and acquired in order to meet the outlined requirements of the project. The implementation and testing phase will consist of describing processes of the web application and how those processes occur. The concept phase of the report will outline the idea generation surrounding the project and design decisions that are made regarding the visual aspects of the project. The design phase will explore the visual journey of the project regarding the mockups that are produced and what UX resolutions have been proposed.

Once the web application is at or near the stage of completion the testing process can begin in order to test the features present in the system and ensure they are each carrying out their desired tasks and getting outlined requirements.

2.0 - Concept Definition & Testing

2.1 - Idea Generation

The idea for Moodmemo was born through a brainstorming session carried out by the creator searching for ideas of which to develop into a side project. Mental health was a sizeable talking point at the time, particularly within the tech and design community, and he felt that this is an area where more focus is needed. This focus lead to the idea of a web app that would allow users to track their moods and position them on a map automatically. Research was carried out and whilst there were tools that allowed you to keep track of your moods, none of the discovered were tracking the user's location and pinpointing them on a map. This, coupled with the Quantified Self movement gaining more and more traction with each year as more people start to track aspects of their everyday lives [Stone, L. (2013)], lead the creator to believe there was a lot of potential for this idea to become a viable project and so it was decided after initial feedback from colleagues that this would be the most rewarding idea to pursue.

After the idea for the Major Project had been generated and research had been carried out it was time to generate some ideas for the project itself to develop an understanding of the project and in turn possibly generate potential ideas for features. **Figure 2.1** and **Figure 2.2** showcase some of the thinking methodologies carried out in the idea generation process.

Michael McKeever



Michael McKeever



2.2 - Requirements specification

Before proceeding with the task of building the product it is first necessary to outline requirements that must be met in order for the product to be deemed successful and to complete within the project deadline. Requirements have been listed into functional and non-functional groups. Functional requirements are necessary requirements that need to be met in order for the product to function as specified. Non-functional requirements are requirements that are not necessary to the functioning of the product and the product will not fail if they are not met but a great effort should be made to ensure they are met and if they are not the reasons for not doing so should be listed.

Requirements are created using a revised version of the Volere Requirements Specification Template to list each of the requirements in a clear and concise manner. The requirements are listed in order of importance based upon their ID, the greater the importance of the requirement the greater the risk it poses to the project if it's not evaluated and achieved. Describing the requirements in minute detail within the Volere Template ensures that they remain realistic and measurable.

Functional & Non-functional Requirements can be located in *Appendix One*

2.2.3 - Further Non-functional Requirements

Search Engine Optimisation

- **Semantic markup** - Using semantic markup and HTML5 attributes ensures that the markup is readable to web crawlers and Google is able to index the website more proficiently.
- **Sitemap** - A sitemap will be created to ensure that the application can be indexed

- **Meta data** - meta data such as author and description content will be created to create better and more relevant searching capabilities with search engines.

Accessibility & Usability

- **Alt attributes** - alt attributes will be used for all images to ensure text can be read by screen readers as an alternative to displaying the image.
- **keyboard navigation** - Tab indexing and input focus styling will be used where appropriate to ensure that the application can be navigated and buttons can be focused and triggered using a keyboard.
- **Responsive** - A mobile first approach will be undertaken to ensure that the application is fully functional on a phone, tablet and desktop computer. Various device and browser testing will be carried out as the product is developed.

Styling & Design

- **Six ups** - These will be created in the paper prototyping stage to start to develop different styles and structures for the outline of content within the application and how it will all come together.
- **Wireframes** - Wireframes will be developed to create an initial overview of the visual design and UX of the product and get a sense of the general structure of the application and how key interactions will occur.
- **Typography** - A typeface will be tested and chosen based on its versatility, style and impact within the application. Moodmemo is a very text focused web application and therefore strong, versatile and professional typography will be key to the design and interaction of the application. A typeface will also be chosen to reflect the style and design choices that are integrated into the product. For instance, a playful design in the product will be reflected with typography that is also playful in design.

- **Colours** - A colour palette will be chosen in unison with the design of the product to reflect a design style that is cheerful, approachable, muted and informative.
- **Brand** - The brand will also be created around the entire design ethos to develop a brand with a design that reflects its values of being undiscriminating, private yet approachable to all.

Security

- **Password Encryption** - All passwords will be encrypted using a 32 character salt and SHA1 hashing.
- **Moodmemo Encryption** - Moodmemos will be encrypted using the PHP 5 mcrypt functionality and a 32 character SALT. They are encrypted before being posted to the database to ensure they are not being stored in plain text and therefore not readable to anyone that gains access to the SQL database tables. The Moodmemos are then decrypted again when being displayed within the timeline and any other aspect of the application that displays them where they are only accessible to the user that created them.
- **SSL certificate** - A SSL certificate will be provided by the hosting company to secure the application behind the https protocol to ensure that all traffic between the web browser and server is secure and therefore provides users a higher degree of trust and peace of mind when using with the product.

Performance & speed

- **Images** - All images should be optimised, compressed and resized using media queries for optimum speed and efficiency.
- **Vector Graphics** - Svgs and icon fonts should be used where appropriate for scalability and file size.

- **Downtime** - The product should be constantly available and accessible with minimal downtime (95 - 99%).
- **User load** - The product will be initially designed to house 500 users with no noticeable performance deduction.
- **Application load** - The product will be able to withstand all users posting moodmemos at once without noticeable performance deduction.

Hosting

- **Domain** - When the name for the application was decided upon research was undertaken to find a domain name for the product. www.moodmemo.co was the domain name that was decided upon and purchased.
- **Hosting** - Hosting is provided through Bigwetfish who provide an excellent rate for reseller accounts for IMD students, they also provide SSL certificates, a dedicated IP address, daily backups and excellent customer support.

Support & Maintenance

- **Maintenance** - Product support should be maintained for as long as possible after it has been developed until all requirements are met.
- **Testing** - All testing and feature development should be done on a production version of the product.
- **Support** - Users will be able to submit their own feedback on the service and links will be provided for getting in touch with the service creator/team.

Social Media

- **Twitter** - A twitter handle was created for Moodmemo at https://twitter.com/Moodmemo_tweets to help promote the service, network with similar services and

provide a point where users can quickly and easily send feedback, get in touch and/or request assistance with the product.

Legal

- Product will fully comply with Data Protection Act 1998.

2.2.4 - Dependencies

A list of dependencies has been created that both the requirements and application itself will rely on if to be met with success:

- The application will not function if the user does not have a working internet connection.
- In order for a user to create Moodmemos and access the service they must first create an account.
- The user must have javascript enabled in their browser in order to create a Moodmemo, as the timeline is generated using javascript.
- The web application must be able to establish a connection with the database in order to validate a user's login credentials and log them into the service and also to access their Moodmemo information so it can be retrieved and generated.
- A user's location cannot be plotted on a map if their browser does not support location tracking or they do not allow location tracking. However a user is still able to create a Moodmemo even if their location is not shared.
- A user must access the application from a modern browser or it may not appear or function correctly. E.g. The application does not support Internet Explorer 8 or below.

2.3 - Paper Prototyping

The next stage in the project involved paper prototyping. Paper prototyping is a largely used procedure with designers and developers to ensure that each project is being approached rudimentarily. It is a lean and rapid method used to determine various features of a project such as the core structure, initial layout, navigation, key interactions, interface design etc. It is a key technique in recognising and determining potential problems within the application at an early stage.

FIGURE 2.3.1 - INITIAL RUDIMENTARY SITEMAP

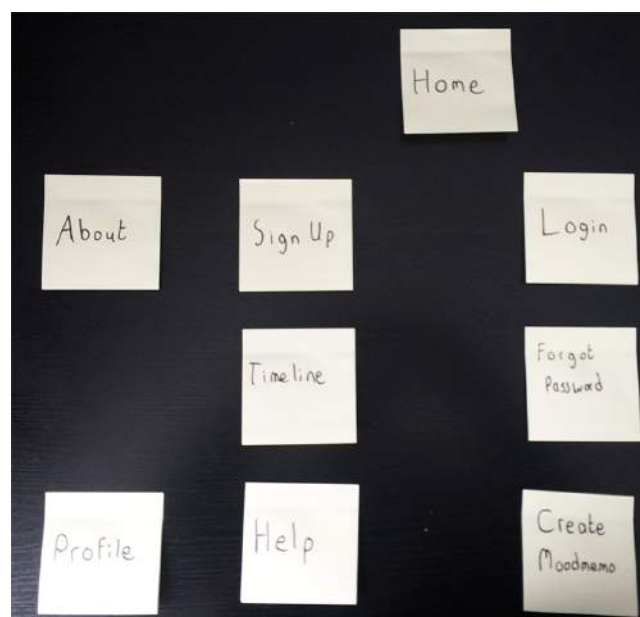
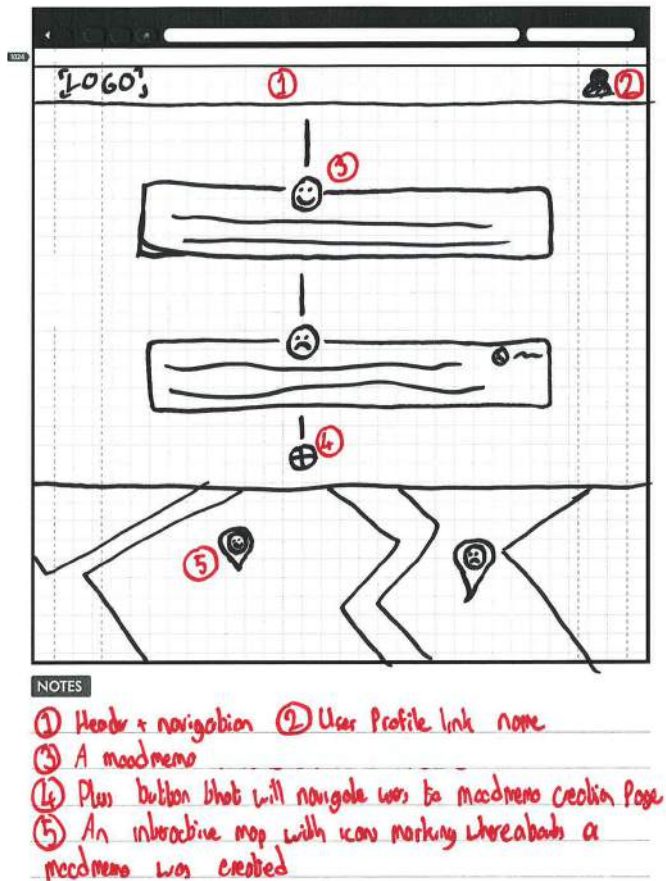


Figure 2.3.1 demonstrates the initial thinking of how the product would be structured and was carried out to create a visual sense of key aspects of the application, such as the creation of a moodmemo and creating an account.

Figure 2.3.2 displays an image of an initial sketch that was created demonstrating a preliminary overview of the main interactions with the product. It displays an initial preview of how the timeline could appear and with the information that could possibly be displayed within it. It also displays the beginning of the map interface and

how a user might interact with this element and how Moodmemos would be presented within it.

FIGURE 2.3.2 - INITIAL SKETCH



2.3.2 - 6-Ups

6-Ups is a technique of generating six possible solutions to one design problem. It's an excellent method for rapid prototyping an idea and limiting the process to 6 different solutions means that a lot of thought needs to be put into each solution and how it differs from the others. *[Downes, J. (2010)]*

Figure 2.3.3 shows the 6-Ups that were created to visualise different abstractions of the Moodmemo feed:

- **Prototype 1:** Features Moodmemos being displayed in a timeline with a user profile modal to the right and map with Moodmemo marker below.
- **Prototype 2:** The Moodmemo creation process is accessible directly from the header with the timeline being featured below it.
- **Prototype 3:** Created Moodmemos are displayed within a large interactive calendar where each day features the Mood emoticon that has been associated with it. This approach would involve limiting the user to only creating one Moodmemo per day.
- **Prototype 4:** Prototype 4 gives the map more prominence by displaying it above the Moodmemos and providing it with more screen real estate. Moodmemos are displayed in a horizontal list rather than timeline, with the Mood emoticon being given more prominence also.
- **Prototype 5:** This prototype doesn't feature header navigation but rather the navigation is located within the sidebar. Moodmemos are displayed within the home page in a horizontal, sortable list along with statistics and charts.
- **Prototype 6:** Prototype 6 is similar to Prototype 1 but the timeline is listed down the centre of the screen with Moodmemos being sorted left to right down the timeline.

FIGURE 2.3.3 - 6-UPS DESIGN FOR MOODMEMO FEED

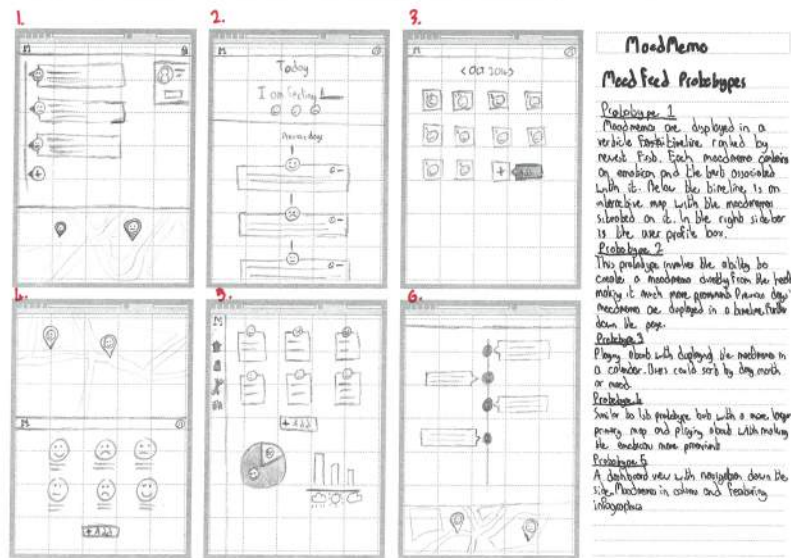


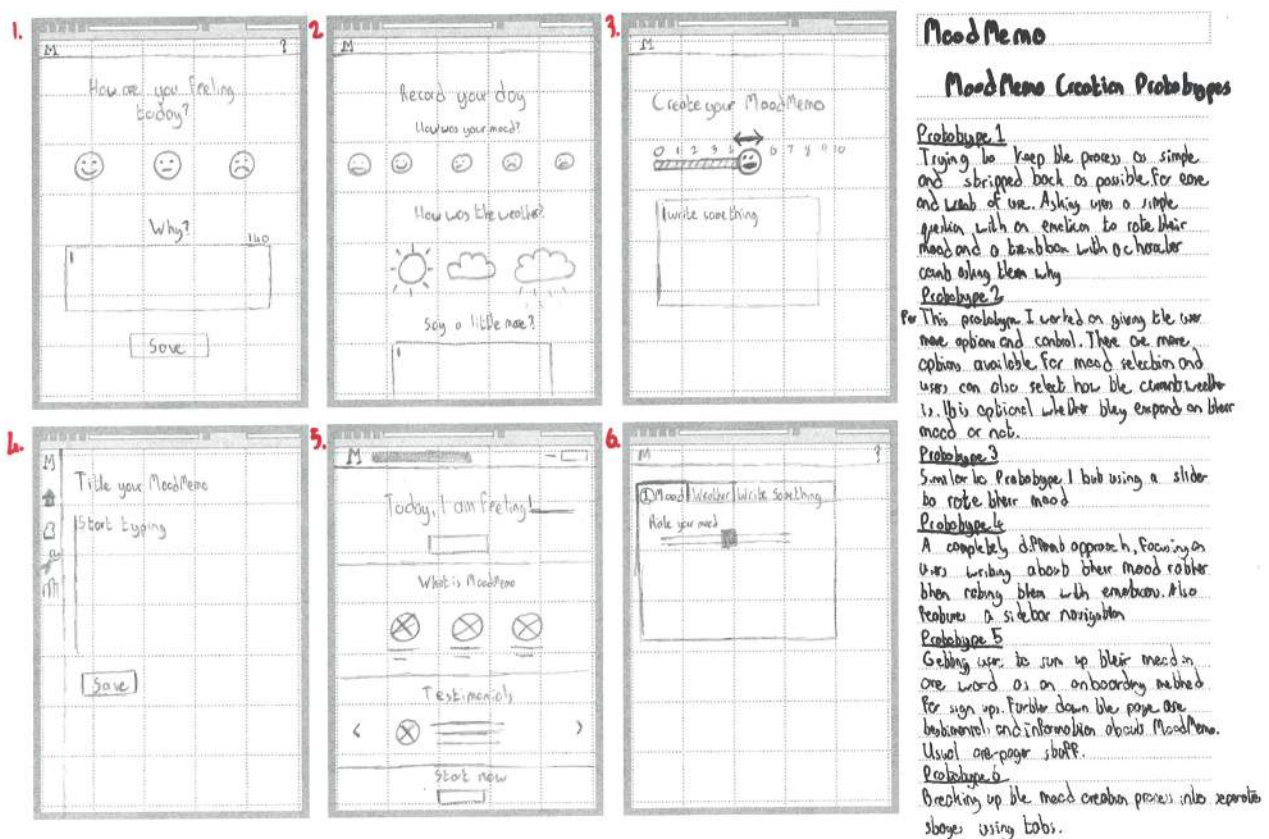
Figure 2.3.4 features a 6-Up created for the Moodmemo creation process, as this is potentially the most important interaction that users will make on the application it was important to get this right. The fundamental idea was to keep the process of creating a Moodmemo as simple, straight forward and non-constrained as possible for the user. This was taken into account when creating the 6-up prototypes:

- **Prototype 1:** Two steps are used in this prototype for the creation of a moodmemo. Selecting an emoticon to reflect the user's current mood and a textbox limited to 140 characters that asks them why they feel that way.
- **Prototype 2:** The 2nd prototype focuses on asking a user to 'record their day' by gathering three bits of information. 1. Their mood. 2. How the weather was and 2. A textbox where they can enter details. This prototype would involve a user recording their mood once a day.
- **Prototype 3:** Similar to Prototype 1 except the user has more control over selecting their mood. Instead they are presented with a slider and asked to rate their mood out of 10 possible ratings. 10 being the happiest.
- **Prototype 4:** Prototype 4 is a much more simple prototype of creating a Moodmemo. There is no mood section but two input boxes, one to title their

Moodmemo and another to type whatever thoughts they have, with no character limit.

- **Prototype 5:** This prototype would involve creating a Moodmemo directly from the timeline page. Users complete the phrase “today I am feeling ...” and then submit their Moodmemo. An emoticon is selected based on the value they submit.
- **Prototype 6:** This prototype involves breaking up the Moodmemo creation process into different stages. First a user selects their mood, then the current weather, then they expand on their current mood with some text.

FIGURE 2.3.4 - 6-UPS DESIGN FOR MOODMEMO CREATION



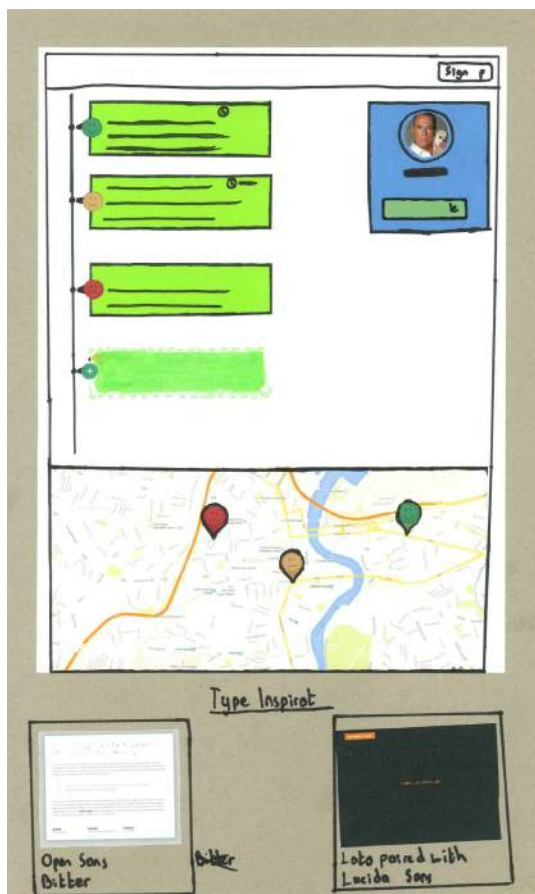
2.3.3 - High fidelity Mockup

Once the 6-ups were created, the options were considered, evaluated and a design solution had been constructed, a high fidelity sketch was then created to take the constructed solution and look at it in more comprehensive detail. The creation of the

high fidelity wireframe allows for more thought to be given to elements such as the structure, layout, design, colour, typography and hierarchy of the prototype.

Figure 2.3.5 shows a high fidelity sketch of a user's timeline feed with listed Moodmemos and their associated emoticons, a user profile modal and the interactive map with positioned Moodmemo markers. Also demonstrated is consideration given to possible typography layouts and hierarchies which is expanded further upon in the creation of the Styletiles.

FIGURE 2.3.5 - HIGH FIDELITY MOCKUP

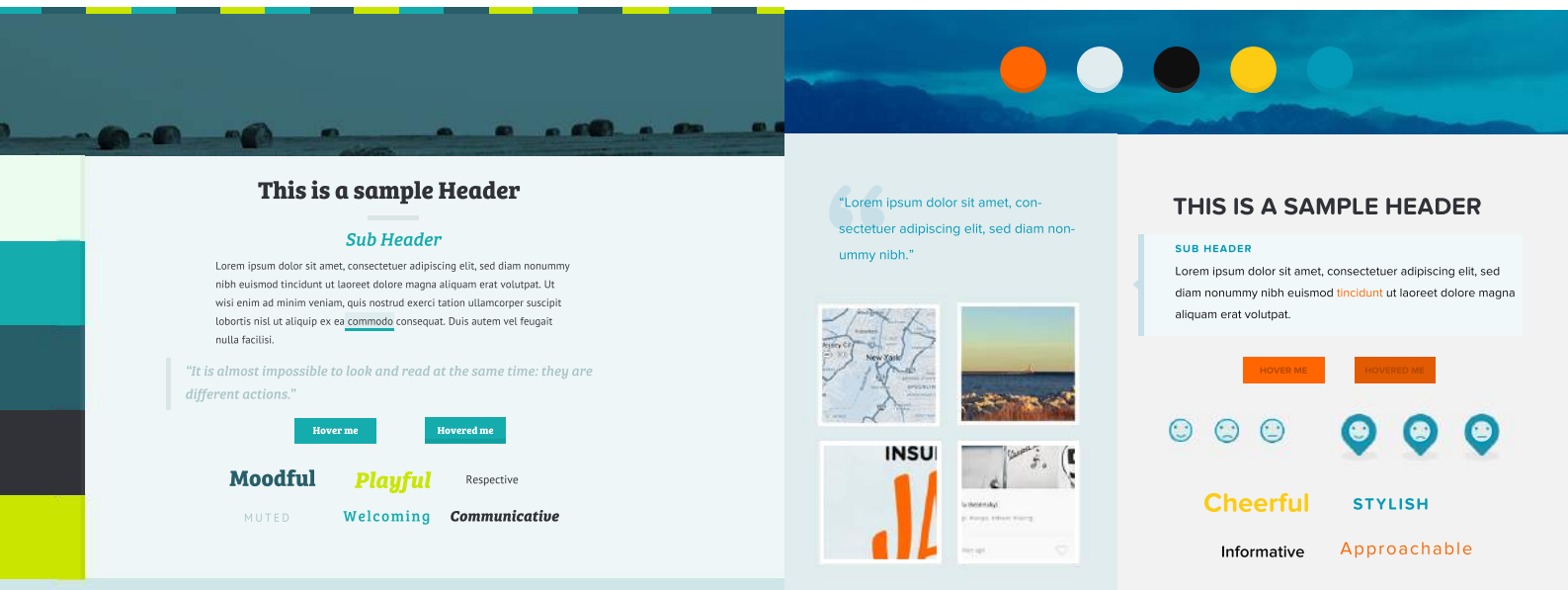


2.3.4 - Styletiles

Styletiles are a design deliverable consisting of fonts, colours and interface elements that communicate the essence of a visual brand for the web. *[Styletil.es, (n.d.)]*

They're a useful tool in helping to form a visual language for the product before approaching the mockup stage of the project. Styletiles are important for establishing a direct connection with elements of an interface without having to define a comprehensive layout.

FIGURE 2.3.6 - STYLETILE V1 & FIGURE 2.3.7 - STYLETILE V2



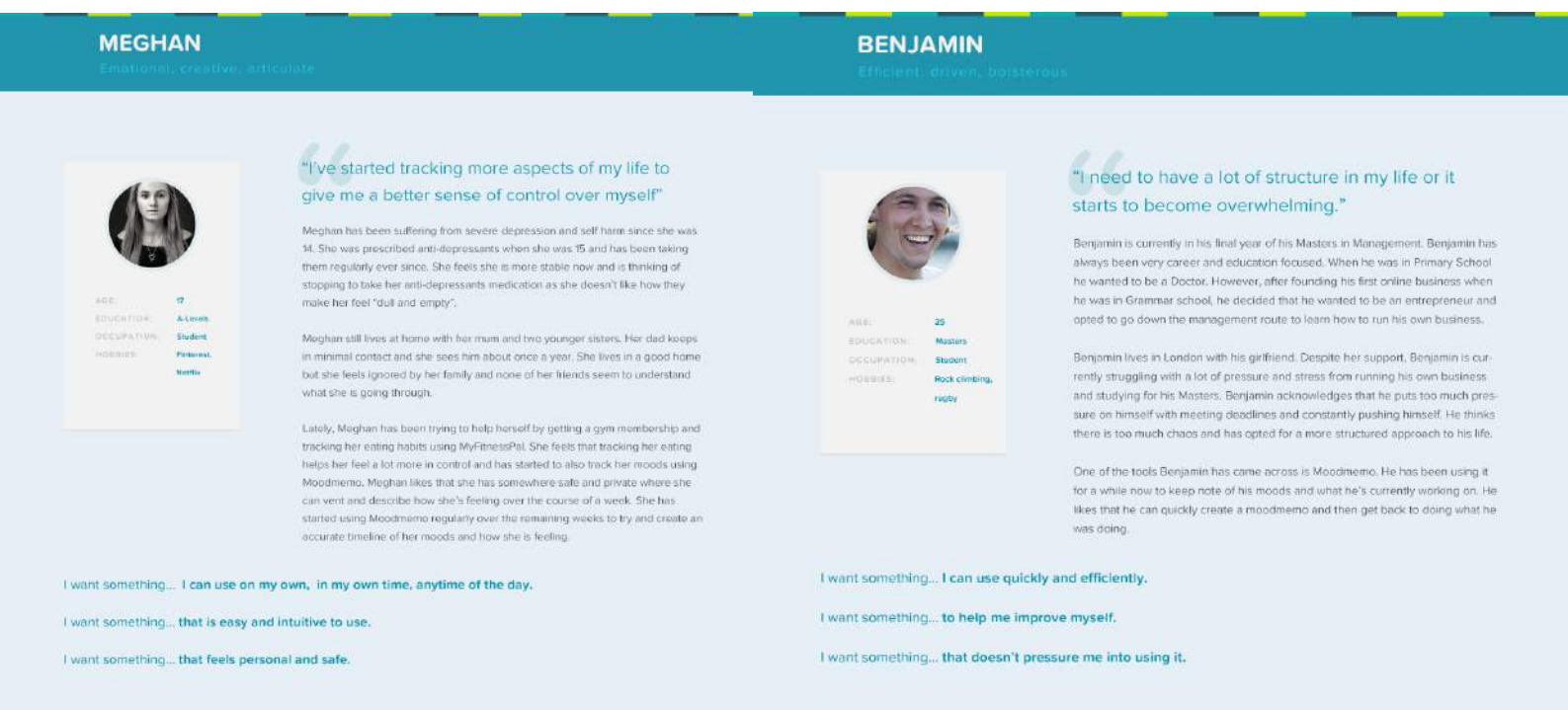
Two Styletiles were created to provide two different approaches that are each unique and evoke different design styles and moods. Styletile v1 as seen in **Figure 2.3.6** evokes a calming sense of style that features muted design patterns to help elicit a visual brand that is welcoming, communicative and respectful. Styletile v2, visualised in **Figure 2.3.7** takes a different approach and serves to create a style that is more cheerful and stylish but also induces a visual brand that appears informative and approachable to the user. These are each key attributes to address for the visual style of the application considering the sensitive subject matter.

2.3.5 - User Personas

User Personas are a method of communicating research about people within the target market for the product. It is depicted as a specific person but fictional. It's a tried and tested method for helping designers to focus on a manageable and memorable portion of the target market rather than attempting to focus on every single person within the target market. They aid designers in developing an idea or product for a specific somebody rather than a generic everybody. *[Goltz, S. (2014)]*

Two user personas were developed for Moodmemo to give a sense of the people that could be using the product. This is beneficial for being able to better visualise the people that may use Moodmemo and how to design the application to cater to them.

FIGURE 2.3.8 & FIGURE 2.3.9 - USER PERSONAS



Two personas were created, one female(Meghan) and one male(Benjamin) each with their own widely different personalities, interests and goals.

Creating these two user personas helped to better understand how a user might approach using the product. For instance, some key points taken from this exercise is that the application should feel personal and safe. It should also not pressure user's into using the product or keeping active. Users should be left to use the product at their own pace.

Text versions of the User Personas can be found in *Appendix Four*

2.4 - Feasibility Testing

Feasibility testing is the practice of evaluating an idea or multiple ideas and helping to distinguish if the idea is practical or otherwise. And if not, providing and evaluating alternative measures.

The aims and objectives for Moodmemo had been outlined and it was now time to test the feasibility of the application's goals and determine if they are realistic or not.

Research was first carried out to ensure that the idea of Moodmemo itself was feasible. Verbal feedback was gathered from both experienced designers and developers from within the industry in Belfast to gain some key insights into the feasibility of the project from a design and development standpoint. The founders and designers of Little Thunder Co., Gabriel Muldoon and Tim Potter both agreed that the project was feasible and did feel that there was a need for a tool such as Moodmemo. Head developer and CTO at Get Invited, David Turner provided initial feedback on the technical and development aspects of the project, stating that the project was indeed very feasible from a technical point of view and providing a short breakdown of the technical tasks that are apparent.

This feedback ensured that the technical tasks of the project could be evaluated and ranked in order of their importance to the outlined requirements of the application to ensure that each could be met. For instance, contingency time was estimated based on research that would need to be carried out and knowledge that would need to be learned in order to complete tasks such as the creation of a Moodmemo, the generation of a timeline, the creation of a user registration and login system, the development of a secure database and the feasibility of an interactive map based on users' locations.

2.5 - Methodology Testing

A number of different methodologies were considered that could be utilised for the project which would dictate the process in how the development of the project would be structured and how tasks would advance.

2.5.1 - Waterfall Methodology

This methodology is better suited for smaller projects and undertakings where the end goal is clear and well defined. There are 6 stages to this process to arrive at the end result:

1. Requirement Analysis
2. Design
3. Coding
4. Integration
5. Testing
6. Deployment

Advantages

Given that this project would be considered a small project this methodology bodes well to it with its well-defined stages, allowing for an effective and well-structured time scale to be used.

Disadvantages

Having testing at the very end of the project when time is nearing completion comes with some risks. There may be a major flaw in the project that went undiscovered until the testing phase thus not leaving enough time to fix it. Many people also find that shipping a product as fast as possible and then developing it from there is a better way of building a product. However those ideals don't play nice with this methodology.

2.5.2 - Prototyping Methodology

The prototyping methodology refers to creating an example prototype before fully setting out on the project to better understand the aim and objectives of the project. They then build upon the project using the newfound objectives. This iteration is applied several times until a product is created that meets the aim.

Advantages

The main advantage to this methodology is getting a prototype out as quick as possible and expanding upon it over iterations. This means a lot more user involvement from an early stage of the product development and potentially a much more useful tool being produced at the end.

Disadvantages

As there are still a lot of unknowns and discrepancies with this project, not to mention a report to write alongside the development, I'm not sure how well that would favour with this methodology.

2.5.3 - Chosen Methodology

Waterfall was the chosen methodology after the research had been carried out. Its process model is well suited for smaller projects such as Moodmemo where the final outcome had been visualised and the key requirements were outlined. The methodology was adhered to throughout the project to ensure the project did not fall off course and risk the possibility of missing requirements and deadlines.

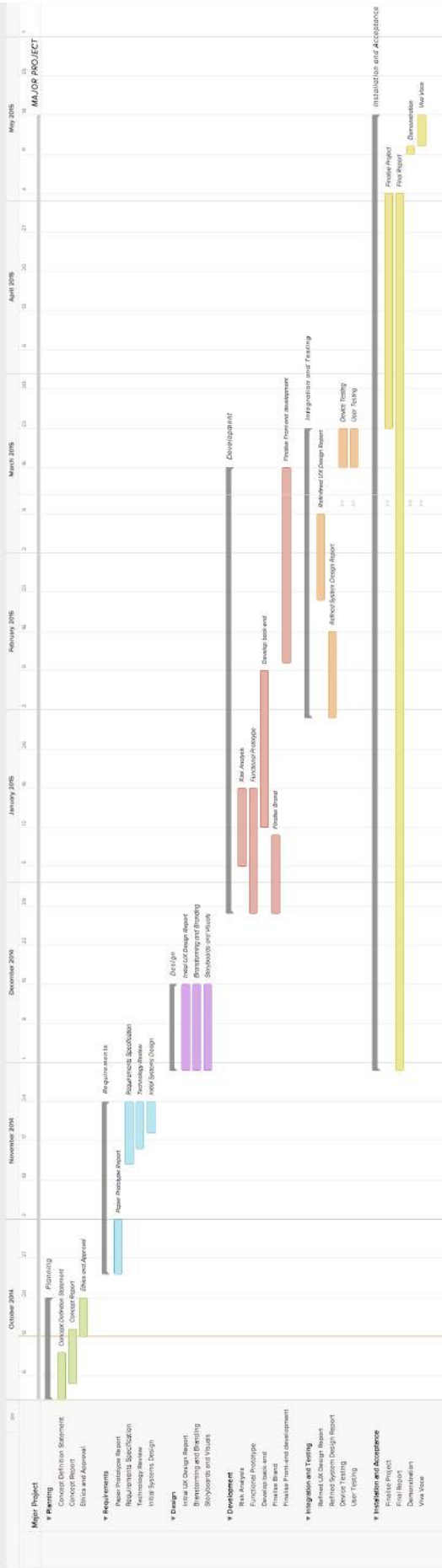
Figure 2.5 displays the gantt chart that was created for the project. It can clearly be seen how the Waterfall methodology was utilised for the breakdown of tasks within the project. All tasks were broken down and defined into the following stages:

1. Planning
2. Requirements
3. Design
4. Development
5. Integration & Testing
6. Installation & Acceptance

This was to ensure that tasks could be accomplished more efficiently as stages were completed and time progressed on.

A consideration that was adhered to regarding the Waterfall methodology was the possibility of a task, feature or requirement not being met and this not becoming apparent until within the final stages of the project. This was avoided by evaluating the progress at the end of each stage and ensuring that requirements were being met and features were performing as intended.

FIGURE 2.5 - MAJOR PROJECT GANTT CHART



3.0 - Design

This section illustrates the evolution of the design process throughout the project, and discusses and justifies used design principles.

3.1 - Branding

The Moodmemo brand was created with the target audience and visual brand style in mind. Considering the subject matter, and the overall style that had been created thus far; key terms were taken into account when creating the brand to ensure it met the overall ethos of Moodmemo that had been realised. The brand needed to reflect:

1. Approachability
2. Cheerfulness
3. A modern style
4. Respectiveness
5. A welcoming nature

FIGURE 3.1 - REVISED BRAND

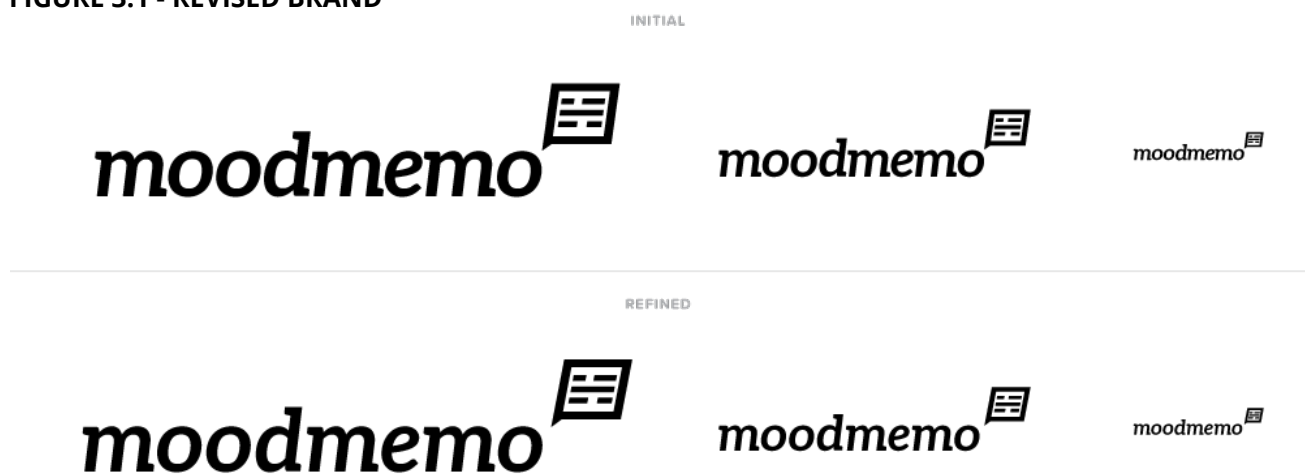


Figure 3.1 shows the finalised Moodmemo logo. Minor adjustments were made between the initial and refined logo to ensure the logo portrayed professionalism and

displayed better at smaller sizes. For instance, the kerning between the letters was loosened slightly and the logo mark was given a steeper angle and equal spacing.

The logo needed to reflect professionalism and boldness whilst also being cheerful and approachable. A slab serif typeface was used to reflect the approachable and modern style of the brand whilst the logomark was designed with sharp, thick and bold lines to illustrate a Moodmemo and reflect the boldness and professionalism in the brand to give users more confidence in the product.

3.2 - Typography

All typography, excluding the logo, within Moodmemo is set with the typeface **Proxima Nova Soft**, created by Mark Simonson. This is the rounded version of the very popular typeface, Proxima Nova. *[Marksimonson.com, (n.d.)]* The soft, rounded letters of Proxima Nova Soft were perfect for the Moodmemo brand and evoking a stylish, cheerful mood for the application. It's optimised for screen use and features robust ligatures, symbols, and extended language support ensuring the typeface is extensible and robust enough for a web application with a large scope.

FIGURE 3.2 - MOODMEMO TYPOGRAPHY (WWW.STYLEGUIDE.MOODMEMO.CO)

The quick brown fox jumped over the lazy dog

```
h1 { font-size: 5rem; }
```

The quick brown fox jumped over the lazy dog

```
h2 { font-size: 4.2rem; }
```

The quick brown fox jumped over the lazy dog

```
h3 { font-size: 3.6rem; }
```

The quick brown fox jumped over the lazy dog

```
h4 { font-size: 3.0rem; }
```

The quick brown fox jumped over the lazy dog

```
h5 { font-size: 2.4rem; }
```



3.3 - Voice & Tone

The voice and tone used in an application is very important. It's the communicative voice between the brand and the user. In relation to Moodmemo, the voice and tone must be informative and rewarding but also playful. MailChimp has been a large source of inspiration in the creation of Moodmemo and when it came to voice & tone design, this was no exception. MailChimp created a voice and tone guide to be used within their company to ensure that a global tone was being portrayed, and that the language patterns were consistent with the brand. *[Voiceandtone.com, (2013)]*

A voice and tone was adhered to when creating language within the app that would involve interactions with the user. One of the most obvious use cases for this is within feedback messages, as displayed in **Figure 3.4**. A friendly voice is used along with the Moodmemo emoticons to display to users when a positive or negative interaction occurs.

FIGURE 3.4 - EXAMPLE OF MOODMEMO FEEDBACK MESSAGES (WWW.STYLEGUIDE.MOODMEMO.CO)



3.4 - Colour

The finalised colour palette was a refined combination of both of the created Styletiles. The colour palette consists primary of a blue palette with contrasting colours for gaining attention. The palette was chosen to reflect a muted but stylish and welcoming atmosphere within the application. Three palettes are used, as seen in **Figure 3.5** and **Figure 3.6**. A primary palette, which consists of 4 colours which are used extensively throughout the application for primary actions such as buttons, links, background colours etc. The typography palette consists of colours that are used for

all text elements within the application and the border palette is used for the border pattern used throughout the application.

FIGURE 3.5 & FIGURE 3.6 - COLOUR PALETTE (WWW.STYLEGUIDE.MOODMEMO.CO)



3.5 - Empty States & Onboarding

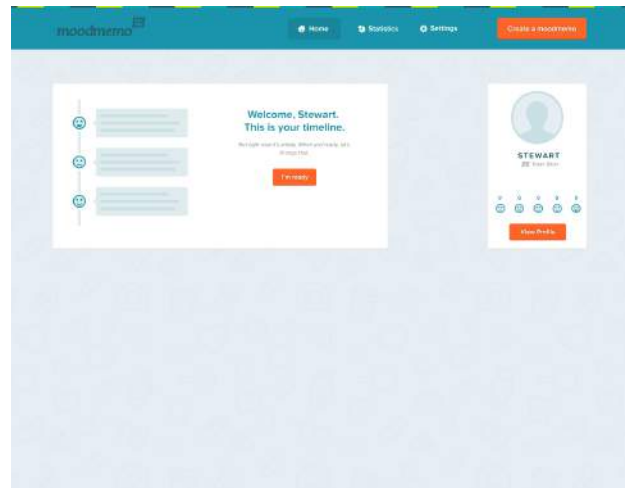
An “empty state” is the state of an application when no data is present. *[Pearson, C. (2014)]* In other words, it is what a user will see the first time they use the product. The first time a user visits the app there will be no Moodmemos, statistics, or map data to present. An empty state is a critical part of UX, it provides an opportunity to educate users in how to use your application when no data is present and increase their overall happiness with the experience of using the product. The first-run of an application is when lasting impressions are made by the user, hence why this opportunity should be used to help educate the user and provide instructions on how to use the application.

3.5.1 - Timeline iterations

Figure 3.7 shows the initial timeline mockup design reflecting how a timeline will appear when populated with user data, along with a user profile photo. **Figure 3.8** visualises the screen a user will see when they first login to Moodmemo or if no Moodmemos are present within their timeline. User’s are welcomed and then,

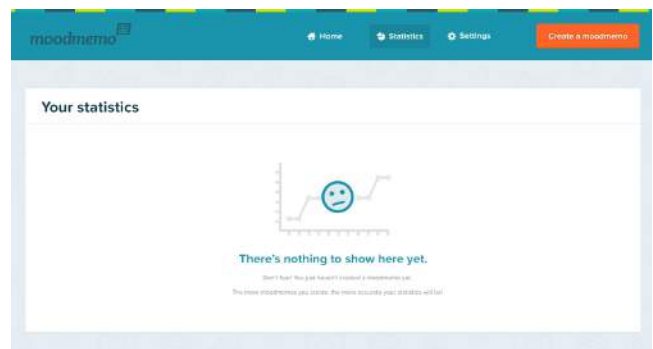
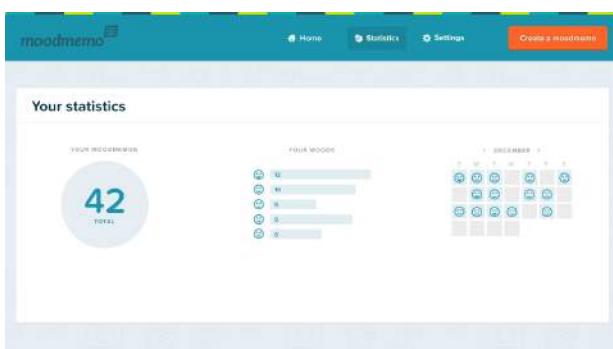
following the outlined voice & tone principles, a friendly message is displayed explaining that this is their timeline but its blank and they can fix that when they're ready.

FIGURE 3.7 - INITIAL TIMELINE MOCKUP & FIGURE 3.8 - REFINED TIMELINE WITH EMPTY STATE



3.5.2 - Statistics

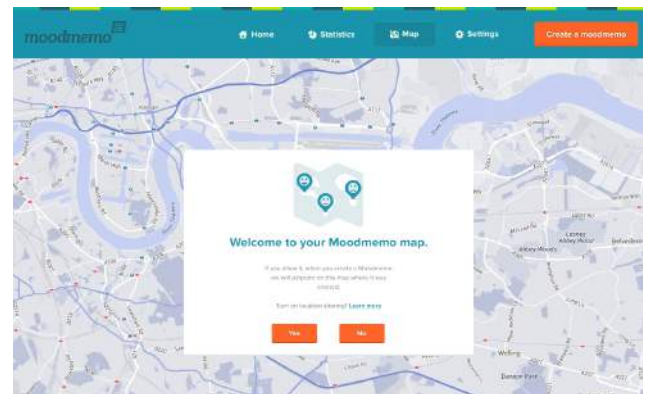
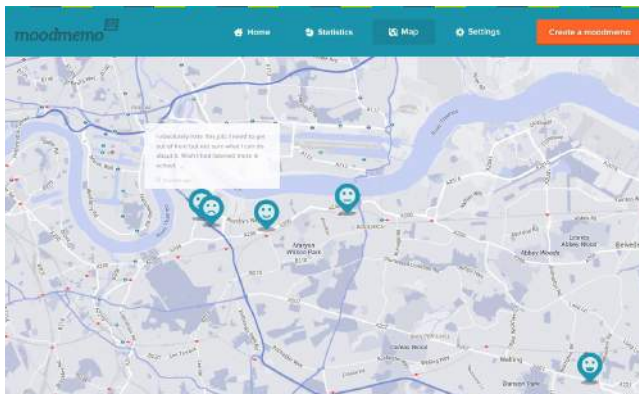
FIGURE 3.9 - INITIAL STATISTICS MOCKUP & FIGURE 3.10 - REFINED STATISTICS WITH EMPTY STATE



The statistics page will contain infographics and statistics on a user's moods and moodmemos. However, If a user hasn't created any moodmemos, all they will see is a page of empty charts and stats. Instead, a better experience to the user would be if they are informed why these charts are empty and what they can do to start populating them.

3.5.3 - Map

FIGURE 3.11 - REFINED MAP MOCKUP & FIGURE 3.12 - REFINED MAP WITH EMPTY STATE



The maps page will be responsible for displaying a user's created Moodmemos on a map, if they have allowed location sharing. However, if a user has not created any Moodmemos they will be presented with an empty map. Instead an empty state will be designed that displays a modal when a user first visits the page explaining what this page is for and how to use it.

3.6 - Page Errors

Error pages are an important addition to an application in order to inform the user when an error is created. The three main error pages that will be present within the application are:

1. **404 error** - Displayed when a user tries to access a page that does not exist.
2. **403 error** - Displayed when a user tries to access a page that they do not have the permissions to access.
3. **500 error** - The generic Internal server error that displays when something has gone wrong when a user tries to create a server request.

FIGURE 3.13 - EXAMPLE MOCKUP OF A 404 ERROR

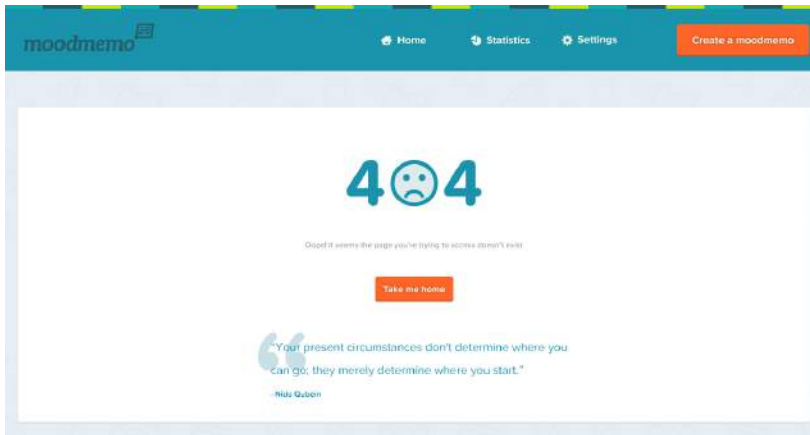


Figure 3.13 demonstrates a 404 error page that will be present within the application. Being met with errors causes frustration for users and therefore the mood was kept informative and light when designing the error. An inspirational quote has also been used within each Error page as a “delighter” that helps to increase a better and more happy experience for the users, even when being met with an error page.

3.7 - Online Styleguide

An online style guide is a living document of code, which details all of the various elements of an application. It is used as a means to collate and consolidate the front end code, and visual language of the product, such as the button, typography, links, form input styles and colour palettes used throughout the application. Its a one-stop place for designers and developers to reference when discussing site changes and iterations. *[Robertson, S. (2014)]*

A style guide has been created for Moodmemo in order to collate a place where the elements of the application can be viewed and evaluated in one place but also as a contingency plan if ownership of the application is passed on or if a developer/ designer is brought onboard for further iterations of the project.

The style guide is further added to and developed as time goes on, it is live and viewable to anyone. It can be viewed here:

<http://styleguide.moodmemo.co/>

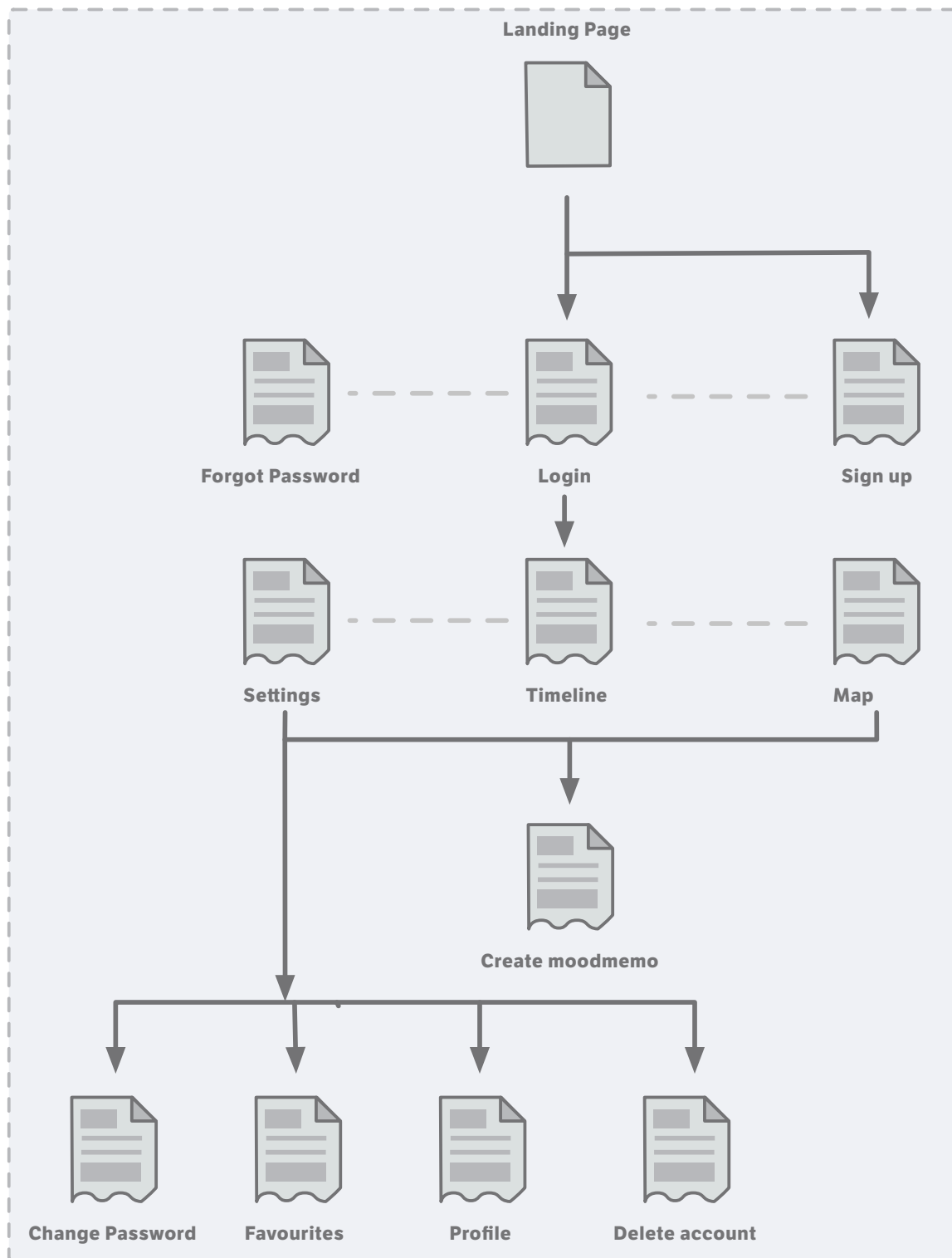
4.0 - System Design & Architecture

After the completion of the UX design phase, the next stage involved the creating and planning of the system design for the project to outline all aspects of the software and ensure a robust foundation was built for the application to run on.

4.1 - Refined Site Map

Initially it was stated that the Moodmemo timeline and Create Moodmemo page would be the most crucial in regards to user interaction of the product and therefore should be easily accessible and also that the majority of the product features should be accessible from the timeline page. These points still stand, however, during the prototyping phase the initial site map of the product was revised as the structure of the site became finalised. A revised sitemap, **Figure 4.1**, has been created to reflect this:

FIGURE 4.1 - REVISED SITE MAP

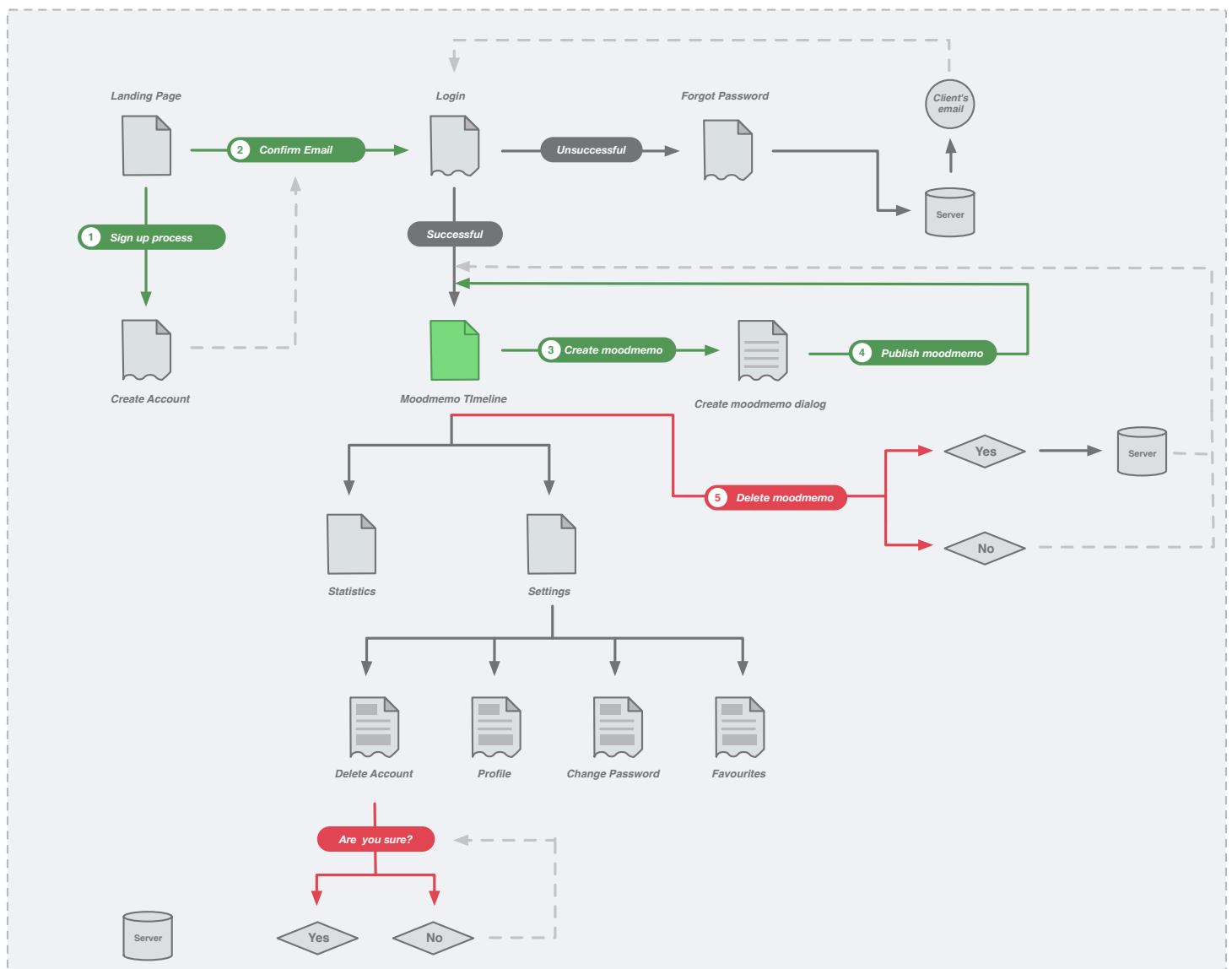


4.2 - User Flow

A user flow diagram, **Figure 4.2**, was also created to summarise a user's journey in using the product and what actions he/she may carry out such as the following:

1. The sign up process/creating an account
2. Activating the Account via email
3. Logging in to their account
4. Forgetting their password and having a new one sent via email
5. Creating a moodmemo
6. Publishing a moodmemo
7. Deleting a moodmemo
8. Deleting their account & data

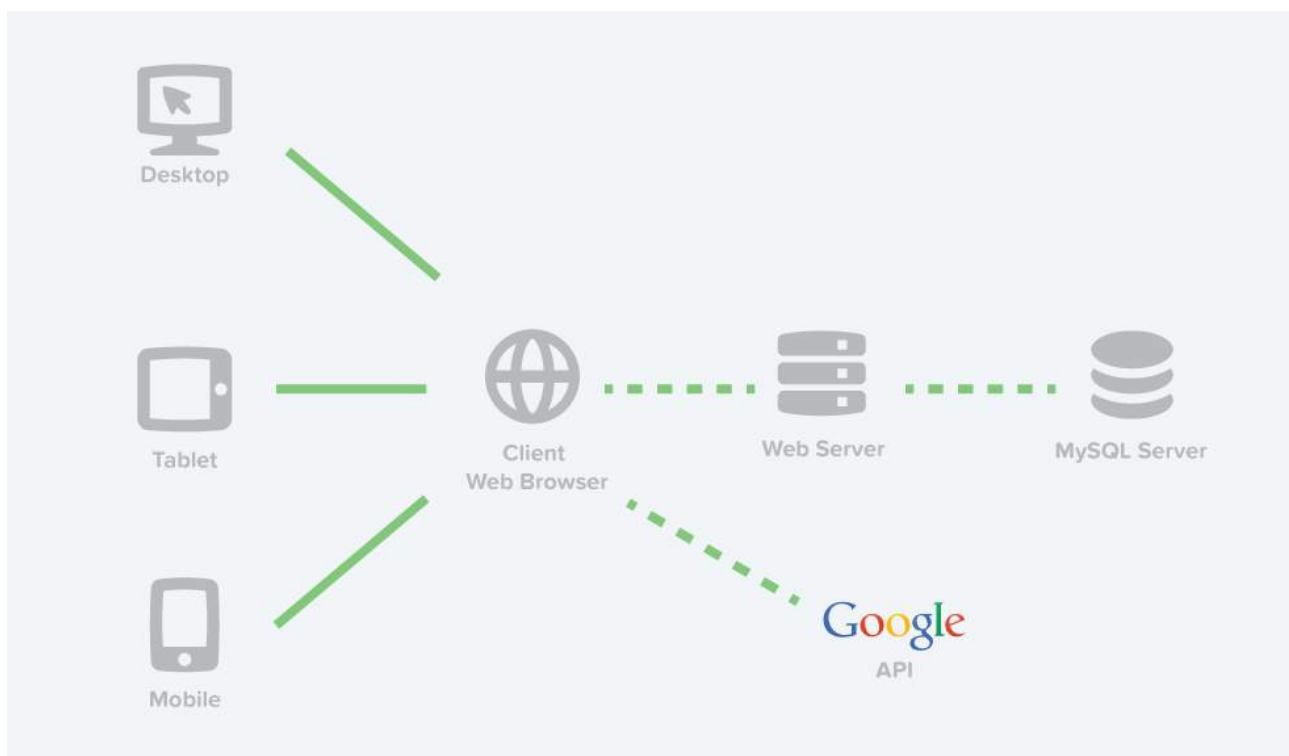
FIGURE 4.2 - USER FLOW DIAGRAM



4.3 - Platform Architecture

A diagram was created, **Figure 4.3**, to reflect the platform architecture of the application. A platform architecture diagram illustrates an application's structure from a hardware point of view. It conveys how information and data is sent across the web application and how a client might access the product. For instance, a client can access Moodmemo from a range of devices through a web browser of their choice. The web browser talks to the web server which hosts the html, css, javascript and php. The browser connects directly with external libraries and APIs such as Google Maps API and the jQuery library without having to access the web server to do so. The web server will connect with the MySQL server when receiving and sending data such as user registration/login details, or when a new moodmemo is created.

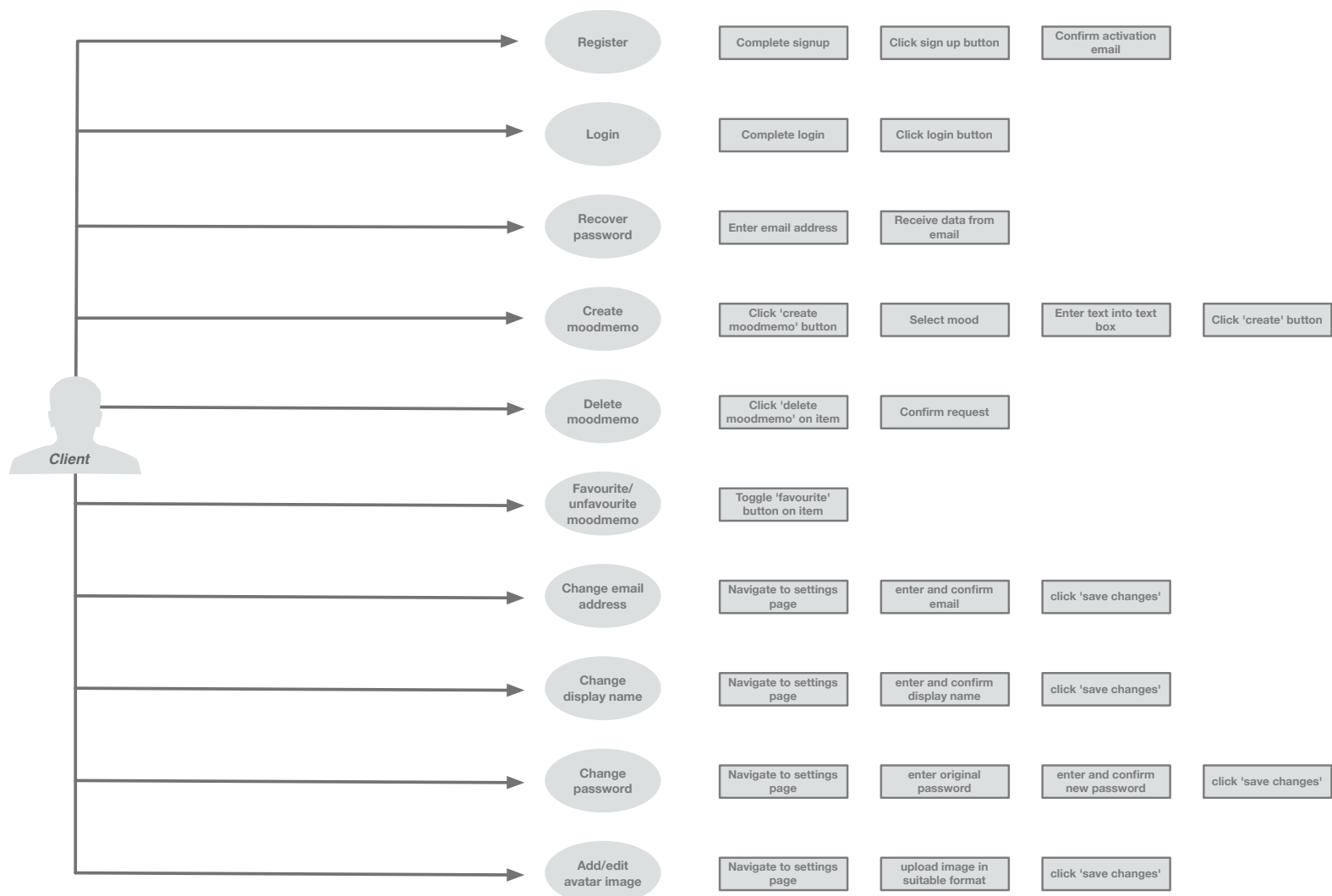
FIGURE 4.3 - PLATFORM ARCHITECTURE DIAGRAM



4.4 - Use Case Diagram

Use cases were previously created to outline how a user might carry out a given task on the application. This was to ensure that each user process was considered and outlined. The previously outlined use cases have not been altered, however some have been expanded upon or added to outline refined processes within the product. A diagram, **Figure 4.4**, was created to illustrate the main use cases and the main steps involved in setting them.

FIGURE 4.4 - USE CASE DIAGRAM



4.5 - Logical Design

Logical design is a method of looking at a system through an abstract representation of the data flows, inputs and outputs of the system. It involves enforcing a logical structure to programming to make a system more efficient, easily modifiable and understandable.

It was decided that creating diagrams to illustrate the structure of the web application and the relationships between key elements of it would be an ideal method for helping to visualise the architectural structure of the product and would benefit when creating the database and forming relationships between tables.

Two diagrams were created to illustrate the main actions which will take place within the product:

1. The login/sign up process
2. The moodmemo creation and deletion process
3. The user profile edit and delete process

FIGURE 4.5 - LOGIN/SIGN UP PROCESS

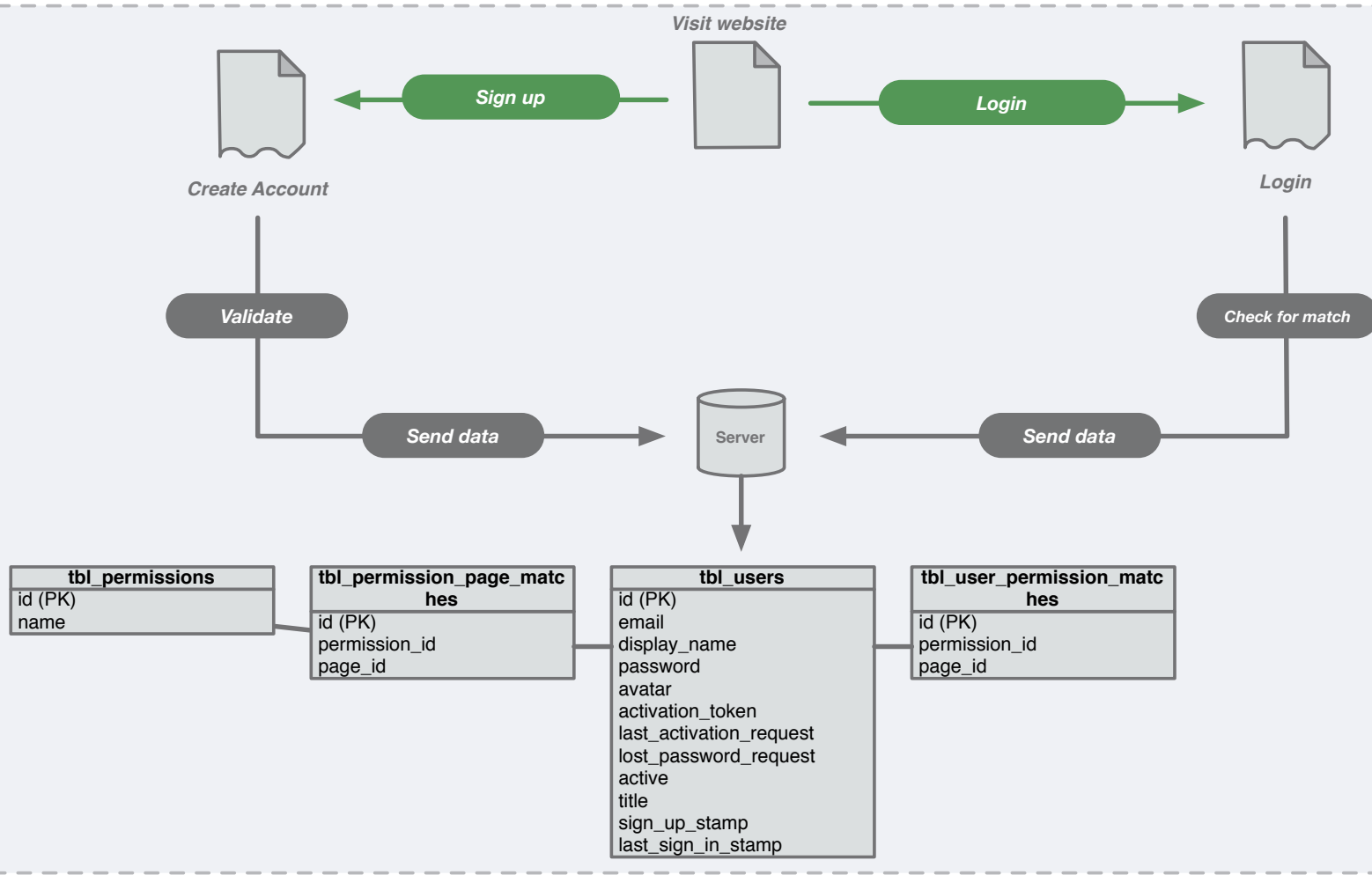
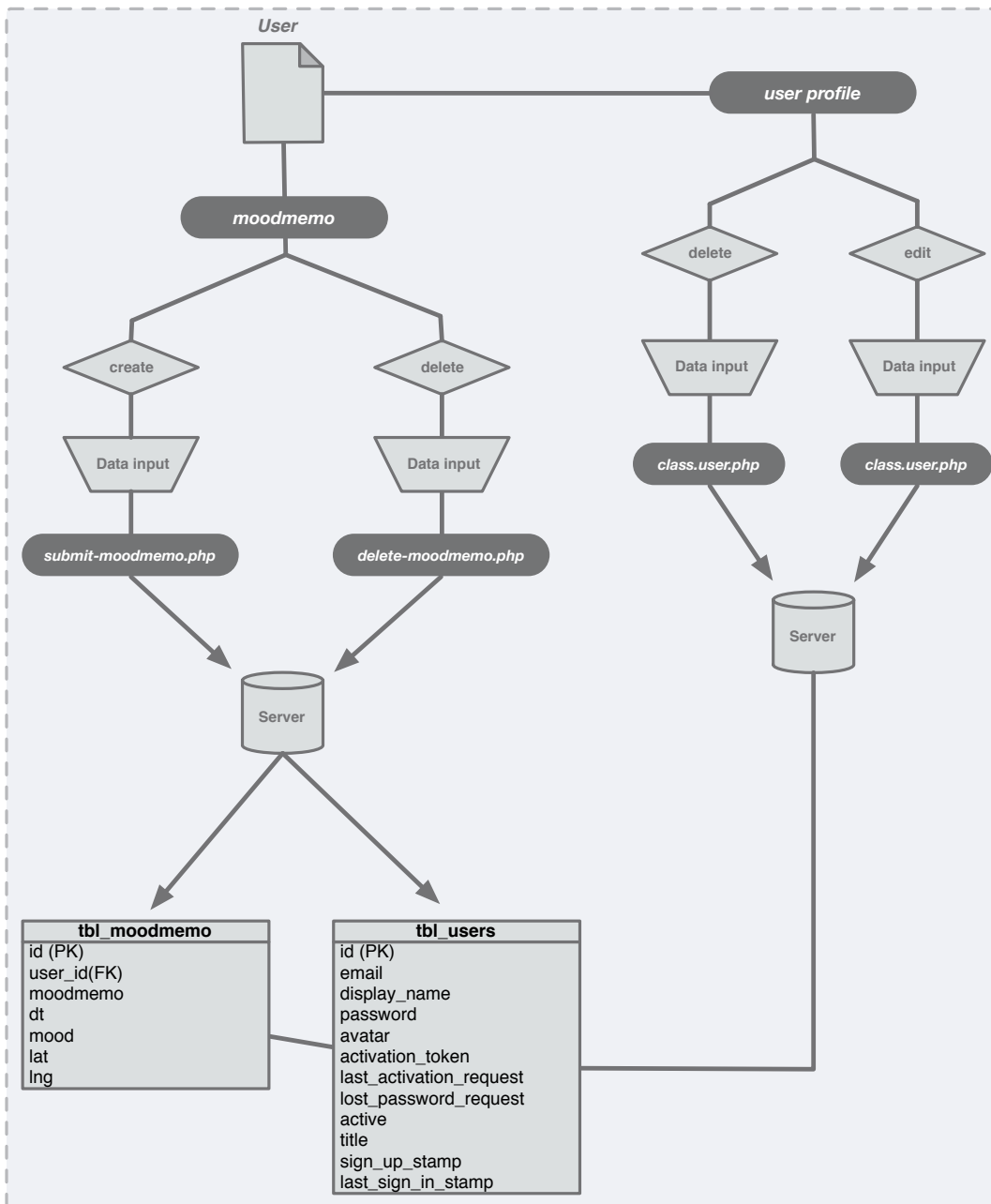


FIGURE 4.6 - MOODMEMO & ACCOUNT INTERACTIONS

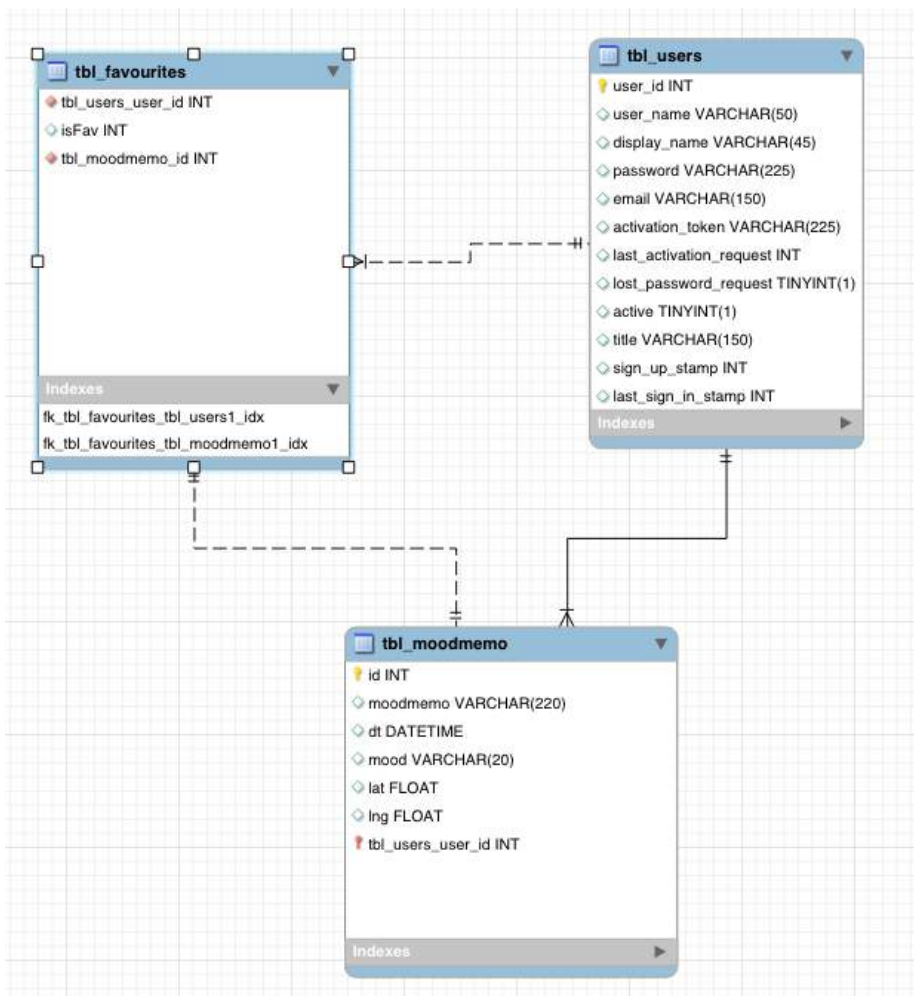


4.6 - Database Design

A database was an essential feature for Moodmemo to operate successfully and store client data. Therefore, the design of the database had to be carefully considered and the tables and entities that would be present in the database should be considered and identified. In order to aid with the design of the database a data model was to be created. The tool MySQLWorkbench was used in this process to maximise the efficiency in designing a database with its intuitive features and interface.

Once the tables and their entities were considered it was then necessary to determine which entities would be primary keys and what datatypes would these entities carry. It was also necessary to determine the entity relationship between the tables. This was done with the creation of an Enhanced Entity Relationship Diagram (EERD):

FIGURE 4.6 - EERD DIAGRAM



For the EERD, three tables were considered:

1. The favourites table will act as a join table between a user ID and Moodmemo ID
2. The users table which will contain information on users and their accounts.
3. The moodmemos table which contains information regarding created moodmemos.

The user table is linked to the favourites table via the user_id. One user can have many favourited Moodmemos therefore the relationship is one-to-many.

A user can create many moodmemos therefore the relationship between the users and moodmemos table is one-to-many.

One Moodmemo can only be favourited once, therefore the relationship between tbl_moodmemo and tbl_favourites is one-to-one.

4.7 - Technology Selection

Research was previously carried out into which technologies would be utilised in carrying out the build of the project. It was important to carry out this research to develop a further understanding of the appropriate technologies to be used when undertaking the build of an interactive web application. The selected technologies have been noted below with documented justifications.

HTML5 - HTML5 is the final and complete fifth revision of the HTML language. It's supported by all browsers, more semantic than the previous version of HTML and has support for many features and built in APIs. HTML5 has been used as the core markup language for this project. This allowed for more semantic markup that benefits modern web browsers and devices. Tags such as <header>, <footer>, <aside> and <nav> were used to ensure accessibility for benefitting devices such as screen readers.

CSS3 - The latest and standardised version of the CSS specification. CSS3 was made use of extensively in the development of the product. As the product was developed using the "mobile first" methodology, media queries, part of the CSS3 specification, were an essential part of the project, enabling the site to scale up to adapt to the size

of the screen. As well as media queries, features such as box-models, transitions, animations, advanced selectors and gradients, to name a few, were used in the project. For example, a multi-coloured border was used on the header of the website. This was created using the CSS3 gradient spec. Using a gradient for this aspect held advantages over using an image such as faster load times and consistency between different devices and browsers.

Sass - Sass is a professional grade CSS extension language that enhances the features of CSS and its capabilities. An object orientated (OOCSS) approach was used in the development of the CSS for the project to encourage code reuse for a faster, more efficient stylesheet that is easier to maintain and scale. Sass helped in this regard as it allowed for the stylesheet to be broken up into numerous 'pages' for separating out styling rules.

PHP - PHP has been used within the project for handling all server side tasks such as POSTing a new user's details to the database for registration, checking login details with those stored in a database, POSTing moodmemos to the database, gathering data for use with statistics, fetching the moodmemo timeline etc.

Fetching the timeline

This was done by creating a query that would fetch the timeline from the database, then creating a variable for the timeline, and running the formatMoodmemo function in this variable which would then retrieve the moodmemos and apply them to the timeline. In order to display the latest moodmemo at the top of the timeline the moodmemos are listed descending by their ID.

Formatting a moodmemo

FIGURE 4.7 - FORMAT MOODMEMO FUNCTION

```
function formatMoodmemo($moodmemo,$dt,$mood,$id)
{
    global $favid, $matches;

    if(is_string($dt)) $dt=strtotime($dt); //if time is mysql data string then covert to timestamp
    $decrypted_moodmemo=mc_decrypt($moodmemo, ENCRYPTION_KEY); //decrypt moodmemo
    $moodmemo=htmlspecialchars(stripslashes($moodmemo)); //prevent vulnerabilities using htmlspecialchars
    //return a formatted moodmemo. Convert any links to a hyperlink
    return'
<li class="timeline-block" id="'.$id.'">
  <div class="timeline-img main-timeline-img">
    
  </div>

  <div class="timeline-content content_box"> <p>'. preg_replace('/((?:http|https|ftp):\\\/\\\/(?:[A-Z0-9-]
  A-Z0-9-]*(?:\.\.[A-Z0-9-]A-Z0-9-]*)+):?(\\d+)?\\\/?[*\s\''\']+/i', '<a href="$1" rel="nofollow"
  target="blank">$1</a>',$decrypted_moodmemo).'</p>
  <div class="row item-info">
    <div class="one-half column">
      <time class="ss-pika">'.relativeTime($dt).'</time>
    </div><!-- end column -->
    <div class="one-half column">
      <ul class="u-pull-right">
        <a title="More" class="show-menu tooltip-top-delay" href="javascript:void(0)"></a>

        <div class="item-functions u-pull-right">
          '.(($matches == 0) ? '<a href="javascript:void(0)" id="'.$id.'" title="Add to favourites"
          class="ss-icon ajaxFavourite tooltip-bottom">Favorite</a>' : '<a href="javascript:void(0)"
          id="'.$id.'" title="Remove from favourites" class="ss-icon ajaxFavourite tooltip-
          bottom">Favorite</a>').'
          <a href="javascript:void(0)" id="'.$id.'" title="Delete moodmemo" class="ss-icon
          ajaxDelete tooltip-bottom">Trash</a>
        </div><!-- end item-functions -->
      </ul>
    </div><!-- end column -->
  </div><!-- end row -->
</div><!-- end timeline-content -->
</li>';
}
```

In order to format the moodmemo into HTML a function was used. This function was created following a tutorial entitled “Making our own Twitter Timeline”. [Tutorialzine, (2009)] Firstly, in order to display the time that the Moodmemo was created we must convert it to a timestamp if the time is a mysql data string. Then, in order to

prevent vulnerabilities and convert predefined characters such as “<”, htmlspecialchars is used and strip slashes is used to strip any slashes.

The Moodmemo is then returned, links are converted using a preg_replace snippet for converting text into links. The Moodmemo is formatted, with all of the necessary information being imported from the database; the ID (\$id), the mood type (\$mood), the Moodmemo text (\$moodmemo) and the datetime (\$dt).

Calculating the relative time

In order to calculate the time that has passed since a Moodmemo was created a function was used that calculates this and displays it in a more user friendly way such as 'less than a minute ago' rather than a time such as '3 hours, 15 minutes and 48

seconds ago'. This function was adapted from the Fuzzy Time method used by the CodeIgniter framework. [Gowri sankar.R,]

FIGURE. 4.8 - RELATIVE TIME FUNCTION

```
1243 //finding out the relative period of time that has passed since the given time
1244 function relativeTime($dt,$precision=2)
1245 {
1246     $times=array( 365*24*60*60 => "year", //setting up times arrays
1247                  30*24*60*60  => "month",
1248                  7*24*60*60   => "week",
1249                  24*60*60     => "day",
1250                  60*60        => "hour",
1251                  60           => "minute",
1252                  1            => "second");
1253
1254     $passed=time()-$dt; //time passed = time - the datetime from the moodmemo
1255
1256     //if the moodmemo was created less than five seconds ago then display as such
1257     //else run a loop to calculate time passed
1258     if($passed<5)
1259     {
1260         $output='less than 5 seconds ago';
1261     }
1262     else
1263     {
1264         $output=array();
1265         $exit=0;
1266
1267         foreach($times as $period=>$name)
1268         {
1269             if($exit>=$precision || ($exit>0 && $period<60)) break;
1270
1271             $result = floor($passed/$period);
1272             if($result>0)
1273             {
1274                 $output[]=$result.' '.$name.($result==1?'':'s');
1275                 $passed-=$result*$period;
1276                 $exit++;
1277             }
1278             else if($exit>0) $exit++;
1279         }
1280
1281         $output=implode(' and ', $output).' ago';
1282     }
1283     return $output;
1284 }
```

The function involves setting up arrays for calculating a month, week, day etc then working out the time passed and storing it in a variable by taking away the time from the datetime of the posted Moodmemo. An if statement is used to determine if the Moodmemo was posted less than 5 seconds ago to display "less than 5 seconds ago" else run a loop to calculate the relative time passed.

jQuery - the most popular Javascript library around, jQuery simplifies the javascript library and is designed to make it easier to select DOM elements, create animations, develop AJAX applications etc. A lot of plug-ins will also require the jQuery library to function. JQuery is used within moodmemo for various features such as client side validation, creating an AJAX call for the Moodmemos, deleting Moodmemos, favouriting Moodmemos etc.

A javascript function was used to calculate the amount of characters in the Moodmemo textarea and to limit this to 220 characters. This was revised from the same tutorial used to create the timeline functionality. [Tutorialzine, (2009)] When the

counter gets to zero then the submit button is disabled and a class is executed to display this. The class, which changes the counter to red, is also added when there are 10 characters remaining in the textarea limit.

FIGURE. 4.9 - RECOUNT FUNCTION

```
function recount()
{
  var maxlen=220; //set max length
  var current = maxlen-$('#inputField').val().length; //set current length
  $('#counter').html(current);

  if(current<0 || current==maxlen) //if current is equal to 0 or max length then disable submit
  {
    $('#counter').addClass('counter--red');
    $('#submitButton').addClass('button-inactive').removeClass('button-primary');
  }
  else //else enabled button
    $('#submitButton').removeAttr('disabled').removeClass('button-inactive').addClass('button-primary');

  if(current<10) // if current is less than 10 change colour of counter
    $('#counter').addClass('counter--red');
  else //else leave at default colour
    $('#counter').removeClass('counter--red');
}
```

UserCake - Originally Moodmemo contained a login and registration system which allowed for registration, logging in, password changing and password recovery. The features and code for this system was outlined in a video walkthrough of the functional prototype. The passwords were encrypted using the md5 hashing algorithm and database connections were made using mysql before being replaced with the more secure mysqli. Research undertaken into cryptography brought to light that md5 is no longer secure for protecting passwords. *[Whittaker, Z. (2012)]*

Considering the sensitive nature surrounding Moodmemo it is implacable that users are confident that their data is secure. Therefore it was decided that a user registration library with much better security features would be better than the current insecure solution. The creator felt that he did not know enough about password hashing and storing data securely to build a system user's could be confident in. After some research into the subject, UserCake *[Cassels, J. (n.d.)]* seemed

like the most favourable solution. UserCake is a secure, fully open source user management script that offers all of the features of the previous solution and more. Most importantly, it has robust security features. UserCake uses a 25 to 32 character SALT along with a SHA1 hash to encrypt passwords. This may not be entirely secure, as no system truly is, but SHA1 is considered much more secure than md5 and no collisions on the hash have ever been produced. It is recommended that any website that stores more than 50,000 passwords should be using its own uniquely designed algorithm but for the scope of this project, UserCake will be a viable solution.

4.7.5 - APIs

Google Maps API - The Google Maps and Google Geolocation API were used for firstly, asking a user for permission to share their location and secondly, displaying their location on a map when a moodmemo is created.

FIGURE. 4.10 - GEOLOCATION RETRIEVAL

```
$('#createMoodmemo').click(function()
{
    $('#lat').val(0);
    $('#lng').val(0);
    if (navigator.geolocation) {
        navigator.geolocation.watchPosition(
            function (position) {
                $('#lat').val(position.coords.latitude);
                $('#lng').val(position.coords.longitude);
            },
            function () {
                $('#lat').val(0);
                $('#lng').val(0);
            }
        );
    }
    else {
        $('#lat').val(0);
        $('#lng').val(0);
    }
});
```

When a user clicks on the button to create a Moodmemo. The hidden fields #lat and #lng that are responsible for posting the latitude and longitude are set to a value of 0. If the user has enabled location sharing, then the #lat and #lng are replaced with the user's co-ordinates. If the user does not allow

location sharing then #lat and #lng are once again given the value of 0. This is to ensure all use cases are covered; enabled location sharing, disabled location sharing,

unsupported location sharing and ignored location sharing (neither enabled or disabled).

FIGURE. 4.11 - MAP MARKER POPULATION

```
<?php
    $fetchmarkers = mysqli_query($mysqli, "SELECT * FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' AND
    lat!='0.000000' AND lng != '0.000000'");//select all and exclude rows where the user has not allowed
    location tracking
    while ($row = mysqli_fetch_array($fetchmarkers)){
        $lat=$row['lat'];
        $lng=$row['lng'];
        $dt=$row['dt'];
        $id=$row['id'];
        $moodmemo=mc_decrypt($row['moodmemo'], ENCRYPTION_KEY);
        $mood=$row['mood'];
        echo ("addMarker($lat, $lng, '<b>$moodmemo</b><br>$dt', ( is_internetExplorer11 ) ? '$mood-marker.png' : '$
        mood-marker.svg');\n");
    }
?>
center = bounds.getCenter();
map.fitBounds(bounds);
```

In regards to retrieving Moodmemos and populating these on the Google Map, this is done using PHP to extract the needed information from the database. Any records with a latitude and longitude equal to zero are excluded from being retrieved to ensure any Moodmemos with disabled/unavailable geolocation co-ordinates are not populated onto the map. A function 'is_internetExplorer11' is used to check the browser vendor and if the user is using Internet Explorer then switch from using svgs to using pngs for the marker icon. The map is also centred between the two farthest marker points, using the getCenter() function, to ensure that all markers are present on the screen when the map is accessed.

Highcharts API - Highcharts is a robust and extensive charting API that was utilised in Moodmemo in the creation of charts and statistics to be presented to the user. This API was chosen over others because of its great API documentation, the creator's previous familiarity with the API and its robust, well supported and powerful library for designing and building charts.

5.0 - Implementation

5.1 - Tool Selection

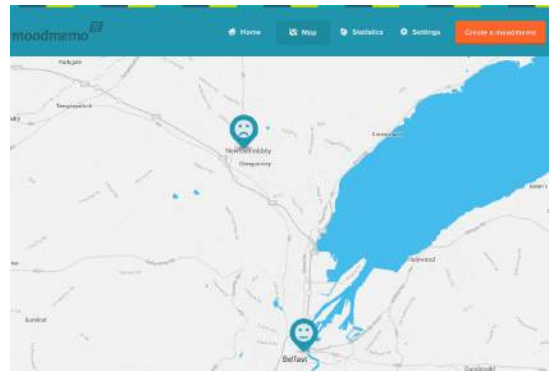
The technology and tools that were used in the project were briefly outlined within section 4.0 of the report where they were researched for their viability in use with the project. PHP was chosen as the language that would communicate with the backend and databases that would enable the use of the project. Javascript and jQuery were chosen to initiate the timeline and post information to the database via Ajax. The Google Maps API was utilised for its geolocation functionality and map features and various JS plugins were implemented to enhance the functionality and experience of using the product.

5.1.1 - Google Maps API

The Google Maps API was implemented within the map.php section of the application. This displays an interactive Google Map where a user can view their created Moodmemos and where they were created.

FIGURE 5.0 & 5.1 - GOOGLE MAPS CODE & VISUAL IMPLEMENTATION

```
google.maps.event.addDomListener(window, "load", function() {
    var is_internetExplorer11= navigator.userAgent.toLowerCase().indexOf('trident') > -1; //calculate
    if browser is IE
    var bounds = new google.maps.LatLngBounds(),currentPopup,
    iconBase = 'https://moodmemo.co/images/',
    map = new google.maps.Map(document.getElementById("google_canvas"), {
        center: new google.maps.LatLng(0, 0),
        zoom: 14,
        mapTypeId: google.maps.MapTypeId.ROADMAP,
        mapTypeControl: false,
        styles: style,
        mapTypeControlOptions: {
            style: google.maps.MapTypeControlStyle.HORIZONTAL_BAR
        },
        navigationControl: true,
        navigationControlOptions: {
            style: google.maps.NavigationControlStyle.SMALL
        }
    });
    addMarker = function(lat, lng, info, mood) {
        var pt = new google.maps.LatLng(lat, lng);
        bounds.extend(pt);
        var marker = new google.maps.Marker({
            position: pt,
            icon: iconBase + mood, //add marker icon based on mood
            map: map,
            zIndex: 1,
            animation: google.maps.Animation.DROP
        });
        var popup = new google.maps.InfoWindow({
            content: info,
            maxWidth: 300
        });
        google.maps.event.addListener(marker, "click", function() {
            if (currentPopup != null) {
                currentPopup.close();
                currentPopup = null;
            }
            popup.open(map, marker);
            currentPopup = popup;
        });
        google.maps.event.addListener(popup, "closeclick", function() {
            currentPopup = null;
        });
    };
});
```



5.1.2 - Skeleton Framework

A framework was used to create the initial foundations of the project. The skeleton framework is a small, lightweight, responsive framework with a small amount of standard HTML elements as a starting point. It includes a 12 column responsive grid. The skeleton framework is noted for its simplicity and easy to utilise and understand grid system. This was used as a starting point for the development of the Moodmemo UI.

5.1.3 - Highcharts API

Highcharts was used for the statistics present on the Statistics page within the application and for an overview chart of created Moodmemos in the Timeline page. In order to create a chart that displays the amount of each type of Moodmemo a user has created based on the selected mood, the number of each mood had to be retrieved from the database. This was done using queries to get the total number of each mood.

FIGURE 5.2 - RETRIEVING TOTAL COUNT FOR MOODS

```
// get moodmemo numbers data
$totalcount = mysqli_query($mysqli, "SELECT * FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id'");
$terriblecount = mysqli_query($mysqli, "SELECT mood FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' && mood='terrible'");
$sadcount = mysqli_query($mysqli, "SELECT mood FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' && mood='sad'");
$okcount = mysqli_query($mysqli, "SELECT mood FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' && mood='ok'");
$happycount = mysqli_query($mysqli, "SELECT mood FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' && mood='happy'");
$greatcount = mysqli_query($mysqli, "SELECT mood FROM tbl_moodmemo WHERE user_id='$loggedInUser->user_id' && mood='great'");
$num_total = mysqli_num_rows($totalcount);
$num_terrible = mysqli_num_rows($terriblecount);
$num_sad = mysqli_num_rows($sadcount);
$num_ok = mysqli_num_rows($okcount);
$num_happy = mysqli_num_rows($happycount);
$num_great = mysqli_num_rows($greatcount);

$values= array($num_terrible, $num_sad, $num_ok, $num_happy, $num_great);
$moodValues = implode(",", $values);
```

The totals are placed into an array that is then assigned to a variable.

FIGURE 5.3 & 5.4 - DATA OPTIONS AND VISUAL OF OUTPUTTED CHART

```
series: [{
  name: 'Mood',
  data: [<?php echo $moodValues ?>]
}]
```



The variable is called within the data options, that loops through the values and outputs them into bars within a bar chart, as seen in **Figure 5.4**.

The highcharts API was also used to create a line graph that plots a user's Moodmemos on a timeline within the chart. This allows the user to visualise how their mood is changing over time.

FIGURE 5.5 - RETRIEVING THE DATA

```
//run query to select dt and mood
$query = mysqli_query($mysqli, "SELECT dt, mood from tbl_moodmemo WHERE user_id='$loggedInUser->user_id' group by dt ORDER BY dt");

//testing
if (!$query) {
  echo var_dump($mysqli);
  die();
}

//loop through and get the data
$row = $query->fetch_assoc();
$json_mood = array();
$json_date = array();

do{
  $mood[] = $row['mood'];
  array_push($json_mood, $row['mood']);
  array_push($json_date, $row['dt']);
} while($row = $query->fetch_assoc());
```

In order to populate the chart with the data, it first needs to be retrieved from the database. A query is used to select the dt(datetime), and mood from the Moodmemo table based on the current user. The data is then looped through to retrieve and sort the Mood and dt.

FIGURE 5.6 - CONVERTING DATA

```
var moodList = ["terrible","sad","ok","happy","great"]; //create array for mood list
var moodDict = {"terrible":0, "sad":1, "ok":2, "happy":3, "great":4}; //assign integer to each mood item
```

In order to make the Moods readable by the chart, it was necessary to assign an integer to each Mood which could then be used to plot the points on the chart as seen in **Figure 5.6**.

FIGURE 5.7 - OUTPUTTING THE DATA

```
series: [{
  showInLegend: false,
  data: (function() {
    var moods = <?php echo json_encode($mood);?>;
    var dates = <?php echo json_encode($json_date,$row);?>

    var data = [];
    var moodImg;

    for (var i = 0; i < moods.length; i++) {

      if (moodDict[moods[i]] == 0) {
        moodImg = 'url(..images/terrible.png)';
      } else if (moodDict[moods[i]] == 1) {
        moodImg = 'url(..images/sad.png)';
      } else if (moodDict[moods[i]] == 2) {
        moodImg = 'url(..images/ok.png)';
      } else if (moodDict[moods[i]] == 3) {
        moodImg = 'url(..images/happy.png)';
      } else if (moodDict[moods[i]] == 4) {
        moodImg = 'url(..images/great.png)';
      }

      data.push({
        x: new Date(dates[i]),
        y: moodDict[moods[i]],
        marker: {
          symbol: moodImg,
          width: 20,
          height: 20
        }
      });
    }
  })()
}]

return data;
}()
```

Variables are then declared that encode the data into json. A 'for loop' is used to find the value of each Mood and assign a symbol to it based on the Mood that value represents as seen in **Figure 5.7**. The visual result of this can be seen in **Figure 5.8**.

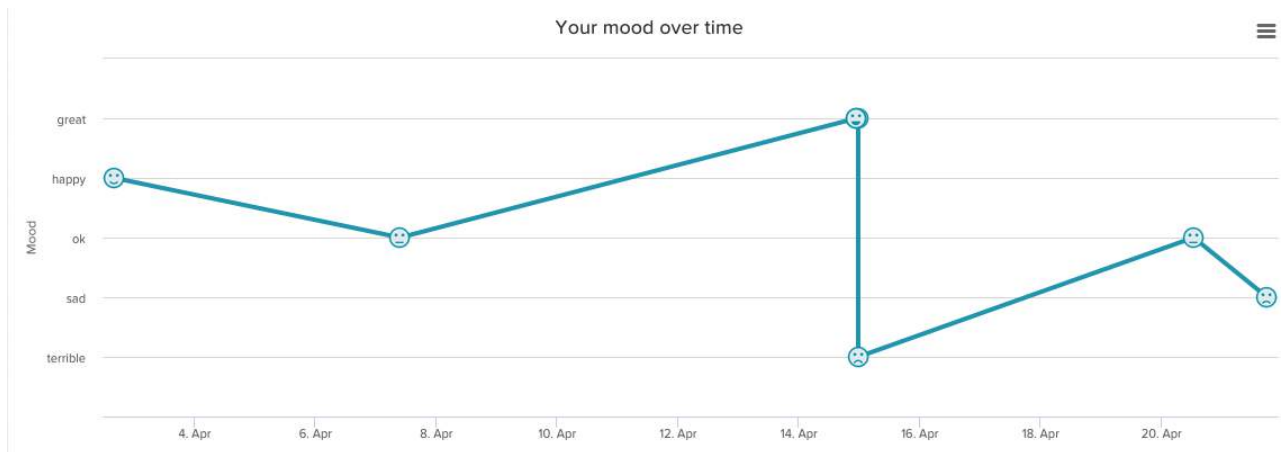


FIGURE 5.8 - VISUALISING THE DATA

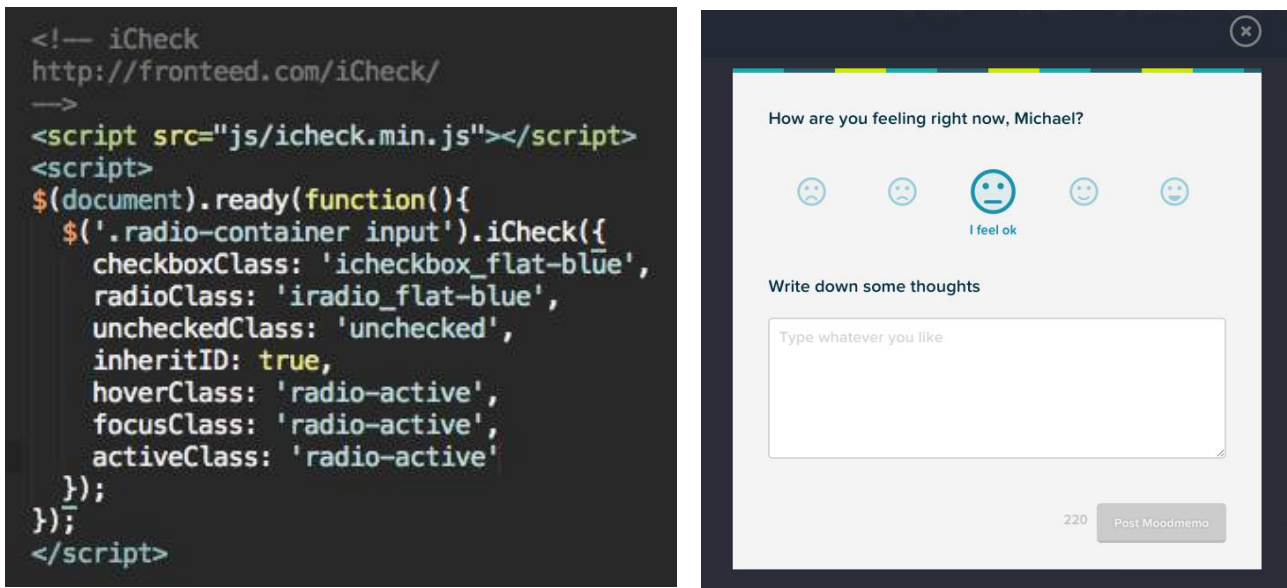
5.2 - Technologies & Tools Used

Once the tools had been selected and utilised the next stage involved implementation of the technologies and tools which were selected. This section also details some of the solutions to implementations and features present within the product.

5.2.1 - jQuery radio buttons

One of the key aspects of the application is the ability to choose a mood when creating a Moodmemo. This is done within the product using radio buttons. However, styling HTML radio buttons using CSS is limited. In order to create a mood selection process that felt more intuitive and attractive, the jQuery plugin, `icheck.js` [Fronteed.com, (n.d.)] was used to provide more functionality to styling radio buttons and labels. **Figure 5.9** shows the `icheck` code used to style the radio inputs. The library adds classes to each stated radio button allowing for customisation of each input. **Figure 5.10** shows the result of the library being utilised to style the radio buttons.

FIGURE 5.9 & 5.10 - ICHECK.JS CODE AND VISUAL IMPLEMENTATION

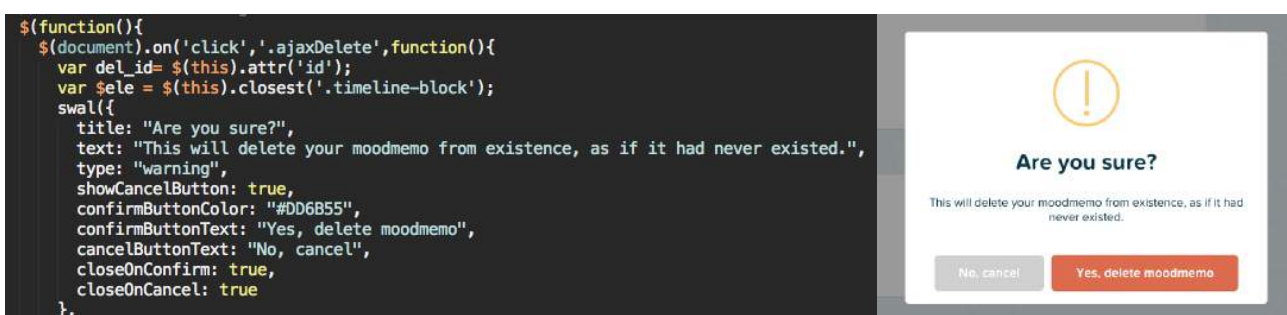


5.2.2 - Sweet Alert

In order to replace the standard javascript alert function with a method that is more customisable and intuitive, a javascript plugin called Sweet Alert was used [T4t5.github.io, (n.d.)]. This plugin simply replaces the default alert and provides more functionality for customising error messages and providing feedback when any action is carried out.

For instance, the plugin was utilised in Moodmemo to trigger a confirmation box when a user attempts to delete a Moodmemo, asking them to confirm their action.

FIGURE 5.11 & 5.12 - CONFIRMATION DIALOG CODE AND VISUAL IMPLEMENTATION



The plugin was also used in conjunction with cookies to display a message when a user first visits the Map and Statistics page, to provide them with information on the page and why it may be currently blank. Cookies are used to display the modal only on the first visit to the page.

FIGURE 5.13 - COOKIES FUNCTION

```
function GetCookie(name) {
    var arg=name+"=";
    var alen=arg.length;
    var clen=document.cookie.length;
    var i=0;

    while (i<clen) {
        var j=i+alen;
        if (document.cookie.substring(i,j)==arg)
            return "here";
        i=document.cookie.indexOf(" "+i)+1;
        if (i==0)
            break;
    }

    return null;
}

$(function() {
    var visit=GetCookie("COOKIE1");

    if (visit==null){
        swal({
            title: "<div class='empty-state-modal title-container'><h4>This is your Moodmemo map</h4></div>",
            text: "<div class='empty-state-modal text-container'><p>If you allow it, when you create a Moodmemo it will be pinpointed on this map when and where it was created. Pretty cool, right?!</p></div>",
            imageUrl: "../images/map-empty-state.png",
            html: true,
            confirmButtonColor: "#ff6428",
            confirmButtonText: "Okay, got it!",
            imageSize: "130x138"
        });
        var expire=new Date();
        expire=new Date(expire.getTime()+7776000000);
        document.cookie="COOKIE1=here; expires="+expire;
    }
});
```

Function getCookie is used to create the cookie. A function then checks on page load to see if the cookie exists and if not then triggers the modal to display.

5.2.3 - Favouriting a Moodmemo with AJAX

In order to provide the user with the option of storing certain Moodmemos in an area that would make them easier to locate, favouriting functionality was implemented for each Moodmemo, so each one could be favourited and unfavourited.

FIGURE 5.14 - JS CODE FOR FAVOURITING A MOODMEMO

```
//function for favouriting a Moodmemo
$(function(){
  $(document).on('click','ajaxFavourite',function(){

    var favid= $(this).attr('id');
    s(this).toggleClass('favourited');

    $.ajax({
      type: 'POST',
      url: 'addremove.php',
      data: {'favid':favid},
      success: function(data){
      },
      error: function(data){
        swal("Oops...", "Something went wrong there :(", "error");
      }
    });
  });
});
```

Figure 5.14 shows the javascript code responsible for favouriting a Moodmemo.

When a user clicks the favourite button present within a Moodmemo, the ID of the Moodmemo is stored in a variable by retrieving the ID of

the button. The variable is then POSTed to a php file.

FIGURE 5.15 - ADDREMOVE.PHP

Figure 5.15 shows the addremove.php file. The variable that was posted with AJAX is then stored within a PHP variable. A query is then ran to see if a favourite exists based on the current user and the ID of the clicked favourite button. If there are no matches then the record is sent to the table, if there is a match, then the record is deleted from the table.

```
$favid=$_REQUEST['favid']; //get this from ajax

// Firstly, check if moodmemo is favourite or not
$query = mysql_query($mysql, "SELECT * FROM tbl_favourites WHERE user_id='loggedInUser->user_id' AND moodmemo_id=$favid");
$matches = mysql_num_rows($query);

// If it is not favourited, add as favourite
if($matches == '0'){
  mysql_query($mysql, "INSERT INTO tbl_favourites (user_id, moodmemo_id, is_favourite) VALUES ('loggedInUser->user_id', '$favid', 1)");
  $_SESSION['btnClicked'] = 'success';
}

// Instead, if it is favourited, then remove from favourites
if($matches != '0'){
  mysql_query($mysql, "DELETE FROM tbl_favourites WHERE user_id='loggedInUser->user_id' AND moodmemo_id=$favid");
  unset($_SESSION['btnClicked']);
}
```

5.2.4 - View password on click/tap

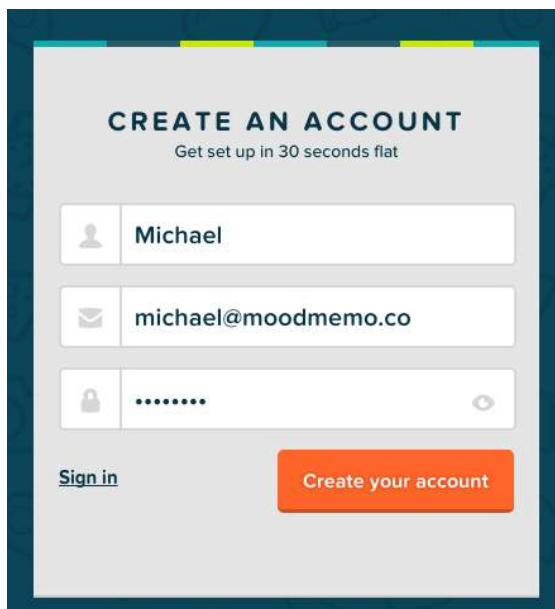
A usability feature that is often absent from many interfaces is the option to view the information you've typed into a password field when logging in or registering. This is particularly useful on touch screen devices such as phones where mistypes are much more common. Considering that there is no confirm password field when registering

for an account with Moodmemo, it was necessary to provide some functionality to the user that allows them to view their typed password before proceeding.

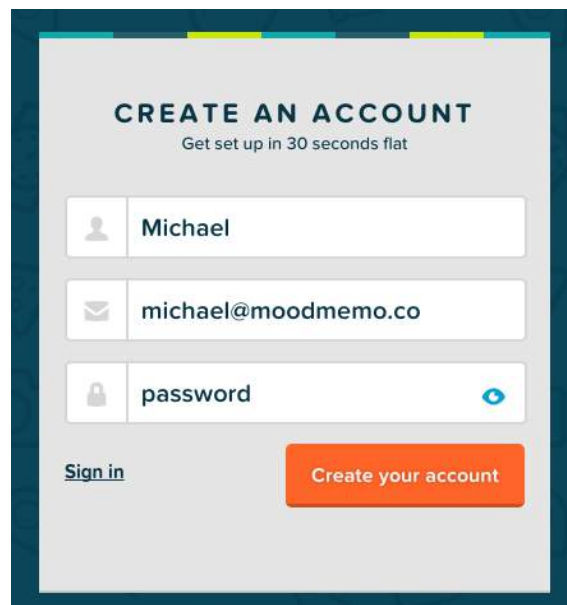
FIGURE 5.16 - SHOW PASSWORD ON TOGGLE WITH JAVASCRIPT

```
//show password on toggle
$(function () {
  $(".show-password").each(function (index, input) {
    var $input = $(input);
    $(".reveal-password").click(function () {
      var change = "";
      if ($(this).prop('title', 'Show password')) {
        $(this).prop('title', 'Hide password')
        $(this).addClass("reveal-password-active")
        change = "text";
      } else {
        $(this).prop('title', 'Show password')
        $(this).removeClass("reveal-password-active")
        change = "password";
      }
      var rep = $("<input type='" + change + "' />")
        .attr("id", $input.attr("id"))
        .attr("name", $input.attr("name"))
        .attr('class', $input.attr('class'))
        .attr('placeholder', $input.attr('placeholder'))
        .attr('tabindex', $input.attr('tabindex'))
        .val($input.val())
        .insertBefore($input);
      $input.remove();
      $input = rep;
    }).insertAfter($input);
  });
});
```

The code functions by changing the input type from 'password' to 'text' when '.reveal-password' is clicked, thus showing the contents of the field. When clicked again the input is changed back to a password type. The title attributes are also changed on click.



The screenshot shows a 'CREATE AN ACCOUNT' form with the subtitle 'Get set up in 30 seconds flat'. It contains three input fields: a name field with 'Michael', an email field with 'michael@moodmemo.co', and a password field with masked characters '.....'. To the right of the password field is an eye icon. At the bottom left is a 'Sign in' link, and at the bottom right is an orange 'Create your account' button.



The screenshot shows the same 'CREATE AN ACCOUNT' form, but the password field now displays the text 'password' instead of masked characters. The eye icon to the right of the field is now blue, indicating it is active. The 'Sign in' link and 'Create your account' button remain at the bottom.

FIGURE 5.17 & FIGURE 5.18 - HIDDEN PASSWORD AND REVEALED PASSWORD

5.2.5 - User Profile Photo

In order to give the user a greater sense of ownership and customisation over their account the functionality to upload their own profile photo was implemented.

FIGURE 5.19 - PHP CODE FOR UPLOADING A USER PHOTO

```
$path = '/home/moodmemo/public_html/user-photos/'.$uid.'/';
$urlpath = 'user-photos/'.$uid.'/';

if(is_dir($path))
{
    $uploadpath = $path;
}
else
{
    mkdir($path);
    $uploadpath = $path;
}

$max_size = 400; // maximum file size, in KiloBytes
$alwidth = 1000; // maximum allowed width, in pixels
$alheight = 1000; // maximum allowed height, in pixels
$allowtype = array('bmp', 'gif', 'jpg', 'jpeg', 'png'); // allowed extensions
$extension = pathinfo($_FILES['fileup']['name'], PATHINFO_EXTENSION);
$extension = strtolower($extension);
while (true) {
    $filename = uniqid(rand(), true) . '.' . $extension;
    if (!file_exists($uploadpath . $filename)) break;
}

if(isset($_FILES['fileup']) && strlen($filename) > 1) {
    $uploadpath = $uploadpath . basename($filename); // gets the file name
    list($width, $height) = getimagesize($_FILES['fileup']['tmp_name']); // gets image width and height
    $err = ''; // to store the errors
```

Figure 5.19 shows the PHP code that was used to develop the user profile photo functionality. First a path is created on the server with a user-photos folder to store the user's profile

image. The name of the folder is based on the ID of the user. Variables are then set for the dimensions and filetype of the image in order to handle validation checks. An 'if statement' is used to check the uploaded image meets the requirements outlined in the variables. If it does not, then an error is thrown and the image is not uploaded, if the image passes the validation checks, a random number is generated and assigned to the image. it is then placed into the 'user-photos' folder and the filename is sent to the database.

FIGURE 5.20 - THE IMAGE UPLOAD FORM

```
<form name='updateAccount' action='php $_SERVER['PHP_SELF']; ?' method='post' enctype='multipart/form-data'>
    <div class='profile-settings'>
        <div class='row'>
            <div class='profile-settings_photo'>
                <?php
                    $result = mysqli_query($mysqli, "SELECT photo FROM tbl_users WHERE id=$loggedInUser->user_id");
                    while($row=mysqli_fetch_assoc($result))
                    {
                        if (!empty($row['photo'])) {
                            echo "<img id='avatar-preview' src='https://".$_SERVER['HTTP_HOST'].rtrim(dirname($_SERVER['REQUEST_URI']), '\\').'/'.$urlpath . $row['photo']
                                . "' alt='your profile photo'>";
                        }
                        else {
                            echo "<img id='avatar-preview' src='images/profile-placeholder.svg' alt='You can change this'>";
                        }
                    }
                <?php
            </div>
        </div>
    </div>
```

Figure 5.20 shows the image upload and image preview form. An If statement is utilised to check if the photo row in the database is empty. If it's empty then the profile-placeholder image is set for the image preview. Otherwise, if the user has set an image then the uploaded image is displayed by retrieving the path.

In order to show a preview of the image that a user has selected but not yet uploaded a javascript function is used to append the image to the src attribute of the preview-image.

FIGURE 5.21 & 5.22 - IMAGE PREVIEW CODE AND VISUAL IMPLEMENTATION

```
//function for displaying a preview of profile avatar
function readURL(input) {
  if (input.files && input.files[0]) {
    var reader = new FileReader();

    reader.onload = function (e) {
      $('#avatar-preview').attr('src', e.target.result);
    }

    reader.readAsDataURL(input.files[0]);
  }
}

$('#avatar-up').change(function(){
  readURL(this);
});
```



MICHAEL

Registered: Apr 02, 2015

Choose file

JPG, GIF, PNG OR BMP no larger than 400Kb.

5.2.6 - Encrypting & decrypting a Moodmemo

In order to protect the privacy of the user it was essential that their Moodmemos were encrypted before being sent to the database to ensure they are unreadable to any individual that has access to the database records. A function was used to encrypt the text before it is sent to the database and then decrypted again when it was required to be read in plain text within the application. This was completed using the PHP mcrypt function.

FIGURE 5.23 - MCRYPT FUNCTION

```
/*
Encrypting function
refined from this guide http://www.warpconduit.net/2013/04/14/highly-secure-data-encryption-decryption-made-easy-with-
-php-mcrypt-rijndael-256-and-cbc/
*/
function mc_encrypt($encrypt, $key){
    $encrypt = serialize($encrypt);
    $iv = mcrypt_create_iv(mcrypt_get_iv_size(MCRYPT_RIJNDAEL_256, MCRYPT_MODE_CBC), MCRYPT_DEV_URANDOM);
    $key = pack('H*', $key);
    // $key = pack('H*', str_replace(' ', '', $key[0]));
    $mac = hash_hmac('sha256', $encrypt, substr(bin2hex($key), -32));
    $passcrypt = mcrypt_encrypt(MCRYPT_RIJNDAEL_256, $key, $encrypt.$mac, MCRYPT_MODE_CBC, $iv);
    $encoded = base64_encode($passcrypt).'|'.base64_encode($iv);
    return $encoded;
}
```

The data is encrypted using the sha256 hash and a key which is stored securely within the application.

To encrypt or decrypt information the function, mc_encrypt or mc_decrypt must be called along with the generated key. Information can't be decrypted again without access to the key.

FIGURE 5.24 - ENCRYPTING A MOODMEMO

```
$encrypted_moodmemo = mc_encrypt($moodmemo, ENCRYPTION_KEY);
```

5.3 - Notable Challenges

There were many notable challenges that were faced when creating the product. The project had a stimulating balance of visual and technical challenges that needed to be addressed if the project is not only to be considered a success, but also to create a product that evokes a great user experience and elicits a strong sense of professionalism.

The very first challenge met was creating a backend system using PHP of which to build Moodmemo upon. Storing information was uncommon ground for the creator and a lot of knowledge had to be relearned and obtained in order to develop

communication with a database for securely storing information. Thankfully, this challenge was met and overcome and the end result was a backend PHP system that carries out all of the tasks and requirements that were originally outlined.

Retrieving the user's location, posting this to the database and then creating markers within Google Maps based on the user's location proved to be a challenge that was difficult to overcome. One of the issues faced with this feature was when a user had disabled or not allowed location sharing then Moodmemo would fail to post to the database. This was eventually overcome using a solution outlined in **Section 4.7.5**.

A challenge that was met but never overcame was one that seemed simple in theory but proved to be a large frustration and no tried method solved the issue. The challenge was checking to see if a Moodmemo had been favourited or not by querying the favourites table in the database and if it had then altering the class and title attribute of the favourites button in each favourited Moodmemo to 'Remove from Favourites'. This way, a user is able to visually differentiate the Moodmemos that have been favourited and unfavourited. This proved to be a difficult task as solutions that by all accounts *should* work, were not working. It was decided that too much time was being spent on this task and considering that it was not a functional or non-functional requirement, it should then be left until completion of the project, when ample time is available to search for a true solution.

5.4 - Notable Achievements

Moodmemo proved to be a difficult project to undertake, design and build. There were a lot of technical challenges and unknown difficulties that evolved during the development of the product. However, this meant there were also a lot of achievements that were accomplished along the way.

The most notable achievement accomplished was the Google Maps feature of populating a map with the user's Moodmemos. This wasn't the most technical challenge however it was the most rewarding because it was originally thought that this would not be possible to achieve and therefore was not set as an initial functional requirement. Achieving this meant the functionality of the application was improved drastically, giving the users the ability to track their moods based on location and build a map of Moodmemos that they could utilise to discover patterns in how they feel at particular locations.

Highcharts was utilised for creating the statistics and charts present in the application. Previous experience with Highcharts was possessed by the creator however no experience was possessed for dynamically charting data. One feature that proved to be very technically challenging was incorporating a method of charting a user's moods and displaying these over time. The solution to this issue was outlined in **Section 5.1.3.**

6.0 - Testing

Testing is an essential aspect of any application for assessing the quality of the product and ensuring that any requirements, both functional and non-functional have been met. In this section, the testing approach will be investigated to explore the approaches that will be utilised in the testing procedure. Various test cases will then be carried out to ensure that the outlined requirements have been met.

6.1 - Testing Approach

After building the application to a functional standard the next phase was to test the different elements of the system. The main aim of this exercise is to locate and repair any issues, bugs or broken features through the use of different methods of testing. This is to ensure that the application works as expected for users of the application.

6.1.1 - White Box Testing

White Box Testing involves testing the internal structures of the application rather than testing the underlying functionality. This method was used to test the application against the defined requirements and ensure that each is met, the functionality is present and it behaves as expected. This method of testing requires the tester to have an in-depth knowledge of the underlying system and infrastructure in order to test for failure scenarios.

6.1.2 - Black Box Testing

Black Box Testing was the other procedure that was selected for testing the application. This procedure involves looking at the system as a 'black box' of sorts. This means having no knowledge of the system or application is encouraged in order to ensure thorough and unbiased testing procedures are adhered to. Black Box

Testing would be utilised to test the functionality of the application and that a third party would be used to test these options to ensure it was not the creator themselves that was carrying out the testing to rule out any possibilities of bias.

6.1.3 - User Testing

User testing is fundamental for any application. This method involves the application being tested by various users whom are instructed to carry out tasks. This is a simple method of testing the usability of an application, particularly if a large and diverse sample of users is used. This method will help determine patterns of nuance or particular causes of frustration within the product. It helps test the primary actions of the system to ensure they are obvious and accessible to the majority of users.

6.2 - Test Process

Now that three testing methods had been researched and outlined it was time to start the testing process.

6.2.1 - White Box Testing

White Box Testing was used first to test the underlying system of the application and ensure that it was functioning as it should. This test method was an ongoing procedure that was carried out throughout the development of the product and used again once the product had been finalised to ensure that the application was still functioning as it should. These tests are located within **Appendix Two**.

6.2.2 - Black Box Testing

This testing method focuses on the functionality of the website. It is tested by third party users who are not involved with the design or development of the application. Therefore test cases needed to be provided to the user in order to test the

functionality aspects of the application. These tests are located within **Appendix Three**.

6.2.3 - User Testing

User Testing was carried out on five users, two of which were recorded for further research and investigating. In order to receive unbiased and un-manipulated tests, users were asked to perform certain tasks within the application but were given minimal instructions on how to do so. This was to ensure that each user could locate and carry out these tasks without intervention. Users with varying technical abilities and within different age groups were used to receive varied results. The two recorded user testing sessions reflect a user with an above average technical skill and understanding of the internet, and a user with very little technical skill and understanding of the internet, to the point where even their knowledge of using a keyboard was severely limited. Links to the recorded user testing sessions can be viewed below:

[User Test 1 on Vimeo \(password: moodmemo imd\)](#)

[User Test 2 on Vimeo \(password: moodmemo imd\)](#)

User test 1 & 2 display two contrasting users. Subject 1 was very familiar with the web and therefore had no issue registering for an account and needed very little prompting on how to carry out tasks on the website. They also showed inquisitiveness by creating more Moodmemos just to see the affect it displayed on different pages. Essentially “gamifying” the process. Subject 2, however, required a lot of prompting due to their unfamiliarity with using the internet. Typically, a user such as this would most likely never use a tool such as Moodmemo but it was important to see how they interacted with the system.

6.3 - Test Results

6.3.1 - Browser Testing

To ensure that the application conveyed a similar experience across the major browsers, testing needed to be carried out. Modern browsers are excellent at maintaining standards compliance so as long as vendor prefixes were being utilised were applicable, very little style differences would occur between the browsers. However IE9 is not up to date with current standards so further methods would need to be utilised to support this browser. For example, adding a fallback option for SVG images that are used throughout the product.

- **Chrome** - This was the primary browser used throughout the development of Moodmemo. Therefore the majority of testing had been carried out during development. This meant that Chrome would take the role as the standard that the other browsers had to adhere to.
- **Firefox** - One of the issues that appeared when testing the application within Firefox was that the typography was not consistent with how it appeared in Chrome and other browsers. This was an issue with how the browser renders text. The solution to this was one line of code that targeted the Mozilla Firefox browser on a Mac OSX system. Adding “-moz-osx-font-smoothing: grayscale;” to the html tag in the CSS fixed the issue.
- **Safari** - No visual issues became apparent when testing the system within Safari. The test concluded that the application was displaying as it appears in Google Chrome. However, a major problem did occur with one of the key interactions of the system, which also persisted in Firefox and other Non-Chrome browsers. The

issue was that within the statistics page, the line chart which displays a user's mood over time was not functioning. Thorough testing and research proved this issue to be with the browsers failing to parse the date formats retrieved from the database correctly, thus, disabling the chart. An external JS library, moment.js [*Momentjs.com, (n.d.)*], had to be utilised to ensure the date could be correctly parsed. Implementing this fixed the issue and the chart was now displaying.

- **Internet Explorer** - It was originally outlined that the application would be built to support versions of Internet Explorer 9 and above. Therefore, the application required testing in versions of Internet Explorer 9 and onwards. Thankfully, the developer tools in IE allow for different versions to be toggled for testing purposes. The modernizr.js library was utilised to create fallbacks for CSS functions that older versions of the browser do not support. For instance, IE9 and below doesn't support SVG images therefore modernizr.js and a javascript function were used to replace any SVG images with a PNG image fallback.

6.3.2 - User Testing

The user tests carried out, both recorded and unrecorded, demonstrated that no users had trouble signing up for an account or creating a Moodmemo, possibly due to the onboarding processes. Many users were quick to visit their Profile settings due to the call to action button located in the profile window in the timeline page. This is not necessarily a bad thing but perhaps too much emphasis has been placed on this secondary action and it distracts users from the primary action of creating and interacting with Moodmemos. Placing less emphasis on or removing the 'View Profile' button from the profile window could solve the issue.

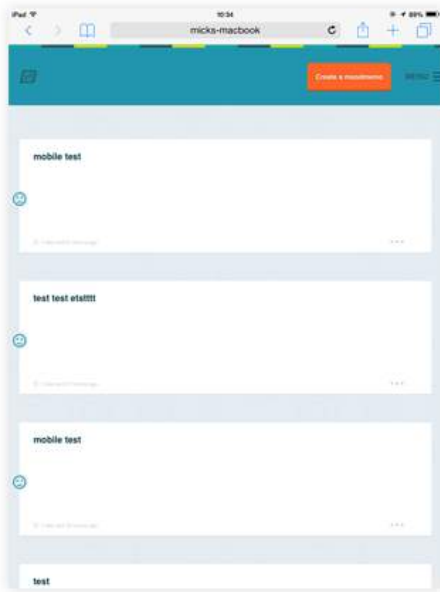
Another concern that arose during testing was that some users were very unaware of the intent of the application. Had they not been prompted beforehand by the test

coordinator, the user's would have no explanation as to how Moodmemo could benefit them or what the tool is for. The slogan on the register and login pages states "Keep track of your moods. Privately." This slogan serves its purpose but is perhaps too vague in demonstrating the key features of the application. Instead a demonstration page should be created to illustrate the features of Moodmemo to help onboard users and explain the process of using the application. Tumblr provides an excellent onboarding process where the application is explained simply and intuitively using impactful copy and interactive graphics. *[Tumblr.com, (n.d.)]*

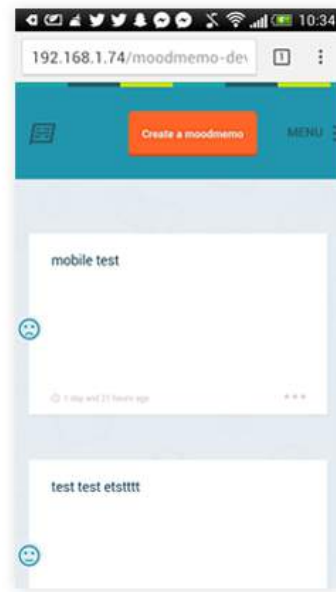
6.3.2 - Device & Responsive Testing

Along with testing the usability and functionality of the application it was also very important to test the application on multiple devices. Not only to ensure that the application is still functional and working correctly but also to ensure that the application is responding correctly to the size of the device i.e. it is responsive. **Figure 6.0** demonstrates some of the mobile devices that the application was tested on. Various mobile browsers were also utilised during this testing process to discover and fix any browser inconsistencies. The user profile box and main navigation is moved at mobile mobile sizes into a slide out menu activated by the 'Menu' link located in the header. The 'Create Moodmemo' button is always present at any width. A conscious decision was made to have the 'Create Moodmemo' button always visible from every page and not placed into a hidden menu when on a mobile device. This is because studies have shown that user engagement can decrease by up to 50% when primary user actions are placed into a slide-out menu. *[Rose, A. (2014)]* The Moodmemo logo was also replaced with just the logomark in order to provide more room within the header.

FIGURE 6.0 - DEVICE TESTING



iPad



HTC One S



iPhone 5s

7.0 - Evaluations

This section uses the final product, aims and outlined requirements to determine its success and to evaluate the progression of the project from the start to the finalised solution.

7.1 - Project Outcomes Evaluation

In order to evaluate the success of the project it was important to compare this with the originally created requirements and aims.

The aim of Moodmemo, that had not changed since the conception and development of the idea was a web app *“that will allow users to keep track of their moods by providing them with the means to create posts, known as ‘moodmemos’, that will be displayed within a timeline to which only they have access to.”*

Testing showed that all refined functional and non-functional requirements that had been outlined, along with the outlined objectives, were met and were functioning as specified. Elements of the application that were discovered during testing to not be functioning correctly were addressed and fixed. One initial non-functional requirement had not been met. This requirement was the addition of an FAQ section where users could get access to questions about the service that they may have. This requirement was not met due to the fact that frequently asked questions should be created from a user’s perspective and surveys on the user base would need to be carried out in order to create questions that address a user’s questions regarding Moodmemo.

Usability testing results showed that users did not have any difficulties creating an account or submitting a Moodmemo. Studying the recorded sessions of user testing showed that a user’s primary action was to first create a Moodmemo by clicking the

“Lets get started” button displayed during the user’s first visit to the web application timeline. User’s were also recorded interacting with the Statistics and Map pages of the application. One user even purposefully created Moodmemos with different selected moods to see the effect this would display within these pages. This was key evidence that the primary action of creating Moodmemos and viewing them in a timeline was being successfully fulfilled by the users.

7.2 - Methodology Evaluation

The ‘Waterfall’ Methodology was chosen as the methodology to be utilised for this project after weighing the options through completed research. The rigidness of the Waterfall Methodology proved complimentary to the approach taken for developing the application and deliverables that were to be completed during this development cycle. Structuring the project into phases, as outlined in the Gantt Chart, proved to be an efficient method for ensuring deliverables were completed and that the application stayed on course for the stated deadlines while also staying within the parameters of the methodology.

7.3 - Plan Evaluation

The Waterfall Methodology Gantt Chart (**Figure 2.5**) contained a structured analysis of all of the different sections that required completion in order to successfully fulfil each task. During the development of the Gantt Chart, research was undertaken into each deliverable to be completed during the development of the project. A defined plan was then formulated to ensure that these dates coincided with the different development stages/milestones of the project. This led to an effective strategy being utilised and ensuring that each milestone was met and within the scope of the timeline. Potential contingencies were accounted for when developing the Gantt Chart to provide some leeway when certain tasks were taking more time than originally planned for.

8.0 - Conclusion

8.1 - Summary of Report

Moodmemo comprised of many stages that required accomplishment to ensure that the project could be deemed successful. The introduction of the report established the background of the project, the conception of the idea and reflection of undertaken research to ascertain competitors within the field. It was surmised that the idea was viable and there was potential for the conceived product. The aims and objectives of the product were then outlined before proceeding to the next stage of defining the concept. This involved a further overview of the idea generation process for Moodmemo and an outlining of the functional and non-functional requirements to be met. The next stage involved showcasing the evolution of the initial stages of the design for Moodmemo. This involved an outlining of the paper prototyping process which helped to determine the functionality and processes of the project. This was then followed by 6-Ups and wire framing to abroad the structure and layout of the application and looking at key interactions within the application in further detail. This then lead to the design stage of the report which involved an overview of the full design and user experience of the application such as brand identity, voice & tone, user onboarding, and an online styleguide.

Once the development of the product had been finalised it needed to be tested and documented. Different methods of testing were carried out to ensure that each of the functional and non-functional requirements were met and behaving as outlined.

8.2 - Reflection

Upon reflection of the project, many technical challenges were faced to create an interactive application with a system in place to sustain a user's requirements for the

product. Building a PHP platform that could interact and communicate with a database for storing and retrieving of information was a known challenge for the creator, as experience in PHP development was very limited at the conception of the project. However, an approach to develop the hardest challenges of the project first, as exclaimed by Stephen Hagan (mentor), before proceeding with the challenges that were more feasible was sound advice and proved to be highly beneficial for the project and ensuring milestone deadlines were on track. Successfully developing the user profile system and timeline interactions for the initial prototype meant that the two largest technical challenges with the most risk had been successfully created and were functional. There were still many more challenges to be faced through the project, but having these developed meant less stress and more time to focus on finer interactions and development.

In hindsight, however, a more robust framework should have been utilised for a project of this scope rather than expanding upon the Skeleton Framework which meant that significant time was spent on the layout and design of the application. Utilisation of a more robust framework, such as the Bootstrap Framework, would have meant more time could have been spent on technical challenges, user interactions, and visual design rather on device testing, browser testing and CSS development for the UI.

8.3 - Reflection on my role

Before deciding and commencing on any of the ideas that had been discussed for the Major Project, I perused feedback from a developer friend to ensure that each idea was viable and to provide a short breakdown of the technical challenges each idea would face. This was highly useful because I've found that a mistake that students can often make is taking on an idea with a lot of technical challenges that they don't fully understand. Final Year is a stressful year for everyone involved and my conviction was

always to take on an idea that was technical but simplistic in nature; the ethos was to under promise and over deliver rather than over promise and under deliver. That being said, there were still unforeseen technical challenges that appeared during development. My experience with PHP, javascript and backend programming at the beginning of the project was very limited. A lot of learning on the fly had to be achieved. In order to keep on top of things and ensure the project stayed feasible I stuck to the Gantt Chart and developed organisational skills for not only myself, with the creation of many to-do lists, but also for the application itself by using git version control and ensuring steady backups were in place.

It's safe to say I am now a lot more confident in my abilities as a developer as well as a designer. I have a much better understanding of how applications work regarding the front-end/back-end and how to build them.

8.4 - Suggestions for further work

Moodmemo is now a live project that anyone can sign up for and interact with. It has been hosted on its own server and will continue to be for the foreseeable future, if it remains active. It is the belief of the creator that a product should be continually improved upon to meet the growing needs of its user base. Therefore, there are many suggestions for further work in order to improve upon the experience for the user.

A feature that was never considered within the requirements of the project but upon reflection would have been a viable addition, is the functionality for a user to filter or search their timeline. A user may have created a considerable amount of Moodmemos and being able to filter or search for particular Moodmemos would be highly beneficial.

Additional features could also be added to increase the experience. For instance, a user may tend to create a majority of their Moodmemos from the same location, this means that on the Maps page there will be a considerable amount of Moodmemos that are very close to each other, making many indistinguishable and unclickable. Research was undertaken for a solution to this and a javascript plugin called `markerClusterer.js` was discovered that allows for markers within a predetermined radius to be clustered together to display a tally. This would be an ideal solution and can be implemented in future revisions of the product.

9.0 - References

Mentalhealth.org.uk, (2015). Mental Health Statistics. [online] Available at: <http://www.mentalhealth.org.uk/help-information/mental-health-statistics/> [Accessed 7 Apr. 2015].

Prompt.engineyard.com, (2015). Prompt - Mental health in the technology industry. [online] Available at: <http://prompt.engineyard.com/> [Accessed 7 Apr. 2015].

Geekmentalhelp.com, (2014). Mental Help Week. [online] Available at: <http://geekmentalhelp.com/> [Accessed 7 Apr. 2015].

MoodPanda.com, (2015). MoodPanda.com - Your Interactive Mood Diary. [online] Available at: <http://www.moodpanda.com/> [Accessed 7 Apr. 2015].

My thoughts, opinions and other notes, (2014). My thoughts, opinions and other notes. [online] Available at: <http://blog.brandonscott.co.uk/posts/anxiety> [Accessed 7 Apr. 2015].

Styletil.es, (n.d.). Style Tiles. [online] Available at: <http://styletil.es/> [Accessed 13 Apr. 2015].

Voiceandtone.com, (2013). Welcome | Voice and Tone. [online] Available at: <http://voiceandtone.com/> [Accessed 14 Apr. 2015].

Robertson, S. (2014). Creating Style Guides. [Blog] A list Apart. Available at: <http://alistapart.com/article/creating-style-guides> [Accessed 15 Apr. 2015].

Fronteed.com, (n.d.). Checkboxes and radio buttons customization (jQuery and Zepto) plugin. [online] Available at: <http://fronteed.com/iCheck/> [Accessed 21 Apr. 2015].

T4t5.github.io, (n.d.). SweetAlert. [online] Available at: <http://t4t5.github.io/sweetalert/> [Accessed 21 Apr. 2015].

Cassels, J. (n.d.). User Cake. [online] Usercake.com. Available at: <http://usercake.com/> [Accessed 21 Apr. 2015].

Whittaker, Z. (2012). MD5 password scrambler 'no longer safe'. [Blog] ZDNet. Available at: <http://www.zdnet.com/article/md5-password-scrambler-no-longer-safe/> [Accessed 21 Apr. 2015].

Walters, A. (2012). Social Login Buttons Aren't Worth It [Mailchimp Blog], [online]. Available: <http://blog.mailchimp.com/social-login-buttons-arent-worth-it/> [09/02/2015].

Facebook, n.d., Anonymous Login [Facebook Developers], [online]. Available: <https://developers.facebook.com/products/anonymous-login/> [09/02/2015].

Gowri sankar.R, 24/09/2013, Fuzzy time by CodeIgniter [Github], [online]. Available: <https://github.com/bcit-ci/CodeIgniter/wiki/Fuzzy-time> [08/02/2015].

Tutorialzine, (2009). Making Our Own Twitter Timeline | Tutorialzine. [online] Available at: <http://tutorialzine.com/2009/09/making-our-own-twitter-timeline/> [Accessed 21 Apr. 2015].

Usability.gov, (n.d.). System Usability Scale (SUS). [online] Available at: <http://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html> [Accessed 22 Apr. 2015].

Getskeleton.com, (2014). Skeleton: Responsive CSS Boilerplate. [online] Available at: <http://getskeleton.com/> [Accessed 28 Apr. 2015].

Stone, L. (2013). Quantified Self to Essential Self: mind and body as partners in health. [Blog] Radar | O'Reilly. Available at: <http://radar.oreilly.com/2013/12/quantified-self-to-essential-self-mind-and-body-as-partners-in-health.html> [Accessed 28 Apr. 2015].

Big Wet Fish Hosting, (n.d.). Big Wet Fish - We Don't do things by the Hosting Handbook. [online] Available at: <http://www.bwfhosting.com/> [Accessed 28 Apr. 2015].

Downes, J. (2010). Using Sketchboards to Design Great User Interfaces Quickly | Box UK. [online] Box UK. Available at: <http://www.boxuk.com/blog/using-sketchboards-to-design-great-user-interfaces/> [Accessed 28 Apr. 2015].

Littlethunder.co, (2015). Little Thunder Co. - A Little Web Design Co. From Belfast, Northern Ireland. [online] Available at: <http://littlethunder.co/> [Accessed 28 Apr. 2015].

Marksimonson.com, (n.d.). Proxima Nova – Mark Simonson. [online] Available at: <http://www.marksimonson.com/fonts/view/proxima-nova> [Accessed 28 Apr. 2015].

Pearson, C. (2014). UI Design for Empty States, Zero Data, and Onboarding - Rareview. [online] Rareview. Available at: <http://www.rareview.com/ui-design-for-empty-states-zero-data-and-onboarding/> [Accessed 28 Apr. 2015].

Goltz, S. (2014). A Closer Look At Personas: What They Are And How They Work (Part 1). [Blog] Smashing Magazine. Available at: <http://www.smashingmagazine.com/2014/08/06/a-closer-look-at-personas-part-1/> [Accessed 28 Apr. 2015].

Finkler, E. (2013). Open Sourcing Mental Illness. [Blog] funkatron. Available at: <http://funkatron.com/posts/open-sourcing-mental-illness.html> [Accessed 30 Apr. 2015].

Murphy, C. (2014). A Non-Graceful Shutdown. [Blog] fsck.monographic. Available at: <http://fsck.monographic.org/a-non-graceful-shutdown.php> [Accessed 30 Apr. 2015].

Clarke, A. (2014). Mental Help Week. [online] Geekmentalhelp.com. Available at: <http://geekmentalhelp.com/> [Accessed 30 Apr. 2015].

Ohlife.com, (2013). OhLife helps you remember what's happened in your life. [online] Available at: <http://ohlife.com/index.php> [Accessed 30 Apr. 2015].

Tumblr.com, (n.d.). Sign up | Tumblr. [online] Available at: <https://www.tumblr.com/> [Accessed 1 May 2015].

Rose, A. (2014). UX designers: Side drawer navigation could be costing you half your user engagement. [Blog] TNW. Available at: <http://thenextweb.com/dd/2014/04/08/ux-designers-side-drawer-navigation-costing-half-user-engagement/> [Accessed 1 May 2015].

Momentjs.com, (n.d.). Moment.js | Home. [online] Available at: <http://momentjs.com/> [Accessed 1 May 2015].

Appendix

Appendix One - Functional & Non-functional Requirements

Requirement	#1
Type:	Functional
Title:	Register
Description:	A web application that allows a user to sign up for an account
Rationale:	A user must be able to sign up in order to access the application features
Dependencies:	Requires user's first name, unique email address and a password over 6 characters that are POSTed and stored within a MySQL database
Fit Criterion:	99% of users should be able to easily navigate to the sign up page and successfully create an account

Requirement	#2
Type:	Functional
Title:	Login
Description:	A web application that allows a user to sign in to their account
Rationale:	A user must be able to login to the application to access their Moodmemos
Dependencies:	Requires that a user has registered and has access to their account details
Fit Criterion:	99% of users should be able to easily navigate to the login page and successfully login with their correct details

Requirement	#3
Type:	Functional
Title:	Post Moodmemo
Description:	A web application that allows a user to post a Moodmemo
Rationale:	A user must be able to post a moodmemo that consists of selecting an emoticon and writing their personal musings into a textbox as this is deemed as the main functionality of the application
Dependencies:	Requires that user is registered and signed in and locates the 'Create a moodmemo' button. User must select an emoticon and enter some text into the textbox.
Fit Criterion:	95% of users should be able to successfully locate the button to create a moodmemo and are able to easily, with little delay or puzzlement, create a mood memo using the necessary steps.

Requirement	#4
Type:	Functional
Title:	View Moodmemos
Description:	A web application that allows a user to view all of their created Moodmemos from within a timeline or other visual element
Rationale:	A user can view each of their created moodmemos on the index page of the application, where they can also view the selected mood and datetime for each Moodmemo created.
Dependencies:	Requires that user is registered and signed in and has created at least 1 Moodmemo
Fit Criterion:	99% of all users, if they have successfully created a Moodmemo, should be able to navigate to the home page of the application and view in a user friendly manner, all of the moodmemos they have created

Requirement	#5
Type:	Functional
Title:	Remove Moodmemos
Description:	A web application that allows a user to remove any or all of their created Moodmemos
Rationale:	A user reserves the right to delete any of their moodmemos at any time so this feature should be available to them without complications
Dependencies:	User must be registered and signed in and have access to a moodmemo within the timeline of which to delete.
Fit Criterion:	99% of all users should be able to find and successfully complete the deletion process of a Moodmemo

Requirement	#6
Type:	Functional
Title:	Change password
Description:	A web application that allows a user to change their password
Rationale:	A user should be provided with the option of changing/updating their password at any time
Dependencies:	Requires that user is registered and signed in, has access to their original password and is providing a new password that is 6 characters or over
Fit Criterion:	95% of all users should be able to locate the settings page from which to change their password and be able to supply the correct information to do so

Requirement	#7
Type:	Functional
Title:	Change email
Description:	A web application that allows a user to change their email address associated with their account
Rationale:	A user should be provided with the option of changing/updating their email address at any time as a valid email address is essential for an account in the web application
Dependencies:	Requires that user is registered and signed in, has access to their original password and is providing a valid email address
Fit Criterion:	95% of all users should be able to locate the settings page from which to change their email address and be able to supply the correct information to do so

Requirement	#8
Type:	Functional
Title:	Delete account
Description:	A web application that allows a user to delete their account and all associated data
Rationale:	A user reserves the right to delete the data that is stored on the application servers at any time and therefore this functionality should be provided to them
Dependencies:	Requires that user is registered and signed in, and is able to successfully enter the Prompt command to confirm the deletion of their data
Fit Criterion:	99% of all users should be able to access the settings page from which they can delete their account and successfully enter in the prompt correctly to delete their account and associated data

Requirement	#9
Type:	Non-functional
Title:	Favourite Moodmemo
Description:	A web application that allows a user to favourite one or more Moodmemos to be displayed in a user-friendly list within the user settings area
Rationale:	A user should be given the ability to favourite one or more Moodmemos in order to find it more easily when they wish to do so
Dependencies:	Requires that user is registered and signed in, has created one or more Moodmemos and is able to interact with the favourite button located within each Moodmemo.
Fit Criterion:	95% of all users should be able to access the settings page from which they can view their favourited Moodmemos

Requirement	#10
Type:	Non-functional
Title:	Change name
Description:	A web application that allows a user to change the first name/ display name associated with their account
Rationale:	A user should be provided with the option of changing the display name associated with their account. A first name isn't unique to a user's account and is not paramount in the functionality of their account but the ability to change this gives a user more ownership over their account
Dependencies:	Requires that user is registered and signed in, has access to their original password and is providing a first name that features 2 characters or more
Fit Criterion:	95% of all users should be able to locate the settings page from which to change their first name and be able to supply the correct information to do so

Requirement	#11
Type:	Non-functional
Title:	Calculate user's location
Description:	A web application that calculates a user's current location
Rationale:	In order to pinpoint on a map where a Moodmemo was created it is necessary to calculate the user's location at the time of posting a Moodmemo
Dependencies:	The latitude and longitude of the user's current location must be retrieved using Google Maps Geolocation. If the user has allowed for their location to be shared, then the latitude and longitude should be retrieved and posted to the database along with the other Moodmemo information
Fit Criterion:	90% of all location retrievals should be successful. This will be entirely dependent on if the user allows for their location to be shared and if the user's browser supports the HTML5 geolocation feature.

Requirement	#12
Type:	Non-functional
Title:	Plot user's location on Map
Description:	A web application that plots a user's location tracked Moodmemo's on a Google Map
Rationale:	To give the user a visual overview of the location at which their Moodmemos were created.
Dependencies:	If the latitude and longitude has been retrieved along with the information for that Moodmemo and stored in the database then the co-ordinates, along with the mood, datetime, and moodmemo text should be retrieved and used to populate the markers within the map
Fit Criterion:	99% of all moodmemos that have latitudes and longitudes associated with them should be retrieved from the database and successfully plotted onto the map in the form of map markers

Requirement	#13
Type:	Non-functional
Title:	On-boarding process
Description:	An on-boarding process to provide users with information on how to use key elements of the application such as posting a Moodmemo, and information regarding the statistics and interactive map.
Rationale:	To ensure user's are provided with instruction on the application when first using it
Dependencies:	jQuery will be used to determine if a user has not posted any Moodmemos yet, and if they have not, then append a div in place of the empty timeline that provides information to the user on what the timeline is used for and how to populate it. Cookies will be used to determine the first visit they perform to the map and statistics page and display a pop up providing them with information on these pages and why they are currently blank.
Fit Criterion:	The user must have no Moodmemos present in their timeline and be visiting the map and statistics page for the first time through that particular browser and/or device

Requirement	#14
Type:	Non-functional
Title:	Posting Moodmemo from all pages
Description:	Post a Moodmemo from any area of the application
Rationale:	Rather than having to navigate to a particular page of the application to create a Moodmemo it should be possible to do this from within any page of the application to ensure that users are always aware of how to create a Moodmemo and therefore do not need to navigate to a separate page to do so
Dependencies:	A button will be located within the header of the application that is present from all areas of the website.
Fit Criterion:	The user must have no Moodmemos present in their timeline and be visiting the map and statistics page for the first time through that particular browser and/or device

Requirement	#15
Type:	Non-functional
Title:	Signed up date
Description:	Display the date a user signed up at by displaying it in their profile settings
Rationale:	A user may want to know when they signed up to the application and how long they have been using the service for
Dependencies:	When a user registers, a timestamp must be created and subsequently stored within the database so this can be retrieved and converted from a MySQL timestamp to a human readable date
Fit Criterion:	User must be registered and have access to their profile settings

Requirement	#16
Type:	Non-functional
Title:	Statistics
Description:	Generate statistics on created Moodmemos
Rationale:	To generate statistics that can be presented to the user and provide them with information regarding their created Moodmemos
Dependencies:	Highcharts will be used to generate the charts and the information will be retrieved from the database by queuing it using php and injecting this into the charts using javascript
Fit Criterion:	User must have created at least one Moodmemo and have javascript enabled

Appendix Two - White Box Testing

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#1	Moodmemo post with special characters such as line spaces, tabs and single quotes.	Creating a Moodmemo and purposefully structuring it with special characters to see the result	The special characters will be formatted correctly but this will mean the Moodmemo text breaking outside of the confines of its parent container due to line spaces being interpreted as one character	The spaces and tabs are not formatted correctly and the Moodmemo breaks outside of the confines of its parent container. The solution to this has been outlined below

In **Test #1** it was expected that the test results would fail but that the spaces and tabs entered when creating the Moodmemo would be formatted as spaces and tabs when outputted. Instead none of the special characters were being formatted correctly. The reason for this is the characters were being escaped when being sent to the database (escaping data means replacing special characters such as ' with letters for security reasons) but were not being unescaped when outputted as Moodmemos.

FIGURE 6.1 - SOLUTION TO TEST #1

```
//remove any slashes
$moodmemo = stripslashes($_POST['inputField']);

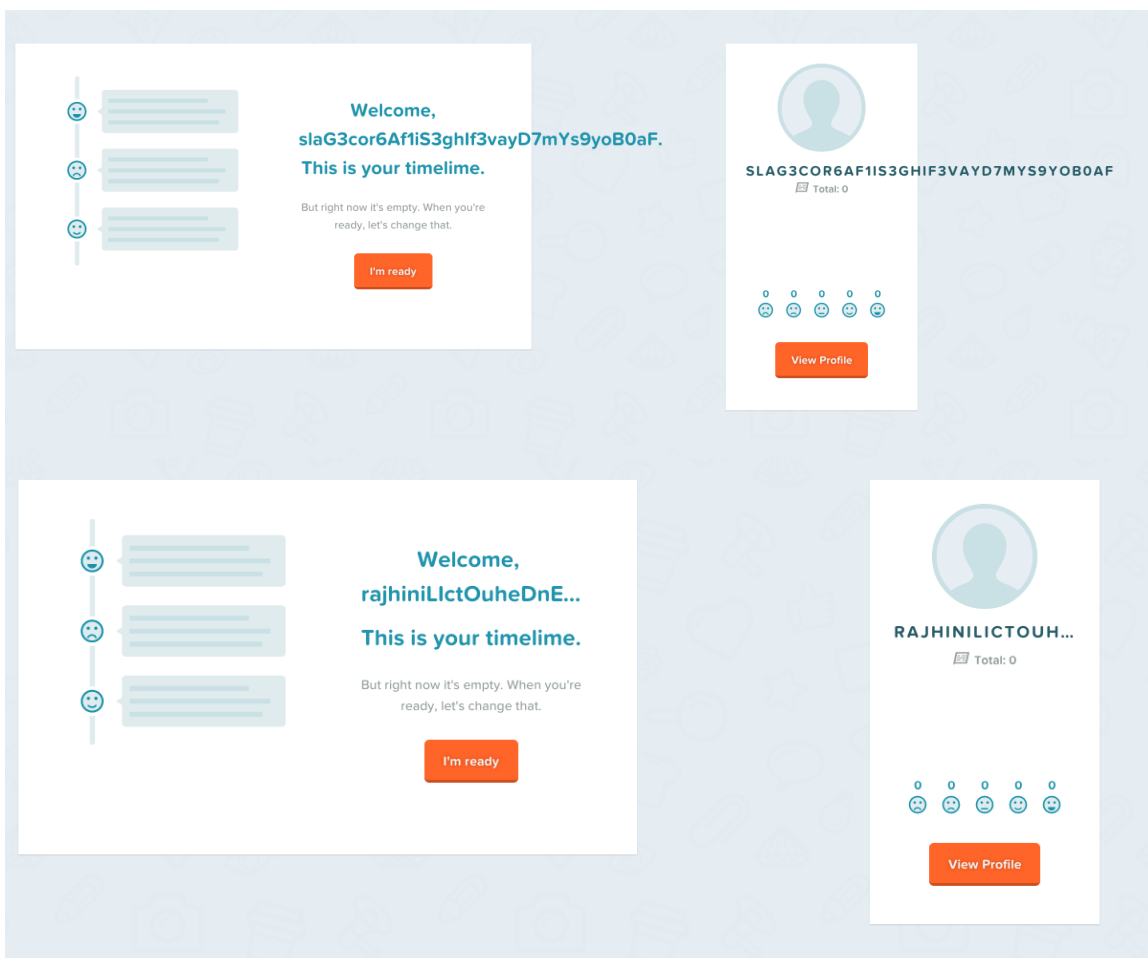
//format tabs and line spaces as spaces
$moodmemo = preg_replace('/[\r\n]+/', "\n", $moodmemo);
```

Figure 6.1 shows the two solutions that were used to fix the issue. The first issue was to use the stripslashes function to remove any slashes when posting a Moodmemo to

the database. The second solution was to replace any formatted line spaces and tabs with normal spaces using the preg_replace method. This meant that a Moodmemo would not break confines of its parent container if line spaces or tabs were used as these would be converted to spaces.

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#2	Max character First Name	Creating a first name that contains the maximum amount of characters (35) to test how to appears throughout the application	First name will be viable but will break alignment issues when displayed within the application	Same as expected outcome, a name with 35 characters causes layout issues within the application(see below)

FIGURE 6.2 & 6.3 - LAYOUT ISSUES CAUSED BY LONG FIRST NAME & THE SOLUTION



The solution was to use a fix width on the name and the CSS property 'text-overflow: ellipsis' in conjunction with "white-space: nowrap" to add three ellipses when a name flows outside of its container.

Test ID	Description	Expected Outcome	Actual Outcome
#3	Logging into account with correct and incorrect details to ensure the system is making valid checks for user details	Logging in with incorrect details triggers a validation error. Logging in with correct details brings user to the Timeline page.	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#4	Creating a Moodmemo and ensuring that the correct time is being posted to the database based on user Time Zone	The posted Moodmemo will display in the database and show the correct time that it was created.	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#5	Creating a Moodmemo and ensuring that the text is encrypted within the database	Moodmemos will be impossible to read when viewed within the database	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#6	Deleting a Moodmemo and ensuring that the data is deleted from the database	Moodmemo data will no longer be present in the database	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#7	Favouriting a Moodmemo will add the user_id and moodmemo_id of the current Moodmemo to the Favourites join table	Moodmemo_id and user_id will be accurate and present within tbl_favourites	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#8	Unfavouriting a Moodmemo will remove the user_id and Moodmemo_id of that Moodmemo from the favourites join table	When clicked the id and user_id associated with that Moodmemo is removed from tbl_favourites	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#9	Deleting a user account will delete all of the user's data associated with their account	When the 'Delete account' button is clicked and the confirmation is met the user's profile data will be deleted from tbl_users, and tbl_user_permission_matches, their moodmemo data will be deleted from tbl_moodmemo, and their favourites data will be deleted from tbl_favourites	Same as expected outcome

Test ID	Description	Expected Outcome	Actual Outcome
#10	Creating a Moodmemo with maximum character limit	Moodmemo will post successfully and will stay within the constraints of the Moodmemo container	Moodmemo appears blank. Further testing showed the issue to be database related. The field for storing Moodmemos had been set to a varchar with a character limit of 200 meaning only part of the encrypted hash for the Moodmemo was being stored within the field which resulted in the encrypted Moodmemo being indecipherable and appearing as blank. Changing the field to a text field with a much larger character limit solved the issue.

Appendix Three - Black Box Testing

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#1	Registration	User fills out registration form and clicks the sign up button	The sign up is successful and users are directed to a 'registered' where a button is present that when clicked, will take users to the timeline page	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#2	Login	User logs in using the credentials they signed up with	The login is successful and the user is directed to their timeline page	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#3	Delete Moodmemo	A user must be able to delete a Moodmemo from within their timeline by clicking on the more icon, selecting the trash icon and confirming the deletion of the Moodmemo.	The user successfully deletes the Moodmemo by clicking on the delete Moodmemo icon and confirming the delete	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#4	Delete account	A user must be given the option to delete their account and all associated data with it.	A user is prompted to type a command and if successful their account and all associated data is deleted and they are returned to the login screen	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#5	Favourite/unfavourite a Moodmemo	A user should be able to favourite and unfavourite any Moodmemo	A user clicks on the favourite icon within a Moodmemo and it is added to the Favourites section within the settings page. If the user clicks the favourite icon again it is removed from the favourites list	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#6	Change first name	A user should be able to change the name associated with their account	A user navigates to the settings page, enters in a first name that is 2 characters or longer and enters in their password to confirm changes. The updated name is now present and displayed.	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#7	Plot location on map	When a user creates a Moodmemo it should be plotted on a map if allowed by the user	The Moodmemo is plotted onto a map when the user has created a Moodmemo and allowed for location sharing to be shared.	Same as expected outcome. The location that is obtained using the Geolocation API will not always be accurate due to certain variables such as browser, ISP etc.

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#8	map and statistics modals	When a user visits the maps and statistics page for the first time a modal will display	A modal will display when the user visits the maps and statistics page for the first time. It will not appear again.	Same as expected outcome. The modals will display again if the user clears the cache of their browser or visits the pages from a different browser

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#9	Post Moodmemo from any page	A 'Create Moodmemo' button will be located in the header of the application and is accessible from any section of the product.	The user can click the 'Create a Moodmemo' button from any section of the application and a Moodmemo will be posted	Same as expected outcome.

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#10	Display user's registration date	User's should be provided with the information on how old their account is by displaying the date they signed up on within their profile settings	User can see their account age from within their profile settings which displays an accurate date of which they signed up	Same as expected outcome.

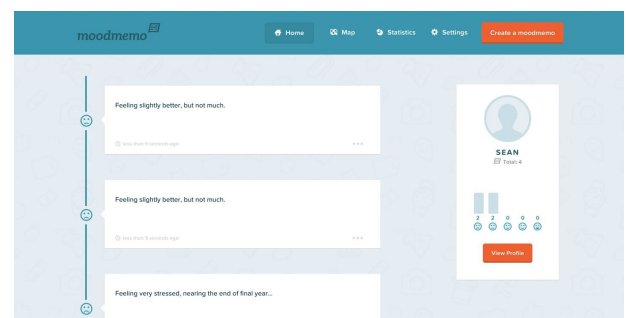
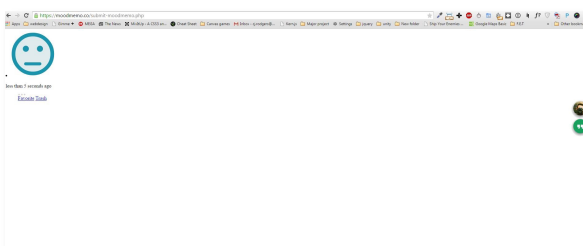
Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#11	Generated statistics	When a user creates Moodmemos statistics are displayed within the statistics page	Statistics will appear showing on a line graph the Mood that was chosen for the Moodmemo and the time at which it was created	Same as expected outcome.

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#12	User profile Image	A user should be able to upload an image that can be displayed as their own profile photo	A user can upload a file that is below 400kb in size and matches the possible image format that can be uploaded (jpb, png, bmp). The image will then replace the current default profile photo.	Same as expected outcome.

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#13	View timeline	A user must be able to view a timeline where they can see all of their created Moodmemos	The timeline is viewable from the timeline page and contains all of the Moodmemos the user has created	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#14	Post Moodmemo	A user must be able to post a Moodmemo by clicking the "create Moodmemo" button, given the choice of five moods to select from and a textarea to type their thoughts into	The Moodmemo modal appears where they are able to create and post their Moodmemo when the user clicks the 'Create a Moodmemo' button	Moodmemo posted successfully but the user was directed to a php file with the outputted Moodmemo and some Moodmemos were posting twice. The issue was with the Moodmemo form directing to moodmemo-submit.php file when the Moodmemo was submitted. Removing the 'action' field from the form fixed both of the issues.

FIGURE 6.3 & 6.4 - PHP REDIRECT & MOODMEMOS POSTING TWICE



Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#15	Recovering password	If a user forgets their password they should be able to retrieve a new password by entering their email address into the Forgot password form	User enters email address associated with their account. An email is sent asking them to confirm if they requested for their password to be sent. If they click the 'Allow' link they are then sent another email with a new password	Same as expected outcome

Test ID	Test Name	Description	Expected Outcome	Actual Outcome
#16	Change password	A user should be able to change the password associated with their account from within the settings page	User enters new password over 8 characters long, the user confirms the new password and enters their existing password to verify the changes. The user's password is now updated	Same as expected outcome

Appendix four - User Personas

Below is the text from the drafted User Personas that can be viewed in **Figure 2.3.8 & Figure 2.3.9** in **Chapter 2.3.5**.

User Persona 1

Meghan

Emotional, creative, articulate

Age: 17

Education: A-L:levels

Occupation: Student

Hobbies: Pinterest, Netflix

"I've started tracking more aspects of my life to give me a better sense of control over myself"

Meghan has been suffering from severe depression and self harm since she was 14. She was prescribed anti-depressants when she was 15 and has been taking them regularly ever since. She feels she is more stable now and is thinking of stopping to take her anti-depressants medication as she doesn't like how they make her feel "dull and empty".

Meghan still lives at home with her mum and two younger sisters. Her dad keeps in minimal contact and she sees him about once a year. She lives in a good home but

she feels ignored by her family and none of her friends seem to understand what she is going through.

Lately, Meghan has been trying to help herself by getting a gym membership and tracking her eating habits using MyFitnessPal. She feels that tracking her eating helps her feel a lot more in control and has started to also track her moods using Moodmemo. Meghan likes that she has somewhere safe and private where she can vent and describe how she's feeling over the course of a week. She has started using Moodmemo regularly over the remaining weeks to try and create an accurate timeline of her moods and how she is feeling.

I want something:

- I can use on my own, in my own time, anytime of the day.
- That is easy and intuitive to use.
- That feels personal and safe.

User Persona 2

Benjamin

Efficient, driven, boisterous

Age: 25

Education: Masters

Occupation: Student

Hobbies: Rock Climbing, Rugby

"I need to have a lot of structure in my life or it starts to become overwhelming."

Benjamin is currently in his final year of his Masters in Management. Benjamin has always been very career and education focused. When he was in Primary School he wanted to be a Doctor. However, after founding his first online business when he was in Grammar school, he decided that he wanted to be an entrepreneur and opted to go down the management route to learn how to run his own business.

Benjamin lives in London with his girlfriend. Despite her support, Benjamin is currently struggling with a lot of pressure and stress from running his own business and studying for his Masters. Benjamin acknowledges that he puts too much pressure on himself with meeting deadlines and constantly pushing himself. He thinks there is too much chaos and has opted for a more structured approach to his life.

One of the tools Benjamin has come across is Moodmemo. He has been using it for a while now to keep note of his moods and what he's currently working on. He likes that he can quickly create a moodmemo and then get back to doing what he was doing.

I want something:

- I can use quickly and efficiently.
- That helps me improve myself.
- That doesn't pressure me into using it.

