workulr

An Information Network for Companies

Project Report

Interactive Multimedia Design

2014

Steven Adams

B00561956

Mentor: Jonathan Wallace

Peer Support Group: 1

Acknowledgements

University

Peter Nicholl - *Course Director*

I would like to thank Peter Nicholl for his help in addressing problems that developed with the module. I would also like to thank Peter for constantly making himself available to give advice and inspiration.

George Moore - *Module Co-ordinator*

I would like to thank George Moore for his guidance on module deliverables and advice on best practices and approach.

Jonathan Wallace - *Mentor*

I would like to thank Jonathan Wallace for his constant help and advice in functionality and approach throughout the project.

Paul McCormack - *Design Lecturer*

I would like to thank Paul McCormack for his guidance in the direction of the user interface for my project.

Other

Christopher Johnson - *Senior software engineer, at Pace*

I would like to thanks Christopher Johnson for his help and guidance in solving development issues with Node.js

Christopher Grant - *Senior Front-End Developer at rehabtudio*

I would like to thanks Christopher Grant for his vast knowledge of JavaScript and taking time to help improve the code in the project.

Mike McNeil - *Founder of Balderdash and Creator of Sails.js*

I would like to thanks Mike McNeil for taking time out of his busy schedule to give guidance and one-on-one support on the Sails.js framework.

Contents

1. Introduction

1.1 The Challenge

In a medium to large workplace staff may sometimes feel disconnected from their colleagues or “out of the loop” on current affairs and upcoming events within the company. Project research indicated that there is a gap in the market for an application that companies can use to manage information for employees to easily access. Although some similar solutions exist, none do exactly what the application proposed in this project will deliver.

LinkedIn offers a place for professionals to store a portfolio that they can share with other professionals in an aim to make them more productive and successful **(About Us | LinkedIn. 2014)**. LinkedIn was not considered a competitor for the proposed application as it is more of a professional skills advertisement network. Companies can set up a closed Facebook group to post events and announcements, only allowing company members to join the group **(Facebook Groups | Facebook. 2014)**. Many companies will not however, permit access to Facebook on their network as it is considered a distraction. Yammer was the closest solution found to the proposed application - “Yammer is a Private Social Network for Your Company”. Yammer is a feature rich application which like Facebook, may distract employees from their work **(Yammer. 2013)**.

The project proposal was to develop workulr, an information hub for companies in the form of a web application that will be easily accessed on multiple devices from anywhere. workulr was designed to provide the following facilities:

- Allow a user to create an account and add their company to it
- Store information on that company
- Store information about company locations
- Store information on the people who work there in the form of short profiles
- Store company related news and events

The information stored in a company's account should be maintained by company account administrators.

The name workulr comes from a combination of the words work and circular as the purpose of the web application is to circulate information throughout the workplace.

1.2 Aim and Objectives

1.2.1 Aim

The aim for this project was to build an easily accessed information hub for companies in the form of a web application.

1.2.2 Objectives

- Relevant research carried out to ensure the end product would meet the needs of the target audience
- Clearly defined requirements of functionality for the end product
- Designs produced which encapsulate the key functionality
- Fully functional application completed within the project timeframe using chosen development languages
- Application hosted online in an environment capable of hosting chosen development languages
- Thoroughly tested application using beta testers and user feedback to ensure that all elements work as expected

1.3 Work Undertaken

This project required the learning of new technologies and putting the learning of these new technologies into practice in the development of the web application, workulr. It required research and planning prior to the development work, and subsequent testing, evaluation, and revision of the application.

1.4 Report Roadmap

The remainder of this report is divided into six chapters beginning at chapter two.
Chapter 2, Concept Definition and Testing, deals with how the idea for workulr was conceived, and goes on to discuss the selection of a suitable project methodology, the production of the requirements specification, and paper prototyping and feasibility testing. Chapter 3 - Design, considers the design evolution for the user experience, and develops this into a system design, logic design, and data design. Implementation is documented in Chapter 4, and includes details of the technologies and tools selected, how they were used, and the notable challenges and achievements of the project.
Testing is considered in Chapter 5, which outlines the various test strategies employed the processes used, and their results. It includes user evaluation of the application. Chapter 6 is an evaluation of the project, and reflects on all aspects of the process. Conclusions are drawn in Chapter 7.

2. Concept Definition and Testing

2.1 Idea Generation

The idea for workulr was inspired by the induction programme for new employees at rehabstudio. There the new employee is asked to send an email to everyone in the company containing an image of themselves and a short biography. The email is sent to everyone who works the company in all three of the company's locations around the world as a means of introducing the new employee to everyone. The new employee is then sent a collage of images of everyone who works for the company with their name under their image. The one thing missing from this induction programme was that the new employee could not see the current employees short biographies. Originally the names-to-faces feature was the stand alone feature of the application, but it was recognised that there was value in having other information such as; basic company information for staff to reference, and company related news and events in the same place.

2.2 Methodology Selection

When selecting a methodology for this project it was important to choose a methodology that would help to structure the workload and would assist in meeting deadlines.

2.2.1 Waterfall Model

The waterfall model is a rigid model as each phase of the model must be completed fully before the next phase is started, and the phases do not overlap. Due to this rigidity the waterfall model is easy to follow but does not lend itself well to necessary scope changes or blockers that may occur in later phases. **(Learnaccessvba.com. 2014)**

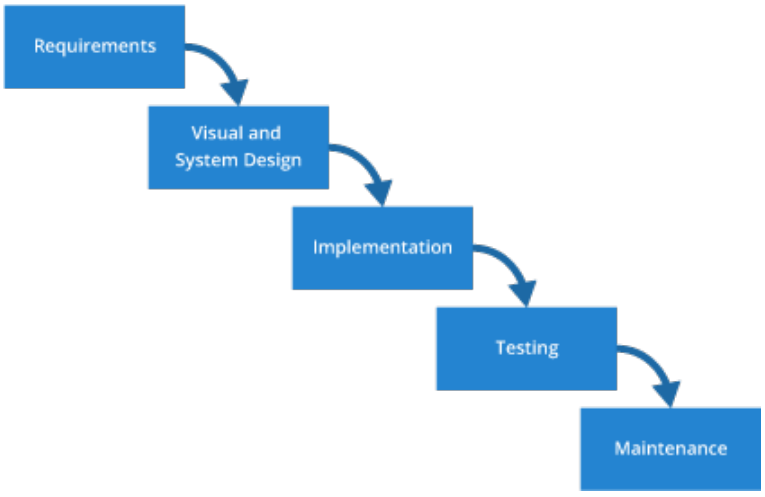


Figure 2.1 Waterfall Model

2.2.2 Modified Waterfall Model

The modified waterfall model is more flexible than the waterfall model. The main difference between the two is that the phases in the modified waterfall model are permitted to overlap. Multiple tasks can therefore be carried out simultaneously and phases can be returned to later if required. **(Satalkar, B. 2014.)**

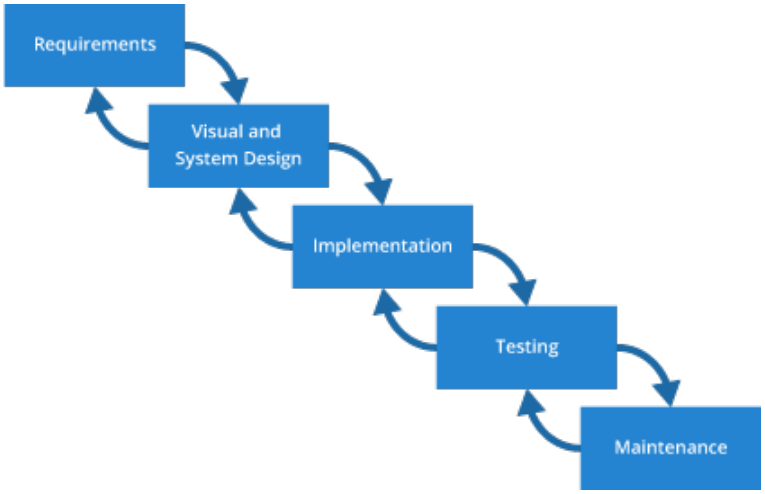


Figure 2.2 Modified Waterfall Model

2.2.3 Agile Model

This model can be considered as an incremental model as the application is developed in incremental, rapid cycles. This results in frequent deploys of the application with each deployed version building on the previously deployed version. Every deploy will be tested thoroughly to make sure the quality of the application is maintained. The main benefit of the agile model is that it allows for adaptation to changing circumstances. The main downfall of the agile model is that there is often a lack of emphasis on necessary design and documentation. This model is also more applicable to team projects.

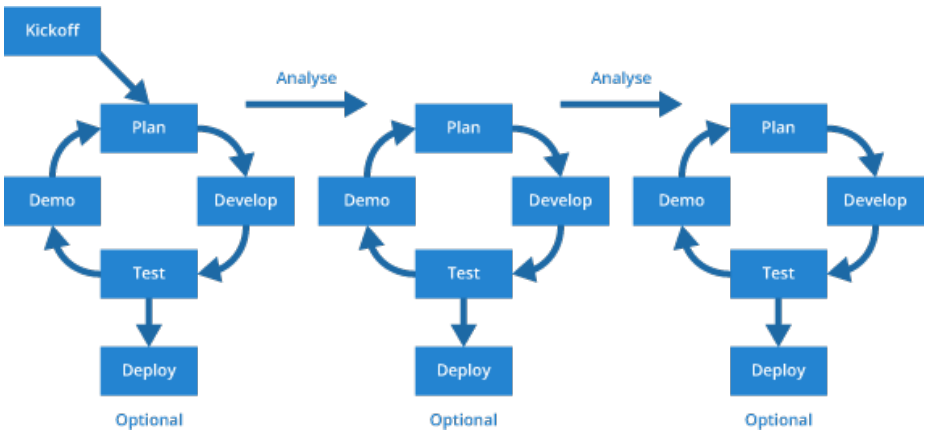


Figure 2.3 Agile Model

2.2.4 Chosen Methodology

It was decided that the modified waterfall model was the best fit for this project. This model provided a structured framework for project management while allowing phases that may need to be revisited. The modified waterfall model is also ideal for single person projects while the agile model is better suited to teams.

2.2.5 Time Plan

After selecting a methodology it was suggested that a time plan should be made to follow throughout the duration of the project. The most common method of producing a time plan for scheduling a project is to us a Gantt Chart. A Gantt Chart shows the different activities that shall be carried out to accomplish the completion of the project, when these activities begin and end, how long each activity is estimated to last, if activities overlap and by how much, and ultimately they show the start and end date of the entire project **(Mindtools.com. 2014)**. The time plan for workulr can be seen in ***Appendix A.***

2.3 Requirements Gathering

In order to establish the requirements for workulr, and thus build a successful product, it was necessary to engage with potential end users. Due to the target audience covering quite a wide market, consisting of medium to large companies, it was decided that the best overall perspective would be gained from gathering information from two different kinds of companies. The two ideal companies selected were rehabstudio (private sector), and Northern Ireland Housing Executive (public sector).

The feedback gathered from the potential end users should mainly reflect their opinions on the initial requirements but also allow room for additional requirements or improvements to the current requirements. The requirements specified after this requirements gathering stage needed to be clear, achievable and measurable as these requirements are the base for testing and evaluating the end product. The proposed methods to gather the requirements were through questionnaires, and casual discussions.

2.3.1 Questionnaires

A questionnaire was constructed and distributed to the companies named above, the potential end users of the product. Ten questionnaires were taken from each company. The feedback from the completed questionnaires was analysed and are reflected in the requirements for workulr. The questionnaire distributed can be found in ***Appendix B.***

2.3.2 Casual Discussion

While visiting the companies named above to distribute the questionnaires casual discussion were held with the person in charge of distributing information throughout the company and organising events etc. Brief notes from these discussions were recorded and any relevant suggestions are reflected in the requirements of the application.

2.4 Requirements Specification

After collecting and analysing the feedback from the questionnaires and casual discussion a set of requirements were composed.

2.4.1 Functional Requirements

There are two different user groups for the product; Company Administrator, and Standard User. The following are examples of the functional requirements of these two user groups with the full set of functional requirements available in ***Appendix C***.

Standard User

The requirements for a Standard User will be based on mostly read functionality, with editing their own profile being the only write functionality.

Requirement # :	1
Description :	The ability to log in to the company account they belong to
Rationale :	To keep everything secure and comply with data protection legislation
Dependencies :	Requires a Company Administrator to have created the company's account and added them to it

Requirement # :	2
Description :	The ability to edit their own profile
Rationale :	Allows the user to add/edit as much information about themselves as they wish
Dependencies :	Requires the user to have logged in to their company's account

Company Administrator(s)

The initial Company Administrator will be the member of the company who signs their company up for the workulr account. The initial administrator will have the option to set other users to be company administrators. This user group will have all the requirements of a Standard User with the additional requirements stated below, made accessible through a dashboard. The main role of the Company Administrator(s) is to populate their company's account with useful content.

Requirement # :	7
Description :	The ability to add/edit/remove company information for their company's account
Rationale :	Allows the Company Administrator to control the content for company information
Dependencies :	Requires their company's account to be active and for the user to have Company Administrative rights.

Requirement # :	8
Description :	The ability to add/edit/remove news and event items for their company's account
Rationale :	Allows the Company Administrator to control the content for on the news and events page
Dependencies :	Requires their company's account to be active and for the user to have Company Administrative rights.

Fit Criterion / Test Case - In the testing phase each of the Functional Requirements will be tested against whether or not the different user groups can carry out the functionality specified in the description of each of the requirements.

2.4.2 System Requirements

There are a number of system requirements which must be in place for the product to be operational. The following are examples of the system requirements for workulr with the full set of system requirements available in ***Appendix D***.

Requirement # :	12
Description :	The system shall allow users to sign up for an account
Rationale :	Uniquely identify the individual user of the system and provide secure access to data associated with their own company

Dependencies :	Front end system in place for data input. Back end system in place for data storage. Data validation system in place.

Requirement # :	13
Description :	The system shall save all data and any modifications to this data
Rationale :	Availability of data is a fundamental requirement of the product
Dependencies :	Backend database in place. Data validation in place. Secure login system in place.

Fit Criterion / Test Case - Similarly to the Functional Requirements, each of the System Requirements will be tested against the whether or not the functionality stated in their description can be carried out.

2.4.3 Non Functional Requirements

Non functional requirements are those which cannot easily be quantified but yet are considered importunes aspects of the project.

Look and Feel

- The product shall be appealing to all audiences of working age
- The product shall reflect current design trends
- The product shall appear open and friendly, but still trustworthy
- The product vocabulary shall be friendly

Usability

- The product shall be very intuitive and easy to navigate
- The product shall use easily interpreted icons for navigation

Performance

- The product shall give quick response to user interaction
- The product shall be available for use 24 hours per day, 365 days per year
- The product shall achieve 99.9% uptime
- The initial system shall be designed with a capacity for 50 companies with an average of 50 users
- The system will be designed in such a manner that it can be scaled to handle hundreds of companies and thousands of users

Operational and Environmental

- The product shall be usable on a wide range of devices including desktops, laptops, tablets and mobile devices
- The product shall be compatible with a wide range of browsers including the current release of Internet Explorer and the previous two releases
- The product will be solely available as a web application
- Updates to the product shall not cause failure of previously available features

Maintainability and Support

- Any updates to the system shall be deployed during off-peak hours i.e. weekends
- User supporter will be available through an online manual and via email
- Future releases of the product may be made available for companies on their own web servers

Security

- The system shall require users to log in to access their company's account
- The system shall be designed in such a manner as to prevent data corruption
- The system shall only provide users with information from the company account they belong to
- The system shall be hosted on a secure platform with appropriate protective measures to prevent hacking and other malicious interference

Legal

- The product shall comply with the Data Protection Act

2.5 Paper Prototyping

Paper prototyping is a fast and cost effective way to mock up the user interface of an application or website, there is no coding involved. It takes very little effort to sketch out multiple concepts where coding each concept would take considerably longer. The designer does not need to be artistically gifted, the purpose of the paper prototype is to evaluate the idea behind the user interface. This process can unveil a wide variety of problems in an interface, including many of some serious ones. The main advantage of paper prototyping is its fast production rate. This provides a way for a User Experience designer to show a client rough considerations for layouts in the application or website and suggest possible functionality. The client can then give instant feedback, the designer can make quick changes and the client can re evaluate. **(Medero, S. 2007)**

2.5.1 Six Up

Six Ups is the method of taking a problematic feature, section or page of a project, sketching out six possible solutions to that problem and choosing the option that works best. This method works best when working in a team, small or large. By working in a team there are more ideas generated which may lead to a solution that could have been missed had the task been carried out individually.

The problematic feature for workulr was that there was originally to be separate pages for News and Events. The issue was the layout of these pages and whether or not they should look exactly the same or should the layout differ between the two pages. It was suggested during the casual discussions in the requirements gathering stages that these pages may not contain much content and that both of the pages could be merged into one page. It was immediately evident that this would work well as having both News and Events on one page gives the user more information in one place, and removes a navigation step. Two challenges still remained - how is this page presented to the users, and how to differentiate between News and Event items. To solve these issues a small team was gathered to brainstorm six different possible solutions. A Six Up including some brief annotations was completed and can be seen in ***Figure 2.4***, with full size version available in ***Appendix E***. Annotation for each of the iterations shown in the Six Up follow on the next page.

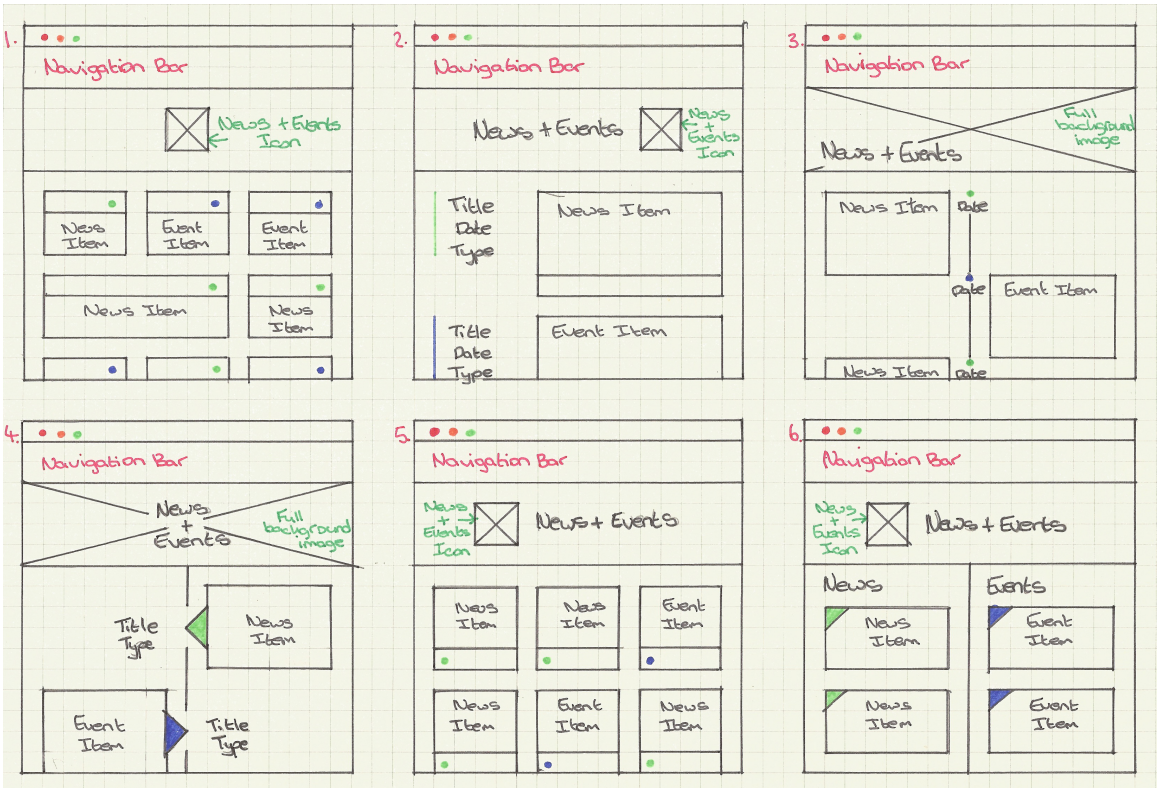


Figure 2.4 Six Up

Iteration One

A simple grid layout where the items are multiple widths and fit together, similar to that implemented by Pinterest but with a more solid structure. This layout gives the page a bit more character than the very plain grid version shown in iteration five. The item type in this iteration would be represented by a coloured dot in the top right corner of each item. The header in this iteration will be an icon which symbolises news and events.

Iteration Two

A layout implemented in a lot of blog websites where the left shows item information - the title, the date posted, and the item type. The right shows an excerpt of the item or an image. Item type in this iteration would be represented by text showing the type of item under the date posted. The item type would be reinforced by a coloured border to the left hand side of the item information. The header in this iteration will be the title “News and Events” and an icon which symbolises news and events.

Iteration Three

Shown in this iteration is a timeline layout inspired by Facebook. This idea was generated through a discussion within the team, where it was agreed that the timeline view, because it was more linear than rows of items, was easier to identify the order in which items should be read. The type of the items in this iteration are represented by the colour of the dot above the date posted on the timeline. This iteration was one of the favourites but it was felt that the layout of the items needs improved as there is white space the equivalent size of the item at the opposite side of the timeline. The header in this iteration will be the title “News and Events” in the bottom left corner of a full width banner image.

Iteration Four

Reflecting on the idea of a timeline layout in iteration three, a slightly different approach was taken. This iteration has the title and type of item (news/event) to one side and an excerpt of the item to the opposite side. Item type would be reinforced by a coloured arrow on the side of the item nearest the timeline. The header in this iteration will be the title “News and Events” in the middle of a full width banner image.

Iteration Five

A very simple block grid of items, which would be similar to the grid used for the desktop web view of Instagram. The item type would be represented by a coloured dot in the bottom left corner of each item. The header in this iteration will be the title “News and Events” and an icon which symbolises news and events.

Iteration Six

A divided layout between the news items and the events items. There is no need to represent the type of item in each individual item as both types are in completely different sections of the page. As a means of experimentation another method of representing item type by showing a coloured coded corner in the top left of the item. The header in this iteration will be the title of the page “News and Events” and an icon designed to symbolise news and events to the left.

2.5.2 One Up

After the Six Up some time was taken to reflect over the six iterations. Some further ideas and suggestions were generated through discussions with design advisers and members of the small team, and a final paper prototype was developed. It was decided that the page should be renamed “Broadcast”. This renaming was decided upon because it is shorter and reflects the purpose of the page as Broadcast means cause to become widely known **(TheFreeDictionary.com. 2014)**. It was also noted that because the intent was to use icons in the primary navigation it would be much easier to represent Broadcast in an icon rather than News and Events**.** The final paper prototype can be seen in ***Figure 2.5***, with a full size version available in ***Appendix F.***

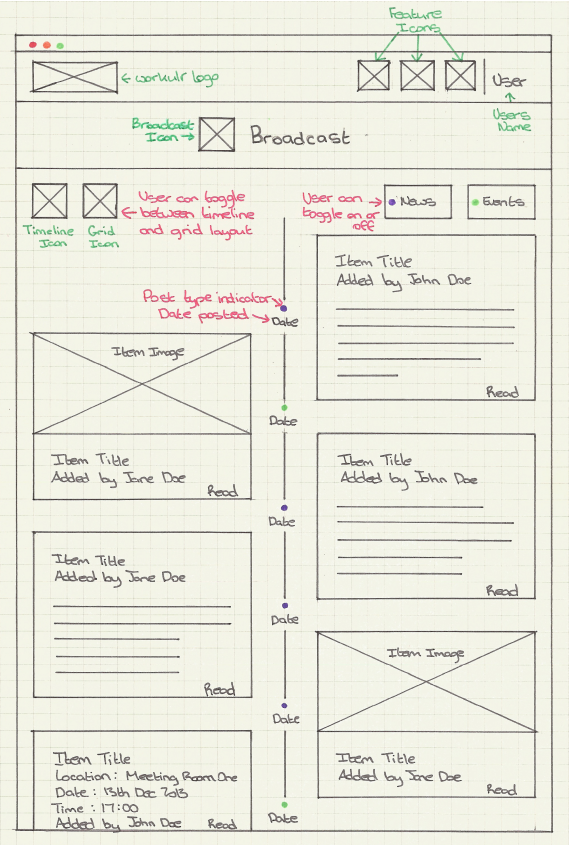


Figure 2.5 Final Paper Prototype

The layout option decided on was an improved version of iteration three. It was suggested that if the only way to identify the item type was a colour coded circle a key would be required to show what each colour represented. It was suggested that this key could be represented using buttons allowing the user to toggle between viewing only news or events items. The final potential feature suggested was the ability for the user to switch between the timeline view to a grid based view. In the final paper prototype the layout issue discussed previously in iteration three of the Six Up was solved. The issue was that there was too much white space below each item. This was solved solved by moving the type indicator and date posted to the middle of the item on the timeline. This allowed for the distance between the items on the timeline to be decreased. While the discussed potential features are shown in

the final paper prototype this does not necessarily mean they will be included in the final implementation of workulr. The Six Up and final paper prototype exercises were very valuable and lead to the solving of a problematic feature while generating some great potential features.

2.6 Feasibility Testing

2.6.1 Technology Feasibility

The technologies to be used for the project have been well established in the field of web development. It is the application of these technologies rather than the technologies themselves that are unique to workulr. As these technologies have been used to develop many similar products this provided a high degree of confidence that they would provide an appropriate toolset for this project.

2.6.2 Project Risks

Before progressing further with the development of workulr it was important to outline any risks that might occur. The level of each risk will be rated using the Traffic Light Rating System **(Wikipedia | Traffic light rating system. 2014)**. An Initial Risk Level (Green, Amber, or Red) will be given for each risk by taking the Initial Impact and multiplying it by the Likelihood. The key for this rating system is shown in the ***Figure 2.6***. It is important to note that should a risk be rated Red, the project should not go forward until this risk is lowered. Mitigation measures for each risk will be determined, if possible. These measures are designed to reduce the impact of the risk, and this is reflected in the Post-Mitigation Impact x Likelihood and Residual Risk Level. The Residual Risk Level is worked out as the product of the Post-Mitigation Impact and Likelihood.

	Green (Low)	**Amber (Medium)**	**Red (High)**
Impact Level x **Likelihood Level**	Score of 1-9	Score of 10-16	Score of 17-25

Figure 2.6 Traffic Light Rating System Key

Some examples of the risks that might occur are:

- Development machine crashes fatally, losing all project files
- Development may be slowed down by need to learn new development languages
- Data from one company's account is exposed to members of another company's account.

The full risk register including impact, likelihood, initial risk level, mitigation, post mitigation impact x likelihood and residual risk level is shown in ***Appendix G.***

3. Design

3.1 UX design Evolution

3.1.1 Visual Design

The application relies heavily on visual design to maximise its usability. This means ensuring that the application works in a way that a person of average ability or experience could use it for its intended purpose. The interface must be intuitive so that the user can use the site without having to think about it too much, and therefore give them confidence in the application they are using.

In an article on Smashing Magazine, Joshua Gross, principal at Planetary, a design and development agency in Brooklyn said “Most designers spend too much time with their designs to be objective about them. The best thing any designer can do is to collect feedback from real users” **(Gross, J. 2013)**. Taking this on board, this report reflects both the opinions of those who are close to the project, and the opinions of others who are completely disconnected, for example family members who have no design background but are frequent web users.

Research identified a psychological phenomenon know as the availability heuristic. This is where users look for, and recognise faster, patterns they have experienced before. This means that if an application is designed following a pattern that a user is likely to be familiar with, the user generally perceives the interface to be more friendly and easier to navigate, therefore enhancing the user experience **(Cherry, K. 2014)**. The visual design of workulr should therefore conform to the current design style trends and layout of other modern web applications focusing closely on social network based applications. workulr itself is not a social network application, it does however have the similar purpose of sharing information, just not necessarily in a social aspect - it could be referred to as an information network.

Overall Design

It was decided from the outset that workulr would incorporate a flat and minimalist design for the user interface. Flat design is a current web design trend that is taking over the way almost all new websites or applications are designed. While the main reason for using flat design is conforming to the latest design trend there are also other benefits of this kind of interface. Most modern web users now appreciate simple styling and styles that they recognise. Non frequent web users will also appreciate this style of design as it portrays a sense of simplicity and that the website or application will be easy to use **(Clum, 2014)**.

Concerning minimalism, the user experience is enhanced by de-cluttering the screen. The minimalism design method originated in print media design where white space (or space in general) was more evident than the text. The text has what is described as “breathing room” and suggests a typographical peace, making the text more readable. Minimalism does not only apply to typography, using empty space wisely by removing unnecessary elements from the page and using this space to give the remaining elements breathing room will draw the users to what it is you want them to see. workulr also makes use of rounded corners on block elements where applicable as this portrays feelings of friendliness and comfort to users. The page layouts were structured using a grid system which enforces a sense of solidarity and organisation.

Colours

workulr will make use of five main colours; blue, navy, green, off white (grey) and white. These colours will be shown later in this section in a style tile. These colours may be darkened or lightened when used in the application interface to give feedback of interaction to the user, examples of this can be seen in the section titled **3.1.6 Interaction**. The main colours when classed together are blue, green, and white, focusing heavily on blues. Blue is the most commonly appealing colour to both genders, it portrays a sense of loyalty and productivity. The colour blue is also widely associated with social networking site and while workulr is not a social network there is a certain social aspect to the product. Green is a positive colour and therefore is a obvious choice for action buttons, giving the user a sense of assurance when clicking on the button. Finally, white portrays a sense of modernism, safety, positivity and happiness.

Typography

The typography for workulr will be set in Open Sans, a sans-serif font. Using a sans-serif font adds to the modern look envisioned for workulr as well as being more legible at smaller sizes and on the web in general. The initial selection was Proxima Nova, but to implement this font would have required a very expensive licence, or to have used Typekit which would be cheaper but not 100% reliable. Open Sans is a very close match for Proxima Nova and gives the option to host the fonts on the application server. The font is made available for use freely by Google Fonts for web and print and has been used in various Google web applications and other web applications including the Rdio home page. Open Sans is optimised for legibility across print, web, and mobile devices - the three places the font is likely to be used. This font will be used for all typography on the site including headings and body copy. The copy hierarchy will be clearly defined through the use of size, weight and colour, this will also be shown later in a style tile. **(Open Sans. 2014)**

Style Tiles

In an age where there are so many screen sizes, it is pointless to design complete compositions for a selected number of different screen sizes, and too time consuming to design complete compositions for every screen size. The newest design methodology, style tiles, removes this issue. Style tiles were created by Samantha Warren as a method to "Present clients with interface choices without making the investment in multiple photoshop mockups" **(Style Tiles. 2014)**. A style tile is a graphic which shows a number of common web page elements including colours, typography, textures, patterns and other design features. They are not a direct representation of how the full website or application will look, but define the general look and feel of the website or application. Shown in ***Figure 3.1*** is the final style tile for workulr, for the original please see ***Appendix H***.

Figure 3.1 workulr Style Tile v2

Responsive

With the increase of mobile web browsing it was assumed that a significant proportion of the target audience for workulr may wish to browse the web application on their tablet or smartphone as well as their desktop. Studies have shown that many web users will rarely use a desktop machine to access the internet, and increasing numbers of users are solely mobile web users.

With this in mind it was decided that workulr should incorporate a fluid, responsive design with three main breakpoints - desktop, tablet, and mobile. Breakpoints allow for page layout and styling to be manipulated when the viewport gets to certain widths and heights. For workulr and most other websites the width of the viewport is the most important. There are multiple different sizes of desktop, tablet, and mobile viewports and that is why a fluid responsive design is the best method, as opposed to selecting three main viewport widths and changing the layout to suit those set widths. Adding fluidity allows the layout widths to increase and decrease to fit any device viewport that falls within the breakpoint parameters. A fluid width layout is achieved by using percentages for widths rather than pixels on the elements.

By manipulating the layout and design this provides the user with an easily accessible user interface that fits their screen rather than presenting them with a scaled down version of the full design layout. This provides each user with the best possible user experience regardless of their screen size. Shown in ***Figure 3.2*** is an example of the benefit of a responsive design using the people avatar elements layout which will be implemented in the “People” page.

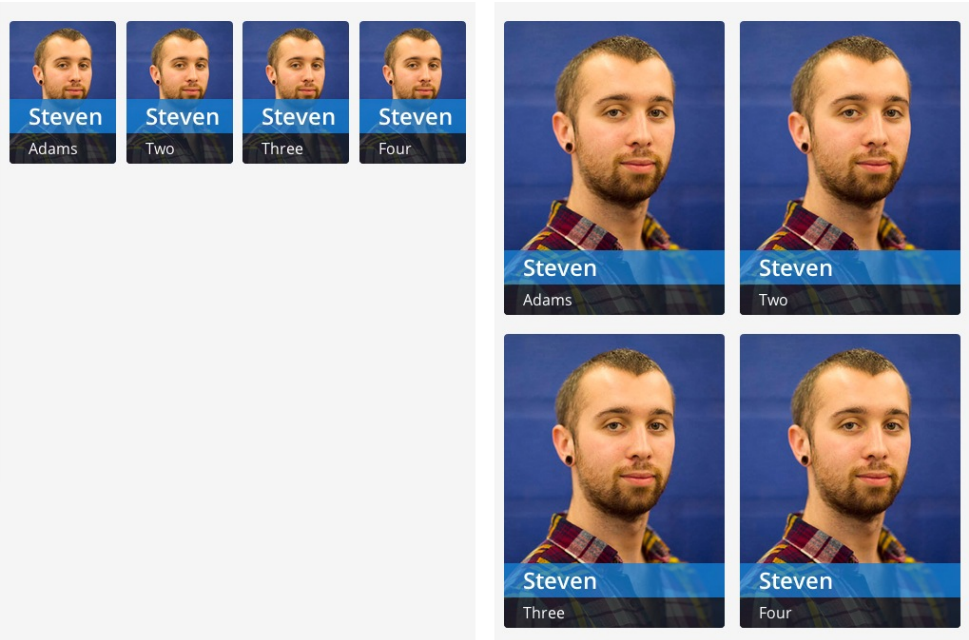


Figure 3.2 Responsive Example

Icons and Icon Font

workulr will make great use of icons throughout it's user interface. It is vital that these icons are legible at both small and large sizes. It is also vital that these icons are intuitive so the user of the application is able to make a quick association with the icon and what it resembles. With a lot of new applications mimicking the line icon style made popular by Apple in iOS7 it make sense to follow suit with workulr's icons. Examples of these icons are shown through this report.

There are a few methods that could be used to display these icons in the user interface, such as single image files or a sprite sheet. Single image files requires too many asset requests, which slows down the page load time, and sprite sheets would require two sprite sheets, one for HDPI screens and one for standard screens. Generating two sprite sheets is not particularly difficult, but it does require the use of media queries to determine which sprite sheet to use. The method selected to deliver these icons is to use an icon font. Icon fonts render in the same way as normal fonts - regardless of the size the font is set to, the icons will render crisply. There are many brilliant icon fonts available - some are free like Font Awesome **(Gandy, D. 2014)**, and some come at a cost such as Symbolset **(Symbolset. 2014)**. There are two main problems with using an already available icon font - the icons are not unique to the application, and the whole set has to be loaded, not just the icons required for the application. The solution to this is to design a custom icon set a generate a font based on this set of icons, there are a number of methods to do this.

The first method is to use a icon font generations web application which allow the generation of an icon font by uploading a set of SVG (Scalable Vector Graphics) file such as Fontello **(Fontello. 2014)** and IcoMoon **(IcoMoon. 2014)** . As this icon font was to be created during the implementation stage of the project, having to upload a folder of icons every time a new icon was added or changed was seen as time consuming. The decided solution for this application was found in FontCustom **(Gross, Y. 2014)**. This program allows the generation of icon fonts through the command line by setting FontCustom to watch a folder that contained SVG files of all my icons. FontCustom also generates a CSS file that is included in the application. Developing a custom icon font for workulr meant that the icons in the font were completely bespoke to the application. It also meant that the file size of the icon font was kept as small as possible as only icons that were necessary to the design were included. Shown in ***Figure 3.3*** are the icons used for the icon font - all of the icons were custom designed for workulr in Adobe Illustrator and have a consistent stroke thickness.

Figure 3.3 Icons for Icon Font

3.1.2 Branding

Branding is not solely the logo for the application, it is the entire persona of the application. The branding of an application is the difference between that application and any other available option, and what makes a user pick that application over any other option. Branding is what the users of the application say about it, as social proof is a great way to persuade users to pick your application.

Persona

If workulr was a person he/she would be the well presented colleague from the company whom everyone knows and finds easy to get on with. He/she is trustworthy and very knowledgeable about company news and events. workulr is very friendly and makes the effort to get to know a little bit about everyone he/she works with. workulr is helpful but not patronising, simple but not boring, trustworthy but not complicated, knowledgeable but not big headed, and easy-going but not sloppy. workulr will have a warm, friendly, and recognisable voice. The voice should come across as more human, loose, and free flowing than a typical web app, in a way that colleagues or friends might speak to one another **(Walter, A. 2014)**.

Logo

The logo is a vital aspect of the branding, it gives a user a symbol which identifies that application or product. A logo can be an icon, word mark, letter mark, or combination mark. The primary consideration for the logo was a simple word mark along with a letter mark that could be used interchangeably. There various examples of this approach that are globally know such as Facebook, Tumblr, and Disney which are all instantly recognisable in either form and are shown in ***Figure 3.4***.

Figure 3.4 Brand Logos

Taking these examples as inspiration experimentation was carried out using different fonts for the word mark. The font choices were narrowed down to six possible options and are shown in ***Figure 3.5***.

Figure 3.5 Font Options

Options in ***Figure 3.5*** - 1. Futura Medium, 2. Gotham Rounded Medium, 3. Brandon Grotesque Medium, 4. Bookman Old Style Bold, 5. Agenda Bold, 6. Open Sans Semibold.

The initial preferred option was Gotham Rounded Medium, it had a modern look and its rounded edges gave a friendly feel without looking childish. The selected peer group were asked for their opinions on the options. Some of the peer group preferred Agenda Bold as it stood out. Others in the group also chose Gotham Rounded Medium. A general opinion within the peer group was that the majority of the options looked very similar and that Bookman Old Style Bold was too heavy and dated. It was decided that Gotham Rounded Medium was the best option but it was also agreed that the word mark as it stood needed some additional work to make it look unique and original. After some modifications the final word mark for workulr was developed and is shown in ***Figure 3.6.***

Figure 3.6 Final Word Mark

After the final word mark was created the first letter of the word mark was taken and experimented with to generate letter mark options. Again this experimentation was narrowed down to six options and presented to the selected peer group for their opinions. These options are shown in ***Figure 3.7***

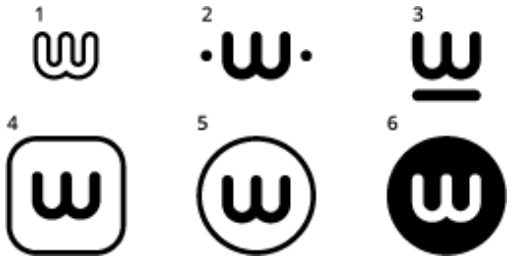


Figure 3.7 Letter Mark Options

The favourites among the selected peer group were options 3, 4 and 6. It was said that option 3 stood out as being different, option 4 resembled strength and reliability with its square shape, but yet easy going with its rounded edges, and option 6 would link in with the brand name which comes from 'work' and 'circular'. Some of the peer group said that circles were too common for letter marks and did not really work in a lot of situations. Some of the peer group opted for option 4 saying that it was more

versatile and looked like an app icon. None of the peer group liked the first three options. The opinions of the peer group were taken into consideration and it was decided that option 4 would be used but there would be multiple options for it to be displayed in, these are shown in ***Figure 3.8***.

Figure 3.8 Final Letter Mark Display Options

After the logo options for workulr were developed it was important to imagine how this would look when applied in different contexts, both online and offline. Through this process it is possible to gain an idea of brand guidelines for workulr that could be published if workulr was put into full production. Brand guidelines are important for the logo among other aspects of the branding so that the logo appears consistently the same wherever it is so that it becomes easily recognised. Some examples guidelines for using the logo would be the spacing around the logo, and logo application. The spacing around the workulr logo should be a minimum of half the overall height of the logo - shown in ***Figure 3.9*** using the word mark.

Figure 3.9 Logo Spacing Guidelines

The application of the logo is another important aspect to include in the guidelines. The workulr logo can be displayed on any background in the blue, navy, white defined in the style tile in the previous section, in any opacity. The logo should always be legible against its background. The workulr logo can be displayed in black for greyscale printing if necessary. Some examples of acceptable application are shown in **Figure 3.10** using the word mark logo.

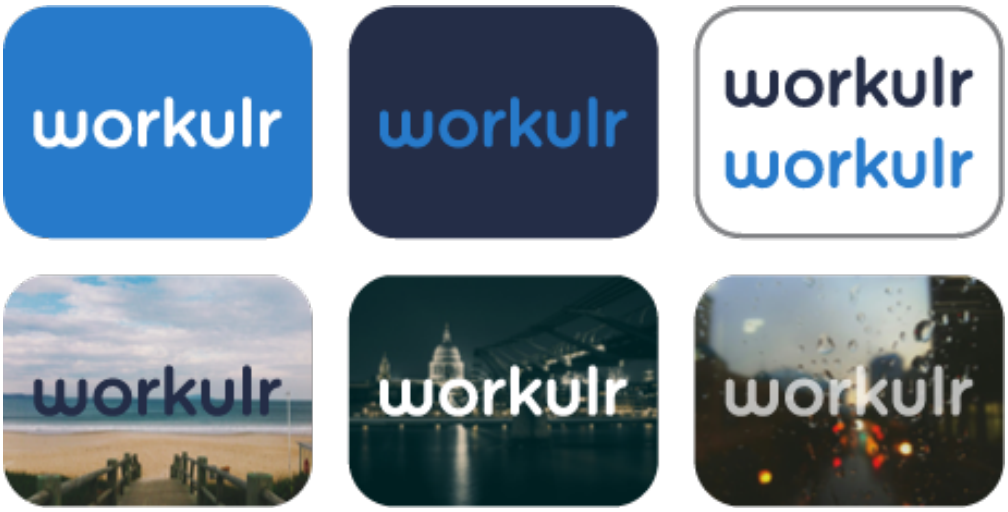


Figure 3.10 Logo Application Guidelines

3.1.3 **Initial** Navigation

Navigation is vital for any website or web application, this is how the user explores the contents of a website or web application. Navigation, specifically the primary navigation, needs to be consistent from page to page, this helps the user learn the navigation system and not feel lost or frustrated. Typically websites or web applications will use three type of navigation; primary, secondary, and tertiary where necessary. workulr will make use of primary and secondary navigation throughout it's user interface to allow the user to easily navigate their way through the application.

Primary Navigation

The original concept for the primary navigation of workulr was to implement a navigation bar horizontally along the top of the page. This style of primary navigation follows the most common primary navigation style seen in web applications such as Facebook, Twitter, and Dribbble shown in ***Figures 3.11, 3.12, and 3.13***.

Figure 3.11 Facebook Primary Navigation

Figure 3.12 Twitter Primary Navigation

Figure 3.13 Dribbble Primary Navigation

The primary navigation should contain the elements such as the workulr logo. Users commonly associate this with taking them to the initial page they started from. Having this functionality available and easy to find gives the user the confidence that even if they get lost in the site they can always start from the start. The primary navigation should also contain links to the key features of the application; company information, people, and broadcast so they are easily and directly accessible by the user. The final thing that need to be included in the primary navigation is the name or avatar of the logged in user. This navigation bar should be consistent throughout the application so it is easily found by the user. As the modern web user is impatient and easily frustrated, implementing good primary navigation will increase the likelihood of the user continuing to use the web application and returning to use it again. Having a consistent layout and design assures users instantly that the page they have navigated to belongs to the same application. The initial primary navigation solution is shown in ***Figure 3.14***.

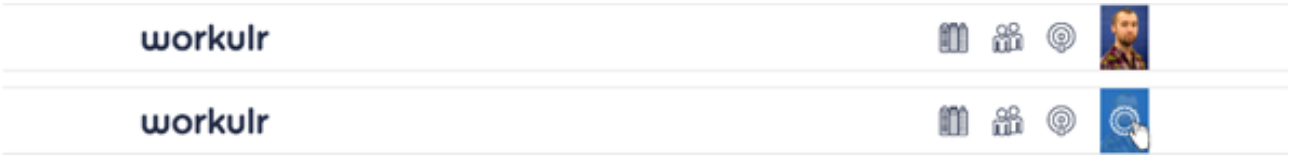


Figure 3.14 Initial Primary Navigation

Secondary Navigation - User Options

Initially there was to be two instances of secondary navigation for workulr. The first of these was the additional options for the logged in user. These options would include viewing their own profile, editing their profile information, and logging out. The most common way for this type of secondary navigation to be implemented is a drop down from the users name, avatar, or a settings icon. Shown in ***Figures 3.15, 3.16, and 3.17*** are examples of this style from Twitter, Dribbble, and Foursquare. It was considered that conforming to the general look and feel most web applications would be applicable, and that implementing something similar to the examples shown in ***Figures 3.15, 3.16, and 3.17*** would work for workulr.

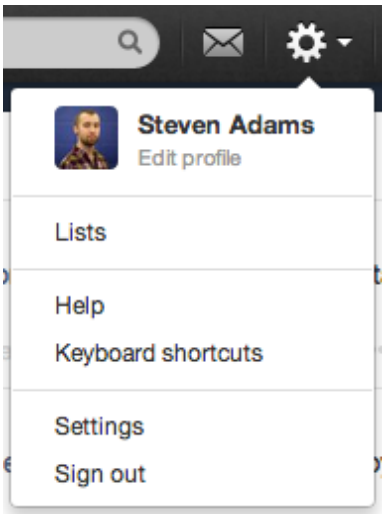


Figure 3.15 Twitter User Options

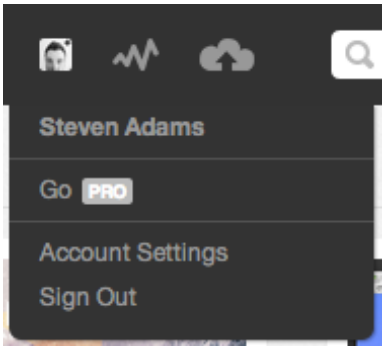


Figure 3.16 Dribbble User Options

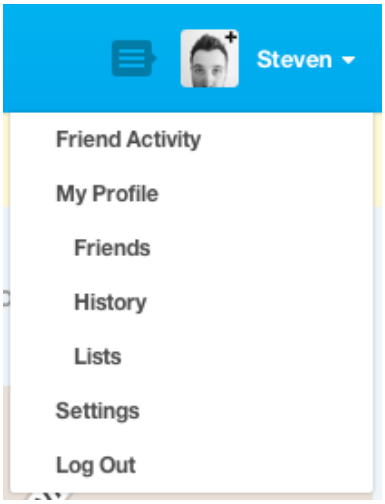


Figure 3.17 Foursquare User Options

It was decided to try a different style of secondary navigation for the user options. ***Figure 3.18*** shows how the initial secondary navigation for user options will look. This concept shows when the user has clicked on their avatar, and the user options are shown. The user options slide down from above the navigation bar and stay active until the options icon is clicked again. A selected peer group were asked to review this user option navigation style. They all said that they would have no issue using this style of navigation. It was also stated that even if the menu was left toggled open that it was not obtrusive.

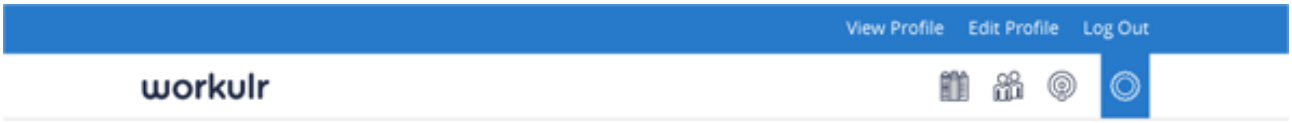


Figure 3.18 Initial Secondary User Options Navigation

Secondary Navigation - Footer

The second instance of secondary navigation considered was the footer. The footer navigation should contain links to information that is less important and less desirable by the user but still may be required such as FAQs, terms and conditions, press, social links etc. Some applications have large, complex footers with many links and some applications try to keep their footer very minimalist with only a few links. Dribbble would be an example of one of the larger footers, shown in ***Figure 3.19*** and Facebook would be an example of the more minimalist approach shown in ***Figure 3.20***.

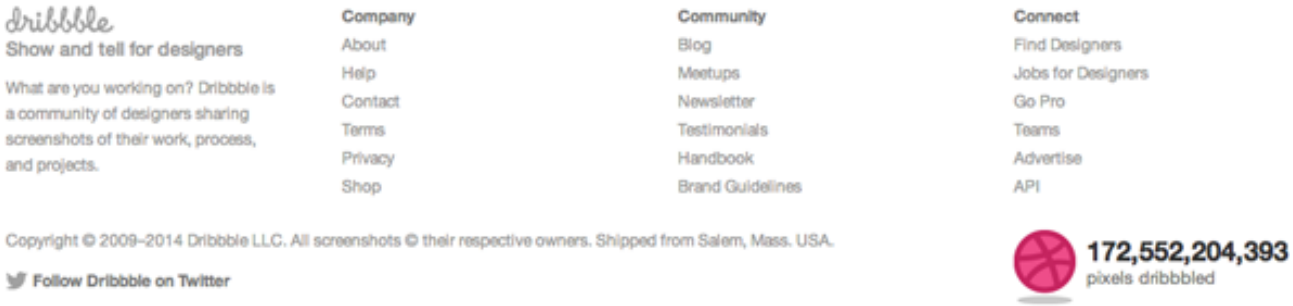


Figure 3.19 Dribbble Footer Navigation

Figure 3.20 Facebook Footer Navigation

It was decided that somewhere between these two examples would work well. The footer for workulr should not be as crowded with content like Dribbble but not as minimalist as Facebook. Shown in ***Figure 3.21*** is the design for how the footer of workulr should look and examples of the links that it might contain. The footer design shown is as tall as that of Dribbble but with less links. The links are given enough spacing so that they can be read easily. Figure 3.21 workulr Footer Navigation

Figure 3.21 Workulr Footer Navigation

Mobile Navigation

It was mentioned previously that workulr will incorporate a responsive design and therefore elements of the pages will have to be reconsidered for tablet and mobile devices. The primary navigation and user options are two of these elements. While the contents of the navigation bar and the user options are very minimal and would fit in the width allowed by most tablet devices, mobile devices will need a change of layout to comfortably fit smaller screen sizes. There are many different ways to implement this type of navigation for mobile devices. A recent trend is to have navigation slide out from the side and move the main content over. An example of this style would be Facebook's mobile browser navigation shown on in ***Figure 3.22***. This option is considered to be more suited to a navigation which contains lots of options, where workulr has six main options, three of which are primary navigation and three of which are secondary user options.

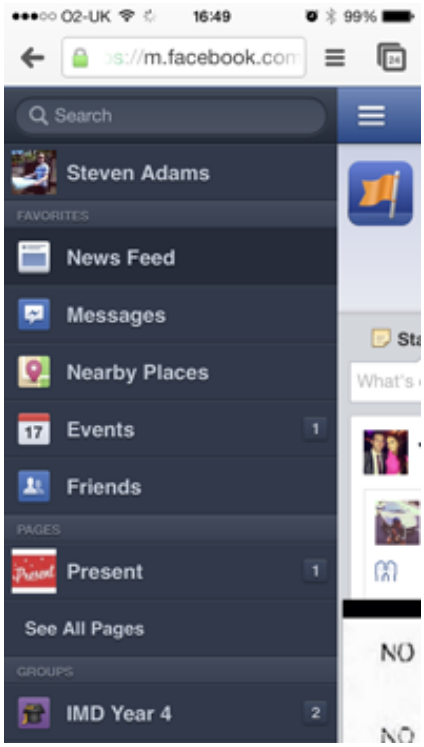


Figure 3.22 Facebook Mobile Navigation

Another common trend for mobile navigation is to use a drop down or slide down menu where the options for the menu appear either above the navigation bar, or below the navigation bar. It was initially felt that this type of mobile navigation was more suited to workulr. ***Figure 3.23*** shows Dropbox's navigation bar from their iOS app with the menu open, this follows the style discussed. It was considered a particularly nice feature how the menu is represented in the navigation bar with three dots rather than the common "hamburger" style that is so commonly used. Another admired design aspect of this example was the use of the dark opaque box over the rest of the screen to clearly separate the menu from the other content on screen.

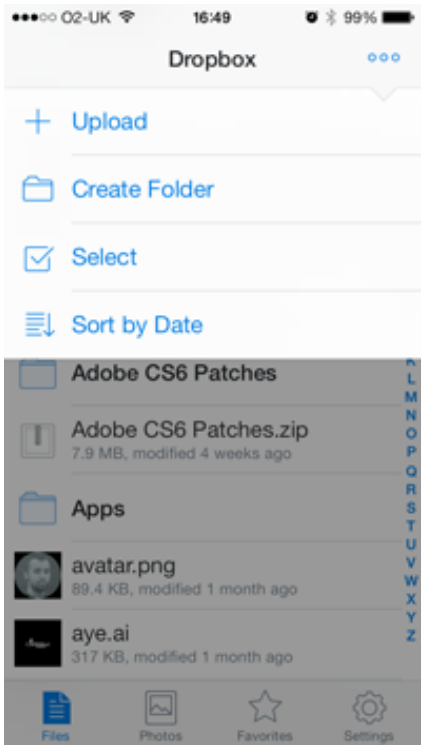


Figure 3.23 Dropbox iOS App Navigation

Shown in ***Figure 3.24*** is the initial concept of the mobile navigation for workulr as it will appear when the page is loaded. ***Figure 3.25*** shows both the primary navigation and secondary user options navigation open. This is achieved by the user clicking on the plus icon to toggle primary navigation open and shut and on their avatar to toggle secondary user options navigation open and shut. The concept for the plus icon to toggle the primary navigation is based on the principal that, like the three dots used

in Dropbox's iOS App navigation, most users will associate the plus icon with providing more options, or to show them more. This icon is then rotated to make an x when the main menu is active, most users will associate an x with a method of closing something. For the user options the desktop implementation was simply modified to have the options stacked instead inline. For usability purposes Apple recommend a minimum target size 44px by 44px for touch devices. This recommendation was taken into consideration and the options for mobile navigation are designed to be 44px tall, while the toggle buttons are both 60px tall **(http://www.uxbooth.com/articles/considerations-for-mobile-design-part-3-behavior/)**.

Figure 3.24 Mobile Navigation Closed

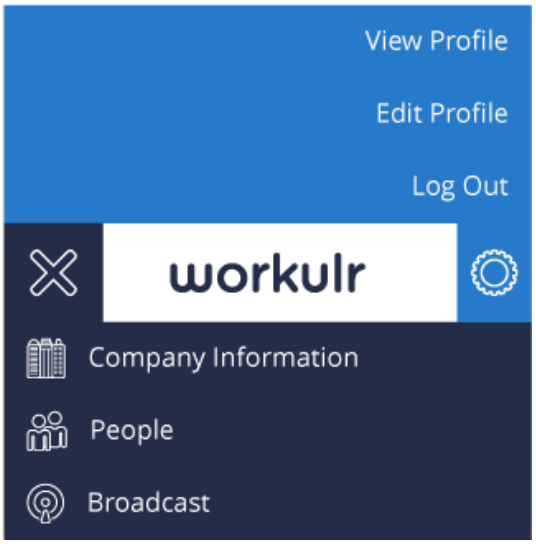


Figure 3.25 Mobile Navigation Open

3.1.4 Revised Navigation

Through comments made during a design review it was pointed out by members of a selected peer group that the icons used for the primary navigation options were not a self explanatory as was initially imagined. Members of the selected peer group also commented that the mobile navigation for the primary navigation options was a better implementation where the icons were accompanied by text explaining what the icon represented. It was stated that a user may or may not guess what the icons represented, and that most users would have to click in to the page to see what it contained. It was also pointed out that some users may not even know that the icons were the navigation.

This called for a rethink for the primary navigation in the application. The final decision was to implement two different types of navigation - 'in-app' and 'pages' navigation. The pages navigation will be similar to the original navigation style and will appear on any static page that is available to view without logging in. The pages navigation will have different options on the right depending on whether or not the user is logged in. Both instances are shown in ***Figure 3.26*** and ***Figure 3.27***.

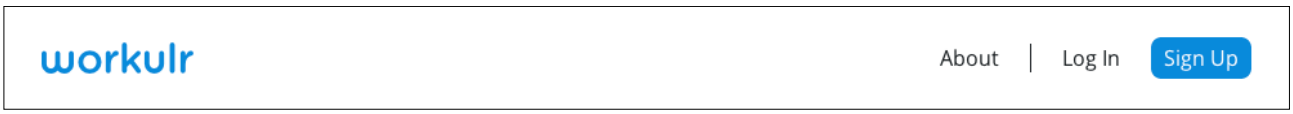


Figure 3.26 Pages Navigation Logged Out

Figure 3.27 Pages Navigation Logged In

The in-app navigation will be shown on application pages only (any page that require a user to log in to view). The in-app navigation is different to the pages navigation and takes the form of a fixed, full height sidebar rather than a bar along the top. One of the main things that this layout allowed for was the addition of the title of each navigation item beside the icon. Since the redesign uses vertical space rather than horizontal space it became possible to fit the secondary user options navigation under the main primary application options. The logged in user is shown their avatar and first name, clicking on this will allow them to view and edit their profile information. The log out option is spaced out further from the other options to prevent the user accidentally clicking it. Shown in ***Figure 3.28*** is the in-app navigation for workulr in three different instances.

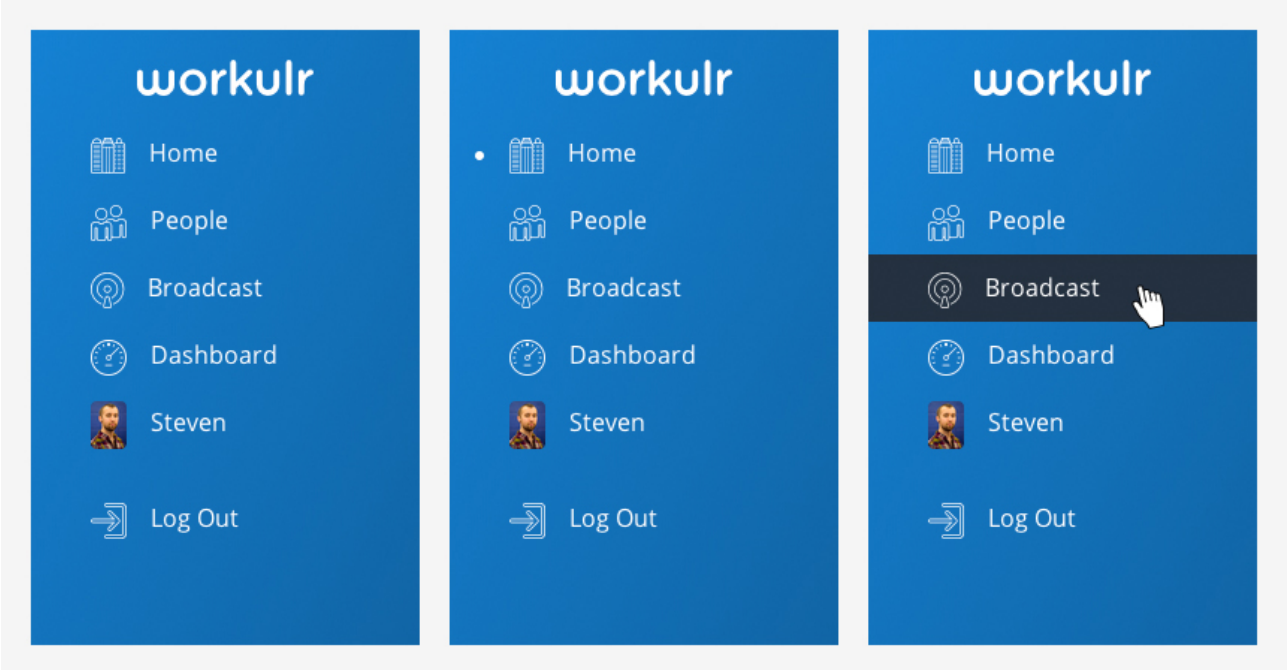


Figure 3.28 In-App Navigation

Reflecting on the three instances shown in ***Figure 3.28*** the first instance shows the general design of the in-app navigation. The second instance shows how the active page is indicated to the user by placing a white dot beside the active page, as good user experience principals dictate that users like to know what page they are on in the website/web application. The third and final instance shows the hover state for the navigation options, again good user experience principals demonstrate that users will

benefit from an indication that something can be clicked, the most common way of implementing this is a design change when an option is hovered over. The dashboard option will only be shown in the in-app navigation if the logged in user is a company administrator.

Two challenges with this new in-app navigation layout were discovered at the implementation stage. The first of theses challenges was how to cater for the layout on mobile devices. The new in-app navigation uses a set horizontal width which, on a mobile device, causes a problem as the navigation would take up the majority of the screen. The solution was a layout change for mobile where the navigation sidebar would disappear and the main content would fill the viewport. When active the navigation would slide out from the left hand side and nudge the content over similar to that seen in ***Figure 3.22***. The second challenge was that when the sidebar disappeared there was no branding and no method for the user to toggle the navigation. A solution was produced through discussions with peers, and design advisers. When the in-app navigation disappears a bar will appear at the top of the screen, containing the workulr icon and an icon to resemble more options. The plus icon in **Figure 3.24** was replaced with the three dots icon used by dropbox in their iOS application. Through further research it was discovered that this icon option was more common than first expected. It appears in iOS 7 in both the iTunes and Music application, Windows 8 Mobile applications also implement a similar icon, and Android applications use three dots to resemble more options only vertically aligned. ***Figure 3.29*** shows the mobile design for the in-app navigation hidden and ***Figure 3.30*** shows the in-app navigation open.

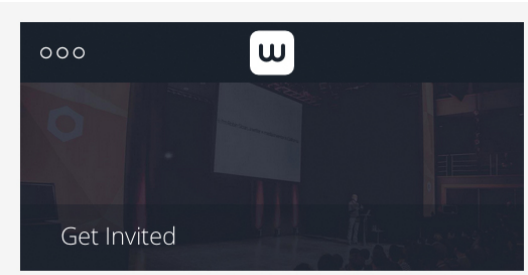


Figure 3.29 Mobile In-App Nav

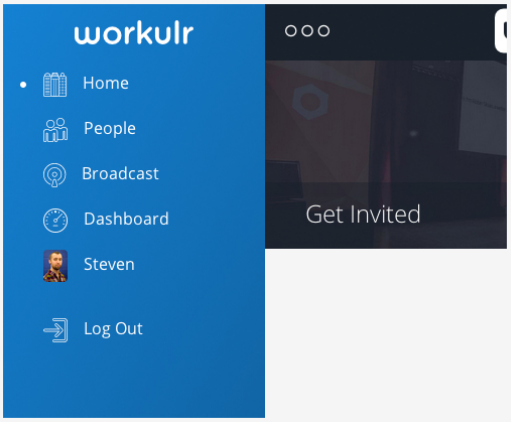


Figure 3.30 Mobile In-App Nav Open

3.1.5 Key Elements

While the style tile covers the general look and feel of the application there are some key elements that will feature in the application. It was seen as necessary to design compositions for these elements before the implementation stage to ensure there was a thorough understanding of what they should look like.

Cover Image

A common trend for account based web applications is to allow the user to add a cover images to their page. This trend can be seen in Twitter, Facebook, Medium and Path for examples. It was felt that allowing the administrators of an account to add a cover image relevant to their company would add a bit more personalisation to the account. The cover image should have a footer that sits over the bottom of the image and contains some basic information. The information would be the company name, the page they are on (if required), the companies description or tagline, the number of people associated with that account, and the number of broadcast items. By displaying this cover image and information to the user of the application at the top of every page it enforces a sense of consistency and reassures them that they are still looking at information from their company's account. ***Figure 3.31*** shows the home page with a cover image where the name of the page is not necessarily required. ***Figure 3.32*** shows the people page with a cover image where it would be useful for the user to be reminded what page they have navigated to.

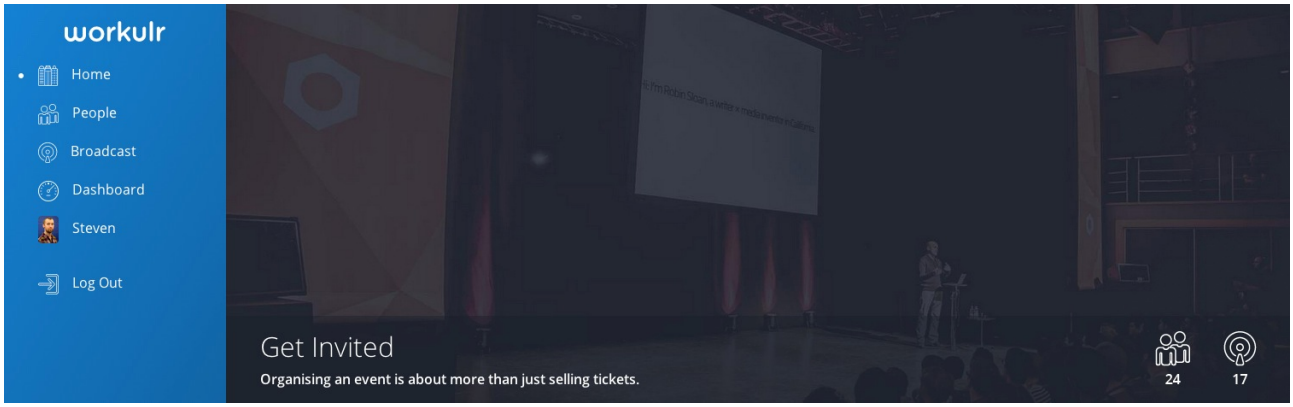


Figure 3.31 Cover Image Without Page

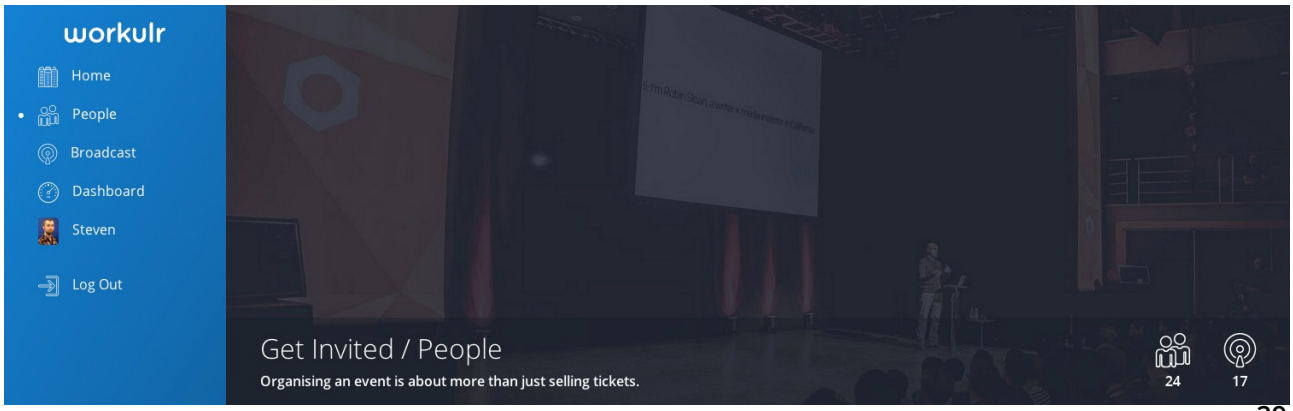


Figure 3.32 Cover Image With Page Name

People Page / Company Page - Person Avatar

One of the key features of workulr is for the users who belong to an account to get to know the people they work with a bit better. The first step in getting to know someone you work with is being able to put a name to a face. The people page will provide the user with a wall of avatars belonging to their colleagues with their names along the bottom of the image. ***Figure 3.33*** shows the design for the person element that will be displayed on the people page. This style of avatar will also be used on the company home page to show the four latest users added to the account.

Figure 3.33 Person Avatar

Forms

The majority of workulr's functionality will involve data input from users. Considering this, the structure and layout of forms will play a vital role in the overall user experience of the application. One of the key elements to a good form design is that the user is fully aware of the information required from them, this is generally achieved by using labels for the input elements. In a study carried out by UX Matters, eye tracking software was used to see how long it took users to pair form labels with their input field. When the labels were left-aligned and positioned to the left of their associated input fields it took an average of 500ms for the user to pair the two up. When the labels were right-aligned and positioned to the left of their associated input fields it took 170ms - 240ms, depending on the users experience of completing online forms. The fastest time of 50ms was found when the labels were left-aligned and positioned above their associated input fields. **(Penzo, M. 2006)**

Taking this into consideration a design was developed that incorporated this research into a valid layout for the data input forms of workulr. Keeping in mind that consistency is important for a good user experience all of the forms should incorporate a similar design. An example of this design can be found in ***Figure 3.34*** which shows the design for the company details page. To enhance the relationship between the label and it's associated input field, both elements are combined them in a box. Also shown in ***Figure 3.34*** is the hover state that will be implemented on these boxes. The example shows a user hovering on the Company Name where the cursor is changed to a pointer to show the user it it clickable, and it is also highlighted to reinforce this. Consideration was given to highlighting the box in a similar fashion to the hover state when the input element was in focus. This idea was presented to a selected peer group who agreed that this was distracting and that the flashing cursor was enough to indicate which field they were in. The forms also make use of placeholders to give instructions or examples for the user's input.

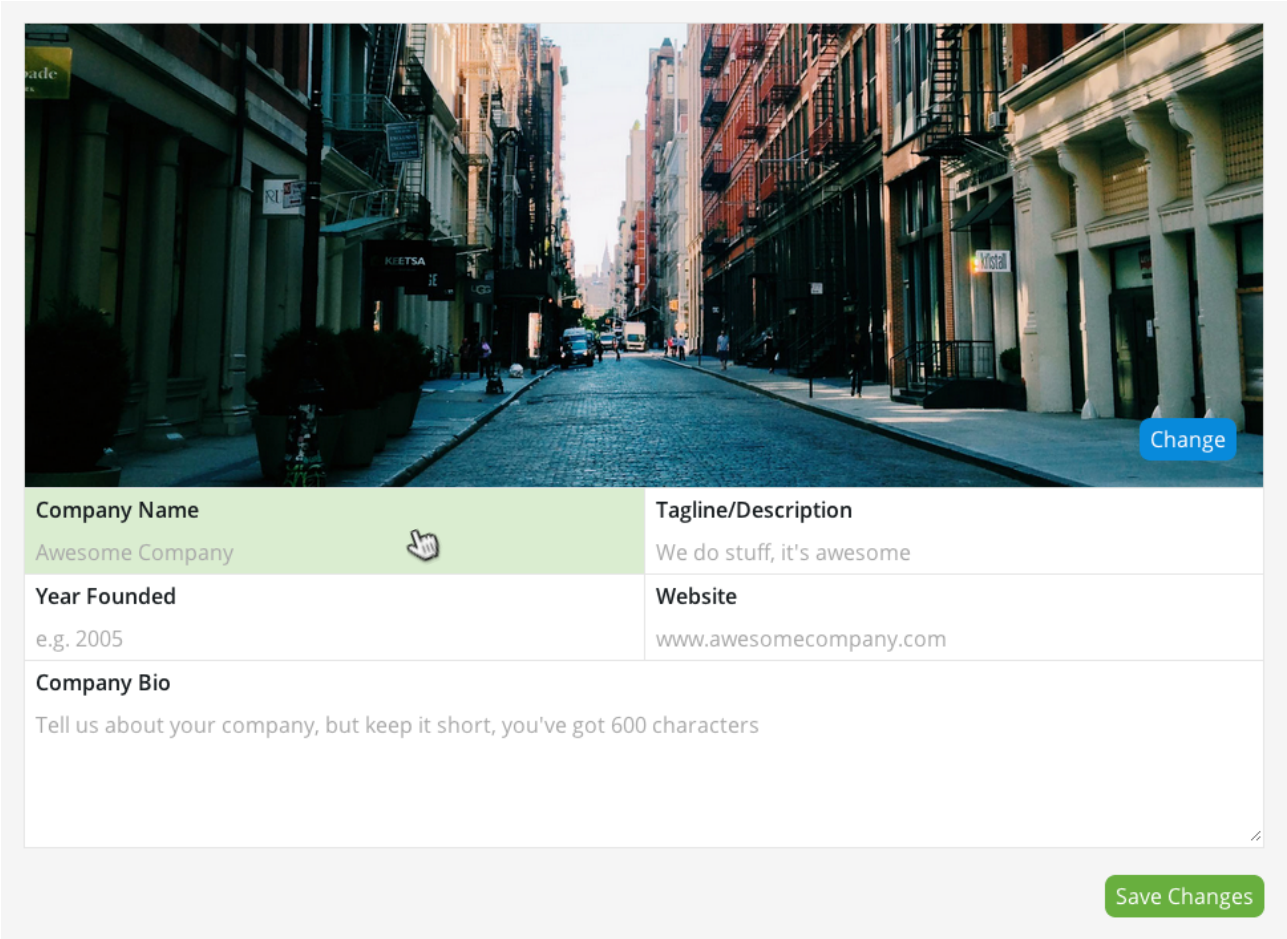


Figure 3.34 Company Information Form Design

Broadcast Page Items

The broadcast page will be the home to news and events related to the company. Originally it had concerted in the paper prototyping stage that the broadcast page should have a timeline style layout with items either side of a central line shown in the One Up solution in ***Appendix F***. When this concept was put into production and in a design revision there were a couple of issues discovered that caused for a rethink of the layout of this page. The main issue was that the items at either side could have been any height which made coding the layout for the front end rather difficult. The only other option was to leave a gap the size of the item at the opposite side of the timeline as seen in iteration three of the Six Up created in the paper prototyping stage, this can be seen in ***Appendix E***. Members of a selected peer group were asked for their views on this, the main opinion was that this layout reminded them of the Facebook timeline which they were not fond of for various reasons. Others in the selected peer group stated that they would prefer a more linear layout where the items were stacked in a row. A solution was developed to cater to these suggestions and can be seen in ***Figure 3.35***. The timeline is still in place but the broadcast items are stacked directly above each other.

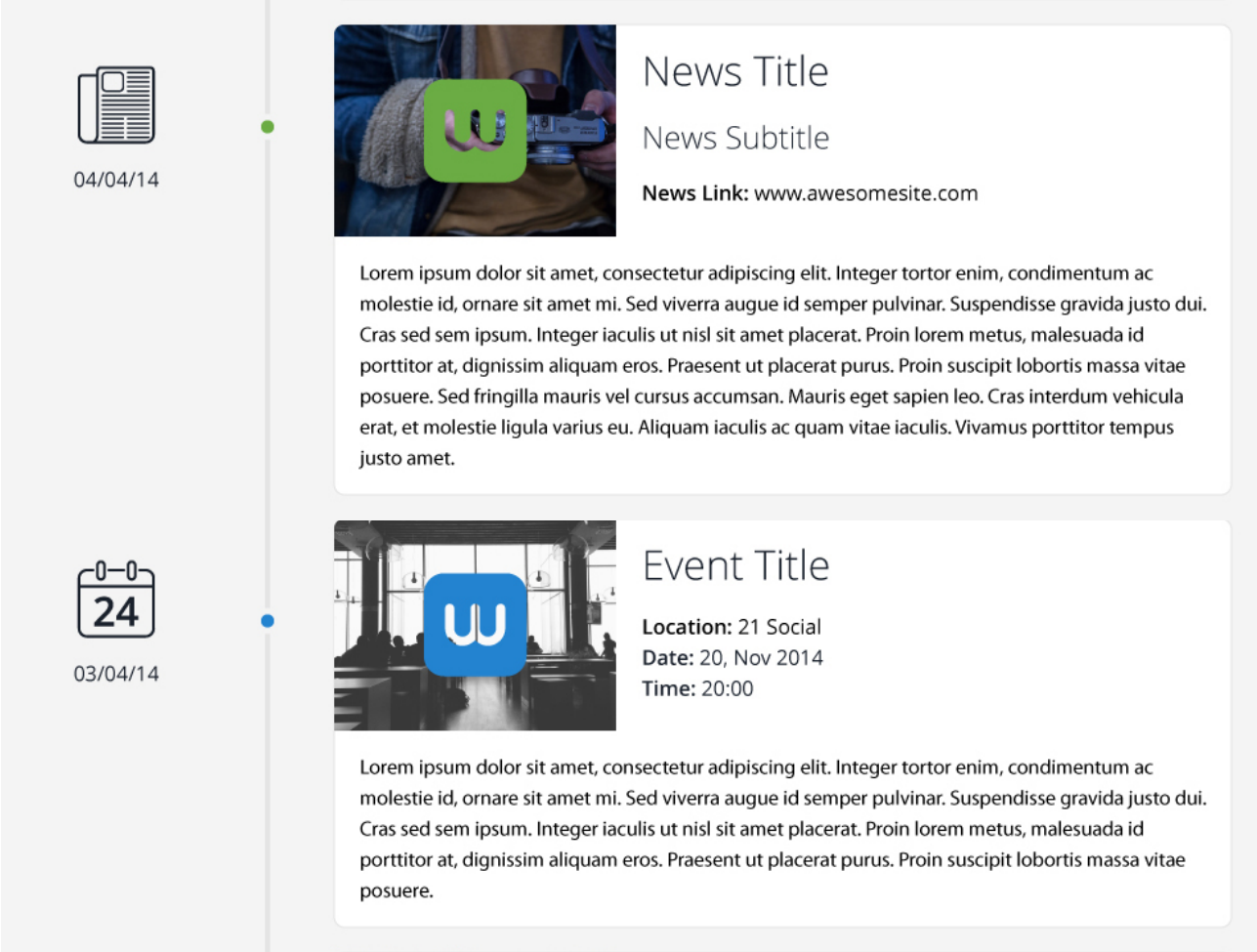


Figure 3.35 Event Broadcast Items and News Broadcast Item

3.1.6 Interactions

There are no interactions for workulr that are particularly unique but they still required thorough consideration as the interactions with the application are vital to the users experience of using the web application. The most obvious of these, and probably the most common are links and buttons. The user needs to know that these elements are actually links or buttons and that they can be clicked. There are two types of links for workulr, standard text links and button links, and as seen in the style tile action buttons are similar in style to button links with a colour change. The most common way of showing a user that a link is clickable is a change in styling for when a link or button is hovered on. The interactions for each of these elements can be seen in ***Figure 3.36.***

Figure 3.36 Link and Button Hover States

There will be situations when using workulr where the user will require feedback from the application such as success/confirmation messages and error messages. A common trend for implementing these type of messages is to use flash messages which are bars that appear at the top of the page. The colour of the bar is relevant to the type of message i.e. green for a success/confirmation, and red for an error. Shown in ***Figure3.37*** are examples of both an error and a success/confirmation flash message.

Figure 3.37 Error and Success/Confirmation Flash Messages

Another very vital interaction for workulr is showing the user that, by clicking on someones avatar on the people page, they can view that persons profile. There were several options to achieve this when a user hovers over a persons avatar. After consulting a selected peer the final solution was decided upon and is shown in ***Figure 3.38***.

Figure 3.38 Person Avatar on Hover

Due to the nature of workulr's content, there may be instances where a location, person, or broadcast item may need to be deleted. Considering usability in this case it is always good to implement a confirmation on a delete action. A common way of presenting this to a user is an alert box or pop up/overlay. Both of these options can be considered intrusive and in most cases unnecessary. An different solution was selected for workulr. When the user clicks on the delete button the colour of the button and the text will change to ask them are they sure, they will then click this button again to confirm the removal of the item. After a two second delay the button will return to it's initial state. This interaction is shown in **Figure 3.39**.

Figure 3.39 Delete Button Interaction

3.1.7 Information

Most web users have the desire, when using a website or application, to save time. They want the information they are looking for to be available as fast as possible. Therefore the information held on workulr has to be analysed to find the best way to present it to the user so that the are getting the information they want to see in the order they expect it, or are able to find it quickly. The first piece of information the user will look for on a page is some sort of page title as an assurance that they have arrived on the right page. ***Figure 3.28*** and ***Figure 3.32*** show examples of how this is to be achieved.

With regard to content there are two main pages that needed to be considered for this, the people page, and the broadcast page. The people page will contain a grid of avatars for the people in that company, with their name on the avatar. The best way to present this information to the user is in alphabetical order sorted by last name ascending, this makes it possible for the user to quickly scan down through the list and find who they are looking for. The second page that needed some consideration was the broadcast page which contains news and event items. The design for news and event elements shown in ***Figure 3.35*** were presented to a selected peer group. It was stated that the icons and coloured indicators were used to reinforce the difference between a news item and an event item, and that the items would be ordered in chronological order to give the user the latest information first.. The general feedback concluded that this proposed method of displaying the information on the broadcast page was clear and understandable.

3.2 System Design

3.2.1 Client Server Model

Figure 3.40 shows a client-server model which gives a high level perspective of the technologies to be used on the client side and the server side for workulr.

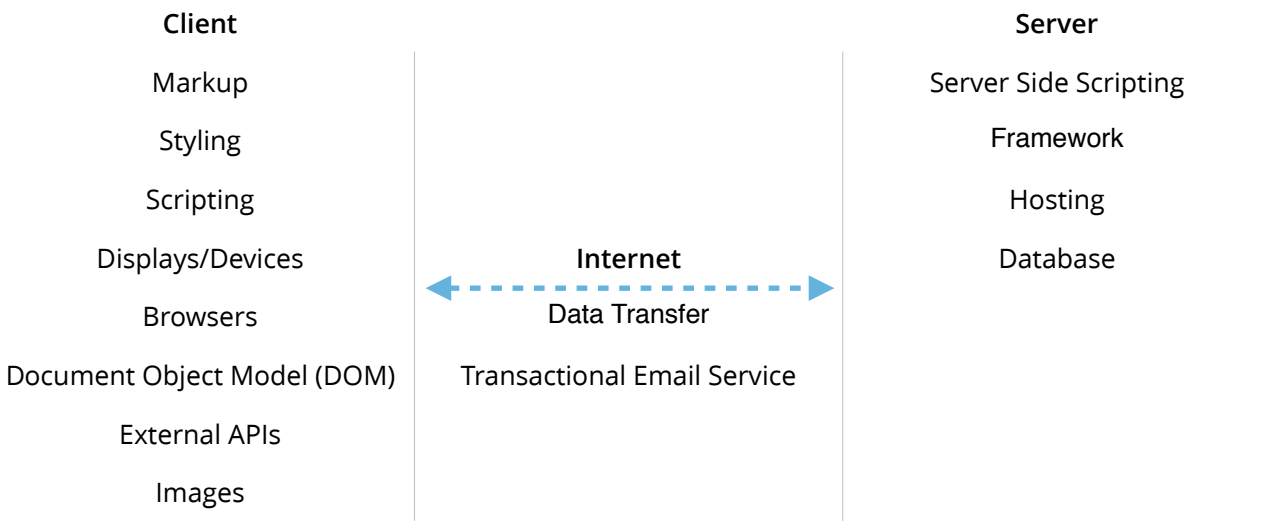


Figure 3.40 Client Sever Model

3.2.2 System Architecture Diagram

The System Architecture diagram in figure 3.41 shows a high level overview of how the major elements of workulr interact with each other. Client requests are handled by the server which processes these and deals with data transfer to and from the database. The server interacts with a transactional email service via an API request to send emails to user.

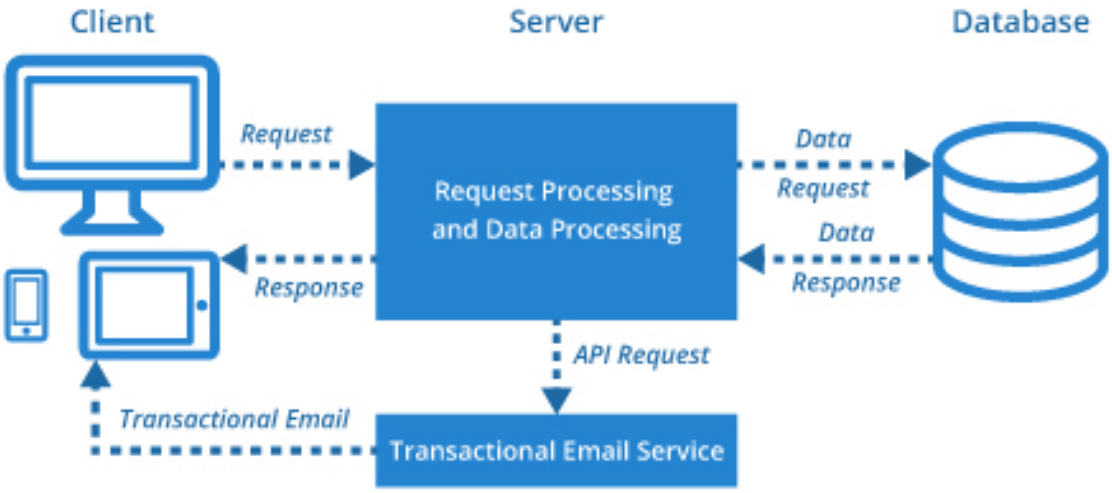


Figure 3.41 System Architecture Diagram

3.2.3 Model-View-Controller

workulr will be built using the Sails.js, a Model-View-Controller(MVC) framework. The Model-View-Controller architecture is designed for the development of interactive applications. ***Figure 3.42*** shows a diagram of the Model-View-Controller framework and how it works with applications.

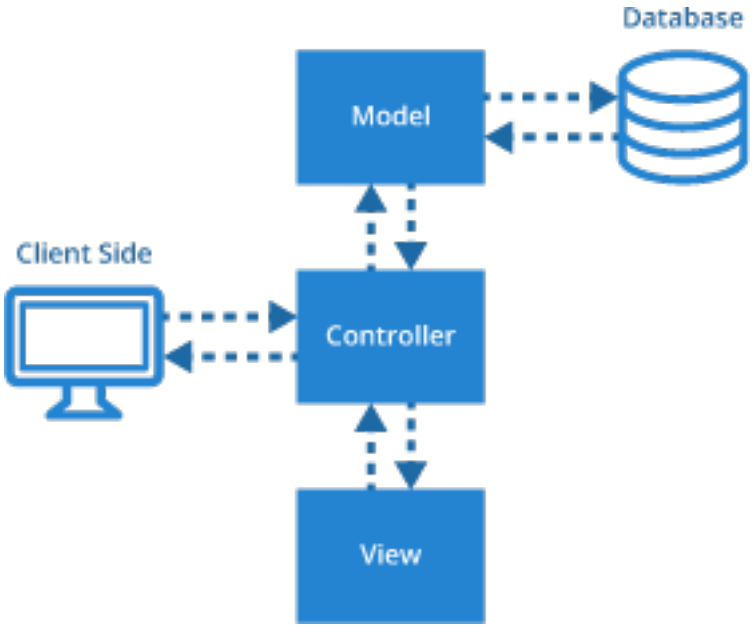


Figure 3.42 Model View Controller

A model is the data and the rules applying to that data, it provides the controller with a data representation of whatever the user requested. This data model will be the same no matter how it is presented to the user. A view is used to present the data received from the model to the user. Views are usually templates populated by the data returned. A controller manages the user requests (like GET and POST) when a user interacts with an element that performs an action. The controller calls the appropriate model for the task and then selects the necessary view **(Pastor, P. 2010)**.

Models

Models are the design of the data structure and any validation rules applied to that data, it provides the controller with a data representation of whatever the user requested. Workulr has five models:

Broadcast - This model will contain the data structure and validation rules for all the broadcast items saved from workulr accounts.

Company - This model will contain the data structure and validation rules for all the companies created by users who set up a workulr account for their company.

Email - This model contains three simple lines which link the Email operator to the Mandrill API adapter, which allows workulr to send transactional emails directly to users inboxes.

Location - This model will contain the data structure and validation rules for any locations added to company accounts by administrative users.

User - This model will contain the data structure and validation rules for all users using the application, whether standard user or administrative user.

The contents of these models is shown in **3.4 Data Design**.

Views

Views provide a way to present data to the user or allow the user to input data. There are quite a number of views that are required for the user interface of workulr. Some are static pages such as the home page of workulr, and some are templates that are dynamically populated by data from the appropriate model(s) through a controller, such as the people page which will show avatars and names for all the people in the company.

Controllers

Controllers manage the user requests like GET and POST when a user interacts with a Graphical User Interface(GUI) element that performs an action. The controller calls the appropriate model for the task and then selects the necessary view. There are seven controllers for workulr:

AuthController - This controller handles the logging in and logging out of users using the application.

BroadcastController - This controller handles the creation, editing, and deletion of broadcast items.

CompanyController - This controller handles the creation, editing, and deletion of companies.

DashboardController - This controller handles the rendering of the data for the dashboard home screen.

LocationController - This controller handles the creation, editing, and deletion of company locations.

RoutesController - This controller has one main purpose - it check to see if there is a logged in user, and if there is, it changes the index route of workulr to render the company home page (or company information page as it has been referred to previously) instead of the landing page. If there is not a logged in user it renders the static landing page.

UserController - This controller handles the creation, editing, and deletion of workulr users, from initial users setting up the company to people added to accounts by admins.

3.3 Logic Design

The design of the logic for the functionality of workulr was vital as this made sure the operation of the application was fully understood before it was implemented. ***Figure 3.43*** shows a use case diagram showing how a user interacts with workulr. The numbers on the use case digram show points in the system where there is functionality that requires logic. This section covers the annotation of those numbers.

1. **Signs up using sign up form**

   The data submitted is validated where validation is required. The email address submitted is searched for in the user model to ensure that it does not already exists. The password supplied is encrypted and a confirmation link is generated before the submitted data is saved in the user model. An email is sent to the user containing the confirmation URL.

2. **Confirms email using confirmation URL**

   The confirmation link parameter of the URL is searched for in the user model to ensure it exists, and if it does, the corresponding user's account is set to active, and the user is directed to the login page.

3. **User logs in**

   The email address and password submitted are checked to see if that combination exists in the user model. A check is also ran to ensure the user's account is activated. If the user passes this authentication a session is created storing the user's first name, second name, ID, company ID, role, avatar file name and an authenticated attribute set to true. The items stored within the session are then used in the logic to provide functionality for other parts of the application.

4. **Directed to company home page**

   Ensures user is logged in. If the user is logged and has a company ID the viewable company information is retrieved from the company model where the ID is equal to the user's company ID. The latest three locations are retrieved from the location model where their company ID is equal to the user's company ID, the latest four people added to the company account are retrieved from the user model where their company ID is equal to the user's company ID, and the latest four broadcast items are returned from the broadcast model where their company ID is equal to the user's company ID.

5. **Navigates to people page**

   Ensures user is logged in. If the user is logged in all the available users that have active accounts are returned from the user model where their company ID is equal to the logged in user's company ID.

6. **Navigates to persons profile**

   Ensures user is logged in. If the user is logged in check to ensure that the company ID of the requested user is equal to the logged in user's company ID. If this authentication is passed user is permitted to view the requested users profile where the viewable information will be retrieved from the user model where the ID is equal to the requested user's ID. If authentication fails the user is redirected to a 403 Forbidden page. If the requested user cannot be found the user is redirected to a 404 Not Found page.

7. **Navigates to broadcast page**

   Ensures user is logged in. If the user is logged in all available broadcast items are returned from the broadcast model where their company ID is equal to the logged in user's company ID.

8. **Navigates to own profile**

   Ensures user is logged in. If the user is logged in their ID is used to retrieve all information that the user is permitted to edit from the user model where the ID is equal to the user's ID.

9. Edits and saves own profile information

Validate any data input that requires validation and update the user in the user model where the ID is equal to the logged in user's ID.

10. Navigates to dashboard

Ensures user is logged in. If the user is logged in and has the role of "co-admin" the dashboard is loaded. If the user's role is not "co-admin" the user is shown a 403 Forbidden page. If the dashboard is loaded the latest latest four locations are retrieved from the location model where their company ID is equal to the logged in user's company ID, the latest four people added to the company account are retrieved from the user model where their company ID is equal to the logged in user's company ID, and the latest four broadcast items are returned from the broadcast model where their company ID is equal to the logged in user's company ID.

11. Add/edit company information

Ensures user is logged in. If the user is logged in and has the role of "co-admin" check to see if the user has a company ID, if they do not a company does not exist. If a company does not exist a new company is created in the company model and the user's company ID is set to the newly created company's ID. The user is then redirected to a page where they can edit the information for the newly created company. If the user does have a company ID all the information that the user is permitted to edit is retrieved from the company model where the ID is equal to the user's company ID.

12. Add/edit/remove company locations

For any of the functionality related to the company locations the user will have to be logged in and have the role of "co-admin". This functionality also requires the logged in user to have a company ID to ensure that they have a company to add locations to.

Add - Validate any data input that requires validation, if validation passes create the new location in the location model. The new location's company ID is set to the same company ID as the logged in user's.

Edit - Check that the requested location has the same company ID as the logged in user. If this authentication is passed the information that the user is permitted to edit is retrieved from the location model where the ID is equal to the requested location ID. If this authentication fails the user is redirected to a 403 Forbidden page. If the requested location cannot be found the user is redirected to a 404 Not Found page.

Remove - Check that the targeted location has the same company ID as the logged in user. If this authentication is passed the targeted location is removed from the location model where the ID is equal to the targeted locations ID. If this authentication fails the user is redirected to a 403 Forbidden page.

13. Add/edit/remove people

For any of the functionality related to people the user will have to be logged in and have the role of “co-admin”. This functionality also requires the user to have a company ID to ensure that they have a company to add people to.

Add - Validate any data input that requires validation. The main validation is to check that the email address provided does not already exist. If validation is passed a random password is generated and encrypted, and a confirmation link is generated before creating the new user in the user model. The newly created user's company ID is set to the same company ID as the logged in user's. An invitation email is sent to the email address submitted containing the confirmation link and the unencrypted generated password.

Edit - Check that the requested user has the same company ID as the logged in user. If this authentication is passed the information that the user is permitted to edit is retrieved from the user model where the ID is equal to the requested user's ID. If authentication fails the user is redirected to a 403 Forbidden page. If the requested user cannot be found the user is redirected to a 404 Not Found page.

Remove - Check that the targeted user has the same company ID as the logged in user and check that the targeted user is not the logged in user. If this authentication is passed the targeted user is removed from the user model where the ID is equal to the targeted user's ID. If this authentication fails the user is redirected to a 403 Forbidden page.

14. Add/edit/remove broadcast items

For any of the functionality related to broadcast items the user will have to be logged in and have the role of “co-admin”. This functionality also requires the user to have a company ID to ensure that they have a company to add broadcast items to.

Add - Validate any data input that requires validation, if validation passes create the new broadcast item in the broadcast model. The newly created broadcast item’s company ID is set to the same company ID as the logged in user’s.

Edit - Check that the requested broadcast item has the same company ID as the logged in user. If this authentication is passed the information that the user is permitted to edit is retrieved from the broadcast model where the ID is equal to the requested location's ID. If authentication fails the user is redirected to a 403 Forbidden page. If the requested user cannot be found the user is redirected to a 404 Not Found page.

Remove - Check that the targeted broadcast item has the same company ID as the logged in user. If this authentication is passed the targeted broadcast item is removed from the broadcast model where the ID is equal to the targeted broadcast item's ID. If authentication fails the user is redirected to a 403 Forbidden page.

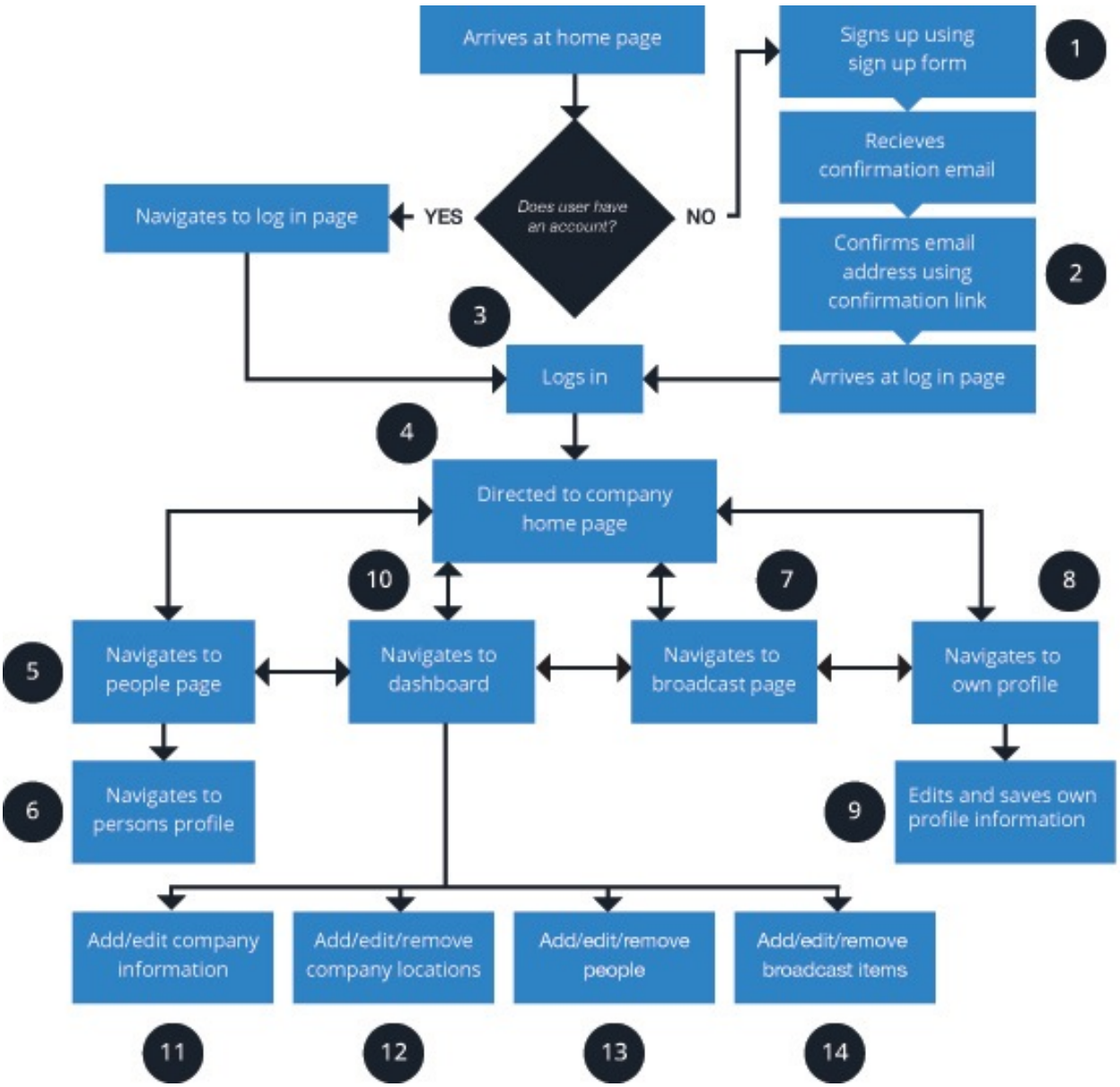


Figure 3.43 Use Case and Logic Diagram

3.4 Data Design

After the logic was clarified the next step was to establish the type of the data the application will store and how it will be stored. This page shows the data design for workulr.

Company Model		
Entity	**Type**	**Required**
_id	ObjectId	TRUE
bio	String	FALSE
cover_image_url	String	FALSE
created_by	String	TRUE
name	String	FALSE
tagline	String	TRUE
website	String	FALSE
founded	Integer	FALSE

User Model		
Entity	**Type**	**Required**
_id	ObjectId	TRUE
avatar_url	String	FALSE
bio	String	FALSE
birthday	Date	FALSE
co_id	String	TRUE
confirmation_link	String	TRUE
confirmed	Boolean	TRUE
email	String	FALSE
fname	String	TRUE
hometown	String	FALSE
interests	Array	FALSE
living	String	FALSE
password	String	TRUE
position	String	FALSE
role	String	TRUE
sname	String	FALSE
tele	String	FALSE

Location Model		
Entity	**Type**	**Required**
_id	ObjectId	TRUE
add_line_one	String	FALSE
add_line_two	String	FALSE
area	String	FALSE
city	String	FALSE
country	String	FALSE
email	String	FALSE
name	String	FALSE
postcode	String	FALSE
tele	String	FALSE

Broadcast Model		
Entity	**Type**	**Required**
_id	ObjectId	TRUE
co_id	String	TRUE
copy	String	FALSE
created_by	String	TRUE
event_date	Date	FALSE
event_location	String	FALSE
event_time	Date	FALSE
image	String	FALSE
item_type	String	TRUE
news_link	String	FALSE
title	String	TRUE
subtitle	String	FALSE

4. Implementation

4.1 Technologies and Tools Selection

4.1.1 Server Side Technologies

Server Side Scripting

The two server side scripting languages considered for workulr were PHP, and Node.js. PHP is one of the most popular scripting languages mainly focused on server-side scripting. PHP is described as “a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML” **(PHP. 2014)**. Node.js is a relatively new language and is described as “a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices” **(node.js. 2014)**. Node.js was selected because it allowed for writing server side scripting in JavaScript which would lead to improved JavaScript skills.

Framework

It was decided at an early stage that the project would benefit greatly from the use of a Model-View-Controller framework as there will be many different functions, views, and user interactions. The Model-View-Controller frameworks considered, which can be used with Node.js, were Express.js, and Sails.js. Express.js is not a complete MVC/MV* solution, but it is ideal for Node.js beginners. It offers very basic tools such as the ability to run a web server and routes. Express.js is not opinionated framework meaning that it doesn’t enforce the user to do things its way **(Express. 2014)**. Sails.js provides a full MVC solution. Sails.js has generate tools for creating models, controllers and API endpoints **(Balderdash, 2014)**. Sails.js uses object-relational mapping and allows the application to use any kind of database. Unlike express Sails.js is an opinionated framework and uses popular conventions. Sails.js was selected, as it offers a full MVC framework, and it allows for faster production time than Express.js. The other very attractive feature is the built in support for realtime, which could be useful for realtime notifications and update of content without a user needing to refresh the page.

Hosting

When beginning work with Node.js it is very easy to get an app up and running on the local machine. When considering hosting services it seemed that most shared hosting packages did not support Node.js. It seemed therefore that the project would either require the purchase of a Virtual Private Server(VPS) configured to host Node.js projects, or the use a hosting service which provide

environments with Node.js support like Heroku **(Heroku. 2014)** or Nodejitsu **(Nodejitsu Inc. 2014)**. Digital Ocean provide VPS for very affordable prices and provide tutorials on how to set a server up to host whatever the user requires **(DigitalOcean. 2014)**. Another option which was considered was Stackful.io which is built on top of Digital Ocean VPS and provides a preconfigured stack which is ready to deploy and run apps **(Stackful.io. 2014)**. Heroku, Nodejitsu and Stackful.io allow the use of Git as a deployment mechanism. This functionality can also be used with a Digital Ocean VPS if installed. Nodejitsu does provide the additional option of the jitsu tool - instead the user simply runs jitsu login then jitsu deploy to deploy your app.

Neither Heroku or Nodejitsu provide support for MySQL databases so this would require hosting the database in another location should this be the selected database. As Digital Ocean allows the installation of MySQL and any other dependencies that the application may require, it was considered as the best option for the project.

Database

The options considered for the database were MySQL **(MySQL. 2014/)**, and MongoDB **(MongoDB. 2014)**. There was a requirement for a large amount of data to be stored in the database and the data would need to be retrieved quickly from the database to satisfy users. The database would need to have multiple models and would require relations between the models to prevent redundant data. MySQL is very popular and is probably the database which most developers have experienced. The other option for the database was MongoDB, a NoSQL database. NoSQL databases are non relational. In comparison to MySQL databases they are more scalable and can provide superior performance when designed correctly. The problem is that mongoDB models are not designed to be related, which is required to get information from one or more models.

Either MySQL or MongoDB would provide a suitable platform for the project to store the applications data. At the start of the project testing and evaluation of these technologies was carried out through prototyping. The initial choice was MySQL as it provides an easy way to relate models to normalise data and prevent data redundancy.

The biggest change in the system design of workulr during development was the change from MySQL to MongoDB. Initially it was decided to use MySQL as it had the ability to relate data between multiple tables to prevent data redundancy. In the initial system design report it was reported that Sails.js would be used, which has a feature called Waterline ORM **(waterline. 2014)**. An ORM(Object-Relational-

Mapper) is a method for designing and querying database models at the theoretical level, where the models is described in terms that can be easily understood by non-technical users.

"Waterline is a brand new kind of storage and retrieval engine. It provides a uniform API for accessing stuff from different kinds of databases, protocols, and 3rd party APIs. That means you write the same code to get users, whether they live in MySQL, LDAP, MongoDB, or Facebook."

After developing some of the features of workulr using the Sails-MySQL adapter for waterline ORM **(sails-mysql. 2014)** it became apparent that MySQL was not being used for it's main technical advantage, i.e. relational tables. The retrieval of data was being pulled back from the database, using information stored in the User Session when the user logged in, to send multiple queries to the different models in the database. MongoDB was considered for it's scalability and superior performance compared to MySQL. MongoDB databases contain collections which contain documents. An example of this would be that workulr is the name of the database, users would be the name of a collection and in that collection there would be multiple documents holding information on users. The main difference in MongoDB documents compared to MySQL rows is that each document in a collection can have a different number of fields, different content, and different size, which make it much more flexible and does not waste space for empty fields as MySQL does **(Tutorialspoint.com. 2014)**. This particular feature allows for storing both News and Event Broadcast items in one collection instead of two separate collections. Moving to MongoDB did not require extensive effort - it was simply a case of installing the Sails-Mongo adapter for Waterline ORM **(sails-mongo. 2014),** updating the adapters.js file to use MongoDB as the default database, and setting the details for the local MongoDB installation.

Node Modules

Modules (or packages) provide a way to include other external Javascript libraries into a Node.js application **(Mixu's Node Book. 2014)**. These module or packages are installed using NPM (Node Package Manager) via the terminal. For example to install bcrypt the command "npm install bcrypt —save" is ran in the root directory of the project in iTerm. This command downloads the contents of this package and saves it as a folder titled "bcrypt" in the "node_modules" folder. The "—save" part saves the version downloaded to the dependencies list in the projects package.json file, this is important as the "node_modules" folder is not pushed up to the project Git repository, so if the contents of the repository is pulled down to another machine and the command "npm install" is ran in the root directory of the project in terminal, it will download all the dependencies listed in package.json. Listed on the next page are the modules used for workulr and what they are being used for:

bcrypt - This module is used to encrypt the users password before it is stored in the database to provide a level of security as the password is not stored in its normal format. This module is also used to encrypt the users email address to make a confirmation code that is emailed to the user so they can confirm their account **(bcrypt. 2014)**.

gm (GraphicsMagick) - This module is used to perform the cropping and resizing of images that users upload to workulr before saving them in the database. Used in three places; company cover image, user avatars, and broadcast item images **(gm. 2014)**.

grunt - This module is a javascript task runner, it provides automation for performing repetitive tasks like minification, compilation, unit testing, etc **(grunt. 2014)**.

jade - This module is the Node.js template engine which was selected for the project **(jade. 2014)**.

moment - This module is used to properly format the date and time retrieved from the database before it is sent to the view **(moment. 2014)**.

node-uuid - This module is used to provide a unique time based id to name images uploaded to the server to prevent multiple users uploading an image with the same name **(node-uuid. 2014)**.

passport - This module handles the login and logout functionality of the application **(passport. 2014)**.

passport-local - This module is an extension of the Passport module. Passport-local is required to use your own database and log a user in using a stored email and password combination **(passport-local. 2014)**.

password-generator - This module is used when an administrative user adds someone to their company account to provide a secure generated password **(password-generator. 2014)**.

sails-mandrill - This module is used to provide methods of using the Mandrill API functionality **(GitHub | sails-mandrill. 2014)**.

sails-mongo - This module is the adapter for Sails.js Waterline which was discussed in the **Database** section of this chapter **(sails-mongo. 2014)**.

validator - This module provides a way to validate data passed to the server before carrying out certain actions

4.1.2 Data Transport Method

Socket.io and AJAX

A data transport method is how web applications send and retrieve data from the server. The two data transport methods considered for use in in workulr were AJAX (Asynchronous JavaScript and XML) and Socket.io. Ajax allows web applications to send and retrieve data from a server asynchronously without requiring the page to refresh. If a standard form transport method was used, every piece of

functionality that involved the sending or retrieving of data from the server would require a page refresh which does not provide a good user experience. An example would be that there is an error in data submitted from a form and it is required to send this error back to the user, if the page is totally refreshed all of the other content that was correct now needs re-entered **(Wikipedia | Ajax. 2014)**.

Socket.io provides a similar service to AJAX with the addition of realtime. Socket.io allows the sending and retrieving of data from the server in realtime. This feature was attractive as it would provide a method of sending notifications to the user without them needing to refresh the page. An example of this, which is hoped to be implemented time permitting, is notifying the user if a person is added to the account they are associated with, or a new broadcast item is added. Socket.io comes packaged with Sails.js so there is no need to install anything on the server side, and the client side library is downloaded to the "js" folder when a new Sails.js application is created. There was an initial concern when considering Socket.io as WebSockets are not supported in all of the browsers workulr aims to support. Through further research it was realised that Socket.io selects the most capable transport at runtime from the options of WebSocket, Adobe® Flash® Socket, AJAX long polling, AJAX multipart streaming, Forever Iframe, and JSONP Polling which enables it to work in all browsers without it affecting the API **(http://socket.io/)**.

During the initial implementation phase it was discovered that Socket.io did not support the transfer of files. AJAX by itself does not support this functionality either, however the work around solution required the use of AJAX. It was decided that a combination of AJAX and Socket.io would be implemented for the data transport method in the application.

4.1.3 Third Party Sever Technology

Transactional Email Service

A transactional email service was required to send out emails to users when they sign up for a workulr account with a confirmation link. Similarly when a company administrator adds a person to their account the person needs to receive an email with an initial generated password and a confirmation link. The main issue when sending transactional emails is that your email may be sent directly to the spam folder of the target email. There a quite a few services who make it easy to send transactional emails and handle the SMTP protocols on your behalf. The main restrictions when choosing a Transactional Email Service was the cost and how easy it was to implement in a node project. Listed on the next page are the prices for three different services.

Postmark - $1.50 per thousand emails and you get 10,000 credits when you sign up **(Wildbit, L. 2014)**.

Mandrill - First 12,000 emails per month are always free and $0.20 per thousand for the next million emails **(Mandrill. 2014)**.

SendGrid - $9.95 per month, 40,000 email credits per month **(SendGrid. 2014)**.

It is clear to see that Mandrill lead the way in value for money. Fortunately there was a module along with clear documentation which makes it easy to implement Mandrill with a Sails.js application **(GitHub | sails-mandrill. 2014)**.

4.1.4 Client Side Technologies

Markup

When it comes to the markup for a web app, HTML is the only real option. The only choice to be considered is which version to use and what tools are to be used used to write it. The chosen version for the markup of this project was HTML5 as this allows the use the newly introduced tags which indicates the content contained in each section and make markup easier to read. The markup was written using Jade, a server side templating engine for Node.js. The Jade template files are compiled to HTML files with the data from the server side of the application before being rendered to the client side of the application. Like any templating engine the main benefit to using Jade is that it allows the developer to code using the DRY (Don't Repeat Yourself) method **(Jade - Template Engine. 2014)**.

Styling

Like markup there is only one choice for styling a web app, and that is CSS. Again there are multiple version of CSS, the version selected for workulr was CSS3, which is the most up-to-date version. CSS3 allows the use of a lot of newly introduced styling techniques such as border-radius, transitions, gradients, which gives more options to tailor the user experience of the user interface for workulr. The CSS was written in SCSS, a version of Sass (Syntactically Awesome Style Sheets) **(Sass. 2014)** which is then compiled to CSS so that it can be applied to the markup on the front end as SCSS files are not currently supported. SCSS, like Jade, promotes the DRY programming method using variables and mixins to write snippets of code that can be used anywhere in the SCSS file. There is also an option to use nesting for specificity where the style for one element is written within the styling for its parent element, this is then rendered to a very specific style rule. The SCSS is compressed when rendered to a the CSS files to increase load speed.

Scripting

There are a few possible client side scripting options, these include JavaScript, ActionScript, Dart, and VBScript. The most widely used client-side scripting language is JavaScript, and was the one selected for the project. JavaScript is an object-oriented scripting language, it runs in the browser and allows the manipulation of elements on a web page. It does so through the Document Object Model (DOM), an API for interacting with a web page.

Scripting Libraries

There are a wide range of libraries which provide programmers with ready built functionality and allow the programmer to write less and do more. The libraries selected for use with workulr are as follow:

jQuery - "jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers." jQuery is not only one of the scripting libraries used it is a dependency of all the other libraries used for workulr. One of the main benefits of using jQuery is how it normalises a lot of browser oddities meaning that the code written will work cross-browser **(jQuery. 2014)**.

jQuery File Upload - This library was developed by Sebastian Tschan and it provides a method of uploading a form with an image through AJAX **(Tschan, S. 2014)**.

jQuery UI Widget - This library was developed by the makers of jQuery and is a dependency of the jQuery File Upload library **(Widget Factory | jQuery UI. 2014)**.

jQuery iFrame Transport - This library, developed by Christopher Lenz, is a dependency of the jQuery File Upload library. It is used to implement an iFrame transport so as to provide $.ajax() calls with the ability to support the uploading of files **(jquery.iframe-transport.js. 2014)**.

PickADate - This lightweight library developed by Amsul provides an easily implemented date picker which is mobile friendly and responsive **(pickadate.js. 2014)**.

Displays and Devices

As a web app workulr will be accessed through web browsers, but there are lots of different devices with different display sizes that have web browsers, and are therefore capable of viewing workulr. The main devices considered for workulr will be desktops/laptops, but with tablets and mobile devices likely to be almost as popular. As workulr is not being developed as a native apps for iOS or Android devices the user interface of workulr will be responsive so that tablet and mobile device users can experience the site and use it easily. The layout will change at the most common device width breakpoints for mobile, tablet and desktop/laptop, everything between these breakpoints will be fluid.

Browsers

When it comes to selecting the browsers to support for a project it is best to research the most popular browsers. Research shows that Chrome is the most popular, followed by IE, then Firefox, and then Safari **(StatCounter Global Stats. 2014)**. Each of these browsers will have multiple versions but it is safe to say that with Chrome and Firefox users will be using the latest version due to automatic updates. From what has been published about Safari it updates along with software updates which are usually quite frequent. The case for Internet Explorer(IE) is a little different. The latest version is IE11 however this version, along with IE10 is only supported on Windows 8 and Windows 7. Windows Vista cannot update from IE9, and Windows XP cannot update from IE8. In the requirements it is stated that workulr would be compatible with a wide range of browsers including the latest release of IE and the previous two releases which would be IE9-IE11. As IE is still the most common browser in business/corporate systems, and such users are less likely than domestic/small business users to update their browsers, it is important that IE works well in IE, since the app is aimed at the business community. Workulr will also need to be supported on, at the least, the native iOS and Android browsers.

External APIs

These will be any APIs(Application Programming Interfaces) that are hosted on a reliable Content Delivery Network (CDN) such as Google Developers **(Google Developers. 2014)**. The only one likely to be required is jQuery. This will be pulled from the Google CDN where it will be retrieved quicker due to cross-site caching. There will need to be a fallback to a self hosted version of any external APIs in case the CDN suffers a failure or the API cannot be retrieved.

4.1.5 Tools

Source Code Editor

The most vital tool that contributes to a good development environment is the source code editor. There are any number of source code editors available that are capable of editing the source code for workulr. It was important to select a source code editor that would provide clear syntax highlight for the different coding languages to be used for the files in this project such as Jade, SCSS, and Javascript. Sublime Text Two was selected for this project as it provides the user with the ability to install packages which enable syntax highlighting for nearly all coding languages. Another useful feature of Sublime Text Two is the ability to have split screen mode. This feature enables the developer to have a markup file at one side and a styling file at the other. **(Sublime Text. 2014)**

Time Management

Due to the scale of the project and the strict deadlines to be met it was decided that a project management tool was used on top of the Gantt Chart time line of the project. The tool was required to keep track of outstanding tasks and a time estimation for completing each of these task. A tool called Teamwork was suggested for doing this. It was stated that Teamwork had the ability to take all of the times specified for each task and give an overall estimation for the completion of the project. The other benefit of using Teamwork was that it was free of charge for 2 projects and 10mb of storage. Teamwork was utilised to make a list of outstanding tasks along with their estimated timing to manage tasks and get an idea of how long it wold take to complete the project. **(Teamwork.com. 2014)**

Backup System

Due to the scale of this project and the strict deadline for the completion of the work the worst case scenario was that a fatal error would occur to a development machine and the work would be lost. The obvious solution to prevent this was to implement a backup system to constantly keep a backup of the work. The backup system that was selected was an Apple AirPort Time Capsule. This device carries out a backup of the development machines wirelessly every hour on the hour when they are within range. This meant that the developer did not need to remember to manually take a backup of the machines.

Version Control

Version control, also known as source control, provides a method to keep track of changes to files stored in what are known as repositories. By doing this developers have the ability to roll back to older versions of files. If something happens and the code that used to work no longer does you can go back to an earlier version and work forward or simply compare what has changed to find the cause. This also provides another form of backup of project files incase something fatal happens to any of the development machines. The most common type of version control is Git, there are services built on top of Git such as GitHub and BitBucket which provide users with an online space to store their repositories. BitBucket was chosen for this project as it provides the user with free private repositories where GitHub only provides users with free public repositories.

Command Line

Many of the selected tools and technologies require a command line interface to run them. Examples of these tools and technologies would be running Sails.js, running MongoDB, compiling icon fonts using Fontcustom, compiling SCSS files to regular CSS, and Git commands to send data to and receive data from BitBucket. Most computers come with a command line tool pre installed, for example Apple comes with a tool called Terminal. There is an application called iTerm which provides extended

functionality over the Terminal tool which enhances developers workflow such as split pane windows and autocomplete, for this reason iTerm was selected for the command line tool for the project. **(iTerm2. 2014)**

MongoDB Graphical User Interface

The main way to view the contents of a MongoDB database is through command line. As there was a need to be able to constantly check for changes to, and modify the data it was necessary to use an application that provided a graphical user interface so as the contents of workulr's database could be displayed in a easily accessible manner. The most commonly used tool used to provide this functionality is MongoHub which was used to examine the actual data contained within collections in the MongoDB database. **(MongoHub. 2014)**

Graphics Production Software

There was a need for graphics production software to create various assets for workulr. The most commonly used software for asset creation and therefore the obvious choice for this project were Adobe Illustrator and Adobe Photoshop.

Hardware

The development of workulr was carried out on an Apple iMac and an Apple MacBook Pro running OSX Mavericks. These machines provided a stable development environment capable of running all of the selected tools and technologies.

4.2 Technologies and Tools Use

All of the required development tools and technologies were installed and ran on both the iMac and the MacBook Pro. This meant that all of the design, development, and testing could be carried out in a single working environment with the added advantage of having that environment replicated on a second machine. The iMac was used as the main development machine because of it's large screen, with the MacBook Pro having the advantage of portability allowing for work to be carried in University or anywhere away the desktop. Duplication of the system effectively provided another form of backup for the project. Some of the server side technologies required the use of command line instruction for configuration and control. This required a understanding of their command line functionality and the use of iTerm as a command line interface. Some of the scripting libraries mentioned previously took considerable effort to integrate in the application, in particular the jQuery File Upload library. The source code of the project was commented where it was felt necessary, in line with best practice.

4.3 Notable Challenges

The most notable challenges involved the learning of new technologies to a degree where they could be confidently implemented in the project. This involved becoming familiar with an extensive toolset which had including Node.js, Sails.js, MongoDB. There was also a need to learn Mandrill's API and implement it seamlessly in the system in order to send transactional emails.

4.4 Notable Achievements

The most notable achievements from this projects were overcoming the challenges identified above and successfully learning new technologies such as Node.js, Sails.js, and MongoDB to an extent where they could be confidently implemented in the project. Another notable achievement was overcoming issues that arose while building the application using these technologies. An example of this was the problem of asynchronous file uploads, as this could not be achieved using Socket.io or AJAX on its own. The solution to this was the use of jQuery File Upload library developed by blueimp which makes use of the iframe transport method.

5. Testing

5.1 Testing Approach Selection

5.1.1 Static Testing

Static testing can be carried out in multiple methods. The most common method of static testing is code reviews. Code reviews do not involve the running of any code, they are close examinations of the developers source code where the aim is to find and fix any mistakes that may have been overlooked in the initial development phase. The benefit of carrying out code reviews is the potential to improve the overall quality of the system and improve the developers' skills. **(WhatIs.com. 2014)**

5.1.2 Dynamic Testing

Dynamic testing involves executing the programmed code with a given set of test cases. This type of testing may be carried out before the program is complete in order to test particular sections of the code. There are two main types of dynamic testing - white box testing and black box testing. White box testing is a method of testing the internal structures or workings of a program using inputs and expected outputs. This type of testing is usually carried out by the developer of the website or application as they have an internal perspective of the system, as well as programming skills. The test cases for white box testing are usually a reflection of the system requirements and written by the

developer to ensure that the system works as expected. Black box testing is a method of testing the functionality without any knowledge of internal implementation. The tester is only aware of what the application is supposed to do, not how it does it. This type of testing is usually performed using test cases that instruct the tester as to what functionality they need to carry out. The results from black box testing are usually a case of true or false. **(HubPages. 2014)**

5.1.6 Cross Browser Testing and Cross Platform Testing

Cross browser testing is used to ensure that websites or applications behave correctly on different browsers **(Appperfect.com. 2014)**. Cross Platform testing involves testing the application on multiple devices such as mobile devices, tablet devices, laptops and desktops **(Wikipedia | Cross-platform. 2014)**. In spite of the many efforts to standardise the web experience unfortunately no website can be guaranteed to look the same across all browsers and platforms. This is why all good application must be tested on multiple configurations.

5.2 Testing Process

The testing process for this project was carried out in two stages. Developer testing, which was carried out during the development of workulr, and user testing, which was carried out by a user test group.

5.2.1 Developer Testing

During the development of workulr, the application was constantly tested using white box testing to ensure that newly written functionality worked as expected and any bugs were fixed upon their discovery. There was also some static testing carried out in the way of code reviews to a peer group fortnightly. The same peer group were given dynamic walkthroughs of the application and provided some useful feedback. By implementing this type of testing would minimise the possibility of bugs being discovered during the end user testing.

Testing was carried out on a wide variety of devices including iPad, iPad mini, iMac MacBook Pro, Windows 7 PC and Windows 8 Laptop, on a LAN. Given the time constraints, and limited number of testers, and devices available, this testing was not as extensive as would be appropriate in the case of a commercial application. It did however provide a reasonable level of confidence that the application operated successfully on multiple platforms. A number of browsers were utilised on these devices including Internet Explorer v11, Chrome v34, Firefox v27, and Safari v7 on desktops machines. The application was also tested in Chrome iOS and Safari iOS on an iPad2, an iPad mini and an iPhone5 all running iOS7.

5.2.2 User Testing

The user testing was carried out using black box testing. The users selected had no previous experience of using the web application and no knowledge of the underlying programming that makes the application work. These users were given test case forms where they were asked to carry out the main functionality of workulr. There are two user groups for workulr, Standard User and Company Administrator. The functionality for a Company Administrator is the same as a Standard User with the extended dashboard functionality so the test cases for the Company Administrator will only test the additional dashboard functionality as the test cases for the standard user will cover the testing of the rest of the functionality. It was decided that two users for each user group would be ample to test the functionality. As the users were carrying out their tests they were observed by the application developer to see how they navigated and used the site. The application developer was also there to answer any questions the users had about the web application, but not to show them how it worked. Shown in ***Appendix I*** is the test case forms given to users for testing.

5.3 Test Results

White box testing was carried out continuously through the implementation phase of the project to ensure that all required functions operated correctly as they were developed. Cross platform and cross browser testing revealed few issues, apart from minor styling problems. For example, the signup form text alignment was incorrect in Internet Explorer, but worked correctly in all other browsers and across platforms. These results demonstrated the importance of cross platform and cross browser testing rather than simply assuming that because an application works correctly on one platform, it will necessarily work on others.

No major functionality issues were recorder in user testing. A number of minor bugs were detected, for example, form validation messages were displayed multiple times as the user resubmitted the form, rather than only displaying error which still remained. Some times a dates for events were returned incorrectly from the server, because the times stamps had not been run through the Moment node modules. All of the errors were easily corrected however their presence in the application demonstrated the importance of having someone other than the developer test the product.

5.4 User Evaluation

The users involved in the testing were asked to comment on different aspects of the application such as usability and the look and feel. In a larger scale project, this may well have been carried out at multiple stages in development, using alpha and beta testers to provide feedback.

The general conclusions from the user evaluation were that the application was true to its purposes. It was easy to understand, easy to use, and the interface was aesthetically pleasing. No test user had any difficulty in determining how to use any aspect of the application, demonstrating that an intuitive design had been implemented.

6. Evaluation

6.1 Testing and User Evaluation

Both the technical testing, and user evaluation demonstrated that the aim and objectives of the project had been successfully achieved. There were no significant problems detected which needed to be corrected. The minor problems which were detected an corrected demonstrated the value of having proper testing and evaluation in place.

6.2 Project Outcomes

The most obvious outcome of the project was the completion of a fully functional application, which in general met the requirements as originally defined in Chapter 1.2. There were a number of aspects of the application which were not completed as had been hoped, for example, real-time notifications of changes in a user's company information, and hosting on a live web server. Hosting was attempted, although did not succeed to the extent where it could have been used as a live, stable, web application.

While it is not documented in the aims and objectives, it is an implied objective of any university project that the student develops his or her knowledge and skills. Given the wide range of leading edge technologies used, it is fair to say that the project delivered in this area.

6.3 Methodology

Using the modified waterfall was a good choice for the project. The actual practice related very closely to the theoretical methodology, as it was frequently necessary to revisit earlier stages of the project as it progressed. This is to be expected in any project where the learning experience of the developer is as important as the project itself - if he or she is acquiring new knowledge and skills, then a natural outcome of this will be the need to revisit decisions and assumptions made at early stages.

As an example, the system design phase decision to use MySQL was revisited during the development phase, when it became apparent that MongoDB was more suited to the job.

6.4 Plan

Although the Gantt chart was a useful guide, it is fair to say that the project deviated considerably from the original time plan. The most significant cause of this was underestimation of the time required to learn new technologies. Insufficient time had been built in to the plan to allow for problem resolution, and there was no flexibility in the plan to allow for “slippage”.

7. Conclusion

7.1 Report Summary

The project set out to build an easily accessible information hub in the form of a web application. An aim and set of objectives were defined, and a suitable methodology selected to guide the project to conclusion. Detailed requirements were developed, and appropriate prototyping and feasibility testing conducted prior to commencement of the design. User interface, system, logic and data designs were completed, and suitable tools and technologies were selected. Using these, the application was developed, tested, and refined to produce an end product meeting the original aim of the project.

7.2 Project Reflection

One of the most difficult aspects of any university project is selecting a suitable task to complete. The project must be realistic in terms of its scale, so that there is a genuine prospect of it being completed within the time available, but must be sufficiently challenging to ensure that the student is provided with a meaningful learning experience, and develops his or her knowledge and skills.

workulr achieved these goals. Its completion required planning, research, skills development, collaboration with others, and self motivation on the part of the developer.

7.3 Learning Outcomes

Although there is the support of lecturers and mentors in developing the project, it is the responsibility of the individual student to carry through his or her project to completion. An exercise such as this is valuable in developing the student as an individual. It provides insight into all aspects of carrying through a project from start to finish. By its nature however, it cannot provide some of the experiences which would be encountered in a “team” project, for example the need to negotiate with and co-operatively develop with, other team members. It lacks the benefits too, of having colleagues on hand to turn to when difficulties are encountered, and does not allow an individual to specialise on their own strong points. The knowledge and skills acquired throughout this project are transferable to other projects and will be an invaluable asset to the student in tackling future projects.

7.4 Future Work

There are a number of possibilities for future work on workulr, such as the addition of the realtime notifications that were left out due to time constraints. This feature would require further research in using Socket.io. A feature that was suggested but was not implemented, was the ability for a user to find other members with similar interests. This would require working out the logic to search through the interests of members of the same company to find matching keywords. A further development which could be implemented is the creation of native applications for mobile devices. This would require the learning of Objective C for iOS or using PhoneGap **(PhoneGap. 2014)** to allow the use of standard web technologies. Extensive user consultation, and “real world” testing may lead to other future developments for workulr.

References

IcoMoon. 2014. [online] Icomoon.io. Available at: <http://icomoon.io/> [Accessed 10 Apr. 2014].

Mixu's Node Book. 2014. [online] Book.mixu.net. Available at: <http://book.mixu.net/node/ch8.html> [Accessed 10 Apr. 2014].

About Us | LinkedIn. 2014. [online] Linkedin.com. Available at: <http://www.linkedin.com/about-us> [Accessed 10 Apr. 2014].

Wikipedia | Ajax. 2014. [online] Wikipedia. Available at: <http://en.wikipedia.org/wiki/Ajax_(programming)> [Accessed 11 Apr. 2014].

sails-mongo. 2014. [online] GitHub. Available at: <https://github.com/balderdashy/sails-mongo> [Accessed 10 Apr. 2014].

sails-mysql. 2014. [online] GitHub. Available at: <https://github.com/balderdashy/sails-mysql> [Accessed 10 Apr. 2014].

bcrypt. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/bcrypt> [Accessed 10 Apr. 2014].

TheFreeDictionary.com. 2014. [online] TheFreeDictionary.com. Available at: <http://www.thefreedictionary.com/broadcast> [Accessed 10 Apr. 2014].

Cherry, K. 2014. [online] About.com Psychology. Available at: <http://psychology.about.com/od/aindex/g/availability-heuristic.htm> [Accessed 10 Apr. 2014].

Clum, 2014. [online] Uxmag.com. Available at: <https://uxmag.com/articles/a-look-at-flat-design-and-why-its-significant> [Accessed 10 Apr. 2014].

Balderdash, 2014. [online] Sailsjs.org. Available at: <http://sailsjs.org/#!> [Accessed 10 Apr. 2014].

Yammer. 2013. [online] Yammer. Available at: <https://about.yammer.com/> [Accessed 10 Apr. 2014].

Appperfect.com. 2014. [online]. Available at: <http://www.appperfect.com/products/web-testing/cross-browser-testing.html> [Accessed 10 Apr. 2014].

Wikipedia | Cross-platform. 2014. [online] Wikipedia. Available at: <http://en.wikipedia.org/wiki/Cross-platform> [Accessed 11 Apr. 2014].

Express. 2014. [online] Expressjs.com. Available at: <http://expressjs.com/> [Accessed 10 Apr. 2014].

Facebook Groups | Facebook. 2014. [online] Facebook.com. Available at: <https://www.facebook.com/about/groups> [Accessed 10 Apr. 2014].

Fontello. 2014. [online] Fontello.com. Available at: <http://fontello.com/> [Accessed 10 Apr. 2014].

Gandy, D. 2014. [online] Fortawesome.github.io. Available at: <http://fortawesome.github.io/Font-Awesome/> [Accessed 10 Apr. 2014].

Mindtools.com. 2014. [online] Mindtools.com. Available at: <http://www.mindtools.com/pages/article/newPPM_03.htm> [Accessed 17 Apr. 2014].

gm. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/gm> [Accessed 10 Apr. 2014].

Google Developers. 2014. [online] Developers.google.com. Available at: <https://developers.google.com/speed/libraries/devguide> [Accessed 10 Apr. 2014].

Gross, J. 2013. [online] Smashingmagazine.com. Available at: <http://www.smashingmagazine.com/2013/11/19/how-usability-testing-drastically-improved-my-clients-app-2/> [Accessed 10 Apr. 2014].

Gross, Y. 2014. [online] Fontcustom.com. Available at: <http://fontcustom.com/> [Accessed 10 Apr. 2014].

grunt. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/grunt> [Accessed 10 Apr. 2014].

Heroku. 2014. [online] Heroku.com. Available at: <https://www.heroku.com/> [Accessed 10 Apr. 2014].

Nodejitsu Inc. 2014. [online] Nodejitsu.com. Available at: <https://www.nodejitsu.com/> [Accessed 10 Apr. 2014].

iTerm2. 2014. [online] Iterm2.com. Available at: <http://www.iterm2.com/#/section/features> [Accessed 10 Apr. 2014].

Jade - Template Engine. 2014. [online] Jade-lang.com. Available at: <http://jade-lang.com/> [Accessed 11 Apr. 2014].

jade. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/jade> [Accessed 10 Apr. 2014].

jQuery. 2014. [online] Jquery.com. Available at: <http://jquery.com/> [Accessed 10 Apr. 2014].

jquery.iframe-transport.js. 2014. [online] Cmlenz.github.io. Available at: <http://cmlenz.github.io/jquery-iframe-transport/> [Accessed 11 Apr. 2014].

Leggett, D., 2014. [online] Uxbooth.com. Available at: <http://www.uxbooth.com/articles/considerations-for-mobile-design-part-3-behavior/> [Accessed 10 Apr. 2014].

Medero, S. 2007. Paper Prototyping. [online] Alistapart.com. Available at: <http://alistapart.com/article/paperprototyping> [Accessed 10 Apr. 2014].

GitHub | sails-mandrill. 2014. [online] GitHub. Available at: <https://github.com/mikermcneil/sails-mandrill> [Accessed 11 Apr. 2014].

moment. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/moment> [Accessed 10 Apr. 2014].

Tutorialspoint.com. 2014. [online]. Available at: <http://www.tutorialspoint.com/mongodb/mongodb_advantages.htm> [Accessed 10 Apr. 2014].

MongoDB. 2014. [online] Mongodb.org. Available at: <https://www.mongodb.org/> [Accessed 10 Apr. 2014].

MongoHub. 2014. [online] Mongohub.todayclose.com. Available at: <http://mongohub.todayclose.com/> [Accessed 10 Apr. 2014].

MySQL. 2014. [online] Mysql.com. Available at: <http://www.mysql.com/> [Accessed 10 Apr. 2014].

node.js. 2014. [online] Nodejs.org. Available at: <http://nodejs.org/> [Accessed 17 Apr. 2014].

node-uuid. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/node-uuid> [Accessed 10 Apr. 2014].

Open Sans. 2014. [online] Opensans.com. Available at: <http://opensans.com/> [Accessed 10 Apr. 2014].

passport. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/passport> [Accessed 10 Apr. 2014].

passport-local. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/passport-local> [Accessed 11 Apr. 2014].

password-generator. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/password-generator> [Accessed 11 Apr. 2014].

Pastor, P. 2010. [online] Code Tuts+. Available at: <http://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488> [Accessed 10 Apr. 2014].

Penzo, M. 2006. [online] Uxmatters.com. Available at: <http://www.uxmatters.com/mt/archives/2006/07/label-placement-in-forms.php> [Accessed 10 Apr. 2014].

PhoneGap. 2014. [online] Phonegap.com. Available at: <http://phonegap.com/> [Accessed 10 Apr. 2014].

PHP. 2014. [online] Php.net. Available at: <http://www.php.net/manual/en/intro-whatis.php> [Accessed 10 Apr. 2014].

pickadate.js. 2014. [online] Amsul.ca. Available at: <http://amsul.ca/pickadate.js/index.htm> [Accessed 10 Apr. 2014].

sails-mongo. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/sails-mongo> [Accessed 11 Apr. 2014].

Sass. 2014. [online] Sass-lang.com. Available at: <http://sass-lang.com/> [Accessed 11 Apr. 2014].

Satalkar, B. 2014. [online] Buzzle. Available at: <http://www.buzzle.com/articles/modified-waterfall-model.html> [Accessed 10 Apr. 2014].

Socket.IO. 2014. [online] Socket.io. Available at: <http://socket.io/> [Accessed 10 Apr. 2014].

HubPages. 2014. [online] HubPages. Available at: <http://booster911.hubpages.com/hub/Dynamic-Testing> [Accessed 10 Apr. 2014].

DigitalOcean. 2014. [online] Digitalocean.com. Available at: <https://www.digitalocean.com/> [Accessed 10 Apr. 2014].

Stackful.io. 2014. [online] Stackful.io. Available at: <http://stackful.io/index.html> [Accessed 10 Apr. 2014].

StatCounter Global Stats. 2014. [online] Gs.statcounter.com. Available at: <http://gs.statcounter.com/#browser-ww-monthly-201303-201403> [Accessed 11 Apr. 2014].

Style Tiles. 2014. [online] Styletil.es. Available at: <http://styletil.es/> [Accessed 10 Apr. 2014].

Sublime Text. 2014. [online] Sublimetext.com. Available at: <http://www.sublimetext.com/2> [Accessed 10 Apr. 2014].

Symbolset. 2014. [online] Symbolset.com. Available at: <https://symbolset.com/> [Accessed 10 Apr. 2014].

Teamwork.com. 2014. [online] Teamwork Project Management. Available at: <https://www.teamwork.com/features> [Accessed 10 Apr. 2014].

Wikipedia | Traffic light rating system. 2014. [online] Wikipedia. Available at: <http://en.wikipedia.org/wiki/Traffic_light_rating_system> [Accessed 10 Apr. 2014].

Mandrill. 2014. [online] Mandrill. Available at: <http://mandrill.com/> [Accessed 11 Apr. 2014].

SendGrid. 2014. [online] Sendgrid.com. Available at: <http://sendgrid.com/transactional-email/pricing> [Accessed 11 Apr. 2014].

Tschan, S. 2014. [online] Blueimp.github.io. Available at: <http://blueimp.github.io/jQuery-File-Upload/> [Accessed 10 Apr. 2014].

Walter, A. 2014. Aarron Walter. [online] Aarronwalter.com. Available at: <http://aarronwalter.com/design-personas/> [Accessed 10 Apr. 2014].

Learnaccessvba.com. 2014. [online] Learnaccessvba.com. Available at: <http://www.learnaccessvba.com/application_development/waterfall_method.htm> [Accessed 10 Apr. 2014].

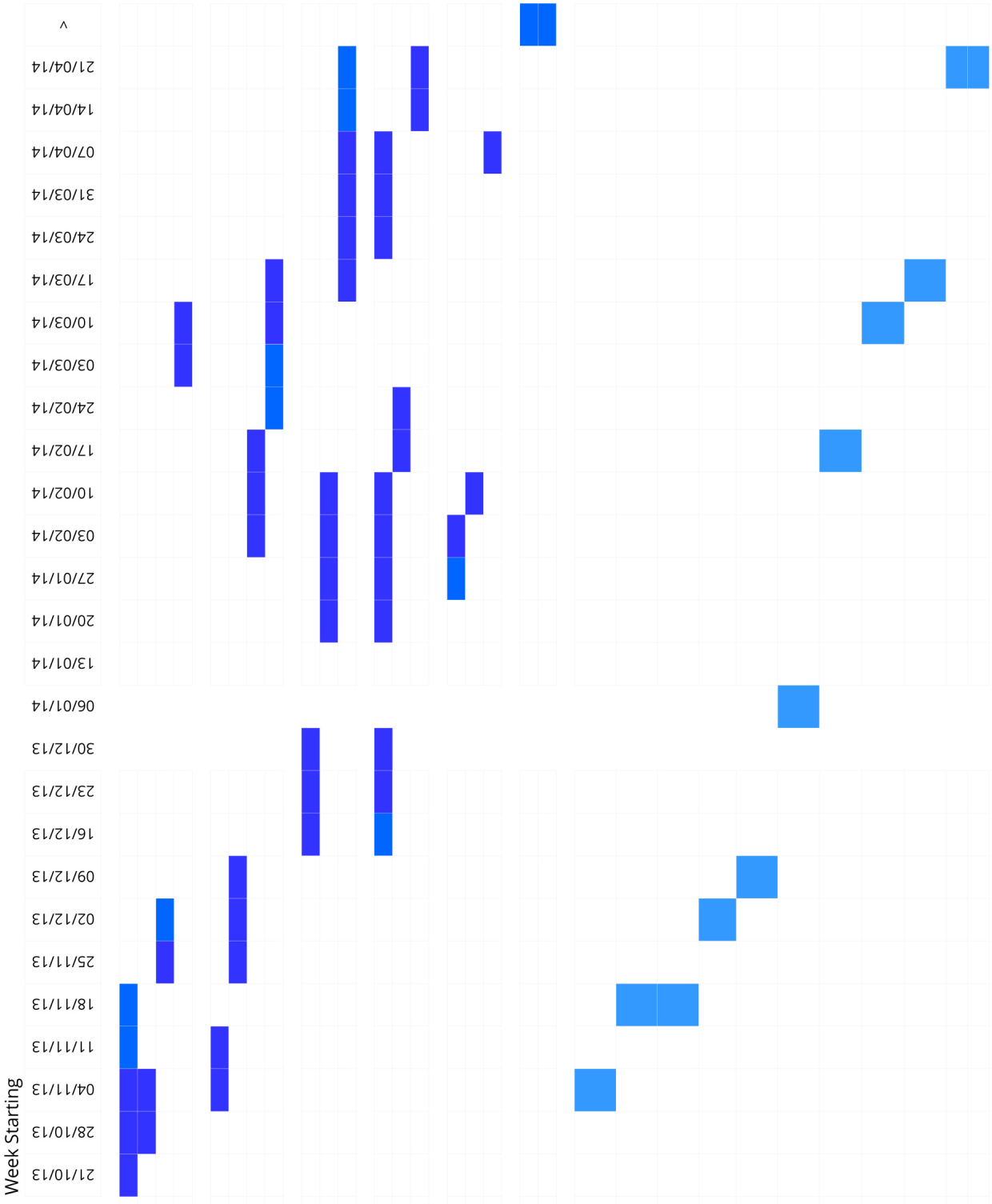
waterline. 2014. [online] Npmjs.org. Available at: <https://www.npmjs.org/package/waterline> [Accessed 17 Apr. 2014].

WhatIs.com. 2014. [online] Whatis.techtarget.com. Available at: <http://whatis.techtarget.com/definition/static-testing> [Accessed 10 Apr. 2014].

Widget Factory | jQuery UI. 2014. [online] Jqueryui.com. Available at: <https://jqueryui.com/widget/> [Accessed 10 Apr. 2014].

Wildbit, L., 2014. [online] Postmarkapp.com. Available at: <https://postmarkapp.com/pricing> [Accessed 11 Apr. 2014].

Appendix A - Gantt Chart Time Plan



Appendix B - Requirements Questionnaire

I am developing a web application service that will allow organisations to store useful information that their staff can access easily. The purpose of this questionnaire is to refine the requirements to address potential end user needs.

How many staff are in your organisation in total?

How many people do you work with in your section or department (approximately)?

How many of those people would you say you know personally?

Would you like to know more about the other staff in your organisation,
such as what they look like, where they are from, and what interests they have?

How many locations does your organisation have?

How many of the staff from other locations are you familiar with?

How well informed of organisation related news and events would you say you are?
(1 - 5, 1 being very well, 5 being not at all)

How are you informed about these events and news?

Email Text Message Social Networks

Would you use a web application service where you could view the following for your organisation:

- Background information about the organisation
- Different locations within your organisation
- Information about the people you work with including what they look like and their interests
- News related to your organisation
- Events related to your organisation

Do you have any suggestions for features of the application that you would find useful?

Appendix C - Functional Requirements

Standard User

Requirement # : 1

Description : The ability to log in to the company account they belong to

Rationale : To keep everything secure and comply with data protection legislation

Dependencies : Requires a Company Administrator to have created the company's account and added them to it

Requirement # : 2

Description : The ability to edit their own profile

Rationale : Allows the user to add/edit as much information about themselves as they wish

Dependencies : Requires the user to have logged in to their company's account

Requirement # : 3

Description : The ability to view all the other people in the company's account

Rationale : Provide the user with an overview of the people they work with

Dependencies : Requires other people to have been added to the account by the Company Administrator(s)

Requirement # : 4

Description : The ability to view each persons profile

Rationale : Allow the user to learn more about the people they work with

Dependencies : Requires other users in the company's account to have populated their profiles

Requirement # : 5

Description : The ability to view information for the company they belong to

Rationale : Allow users to view basic information about the company they work for

Dependencies : Requires Company Administrator(s) to have added information about the company

Requirement # :	6
Description :	The ability to view news and events items for the company they belong to
Rationale :	Allows users to keep up to date with the latest news and events within the company
Dependencies :	Requires Company Administrator(s) to have added news and events items

Company Administrator(s)

The initial Company Administrator will be the member of the company who signs their company up for the workulr account. The initial administrator will have the option to set other users to be company administrators. This user group will have all the requirements of a Standard User with the additional requirements stated below, made accessible through a dashboard. The main role of the Company Administrator(s) is to populate their company's account with useful content.

Requirement # :	7
Description :	The ability to add/edit/remove company information for their company's account
Rationale :	Allows the Company Administrator to control the content for company information
Dependencies :	Requires their company's account to be active and for the user to have Company Administrative rights.

Requirement # :	8
Description :	The ability to add/edit/remove news and events items for their company's account
Rationale :	Allows the Company Administrator to control the content for on the news and events page
Dependencies :	Requires their company's account to be active and for the user to have Company Administrative rights.

Requirement # :	9
Description :	The ability to add/edit/remove locations for their company's account
Rationale :	Takes account for companies who may have multiple locations
Dependencies :	Requires their company's account to be active and for the user to have Company Administrative rights.

Requirement # : 10

Description : The ability to add/remove people to/from their company's account

Rationale : Allows the Company Administrator to control who can access their company's account

Dependencies : Requires their company's account to be active and for the user to have Company Administrative rights.

Requirement # : 11

Description : The ability to edit some information in the profiles of people in their company's account

Rationale : Allows the Company Administrator to ensure data in the system is accurate and suitable

Dependencies : Requires their company's account to be active and for the user to have Company Administrative rights.

Appendix D - System Requirements

Requirement # : 12

Description : The system shall allow users to sign up for an account

Rationale : Uniquely identify the individual user of the system and provide secure access to data associated with their own company

Dependencies : Front end system in place for data input. Back end system in place for data storage. Data validation system in place.

Requirement # : 13

Description : The system shall save all data and any modifications to this data

Rationale : Availability of data is a fundamental requirement of the product

Dependencies : Backend database in place. Data validation in place. Secure login system in place.

Requirement # : 14

Description : The system shall require users of any user group to log in before viewing information

Rationale : Provide the user with only the data connected with their own company. Meet data protection requirements.

Dependencies : User account exists on system. Login screen available.

Requirement # : 15

Description : The system shall detect which user group the logged in user belongs to

Rationale : Differentiate between Standard users and company Administrators to protect control their permissions

Dependencies : Backend database with appropriate user group data in place. Permission controls in place governing the use of various aspects of the system

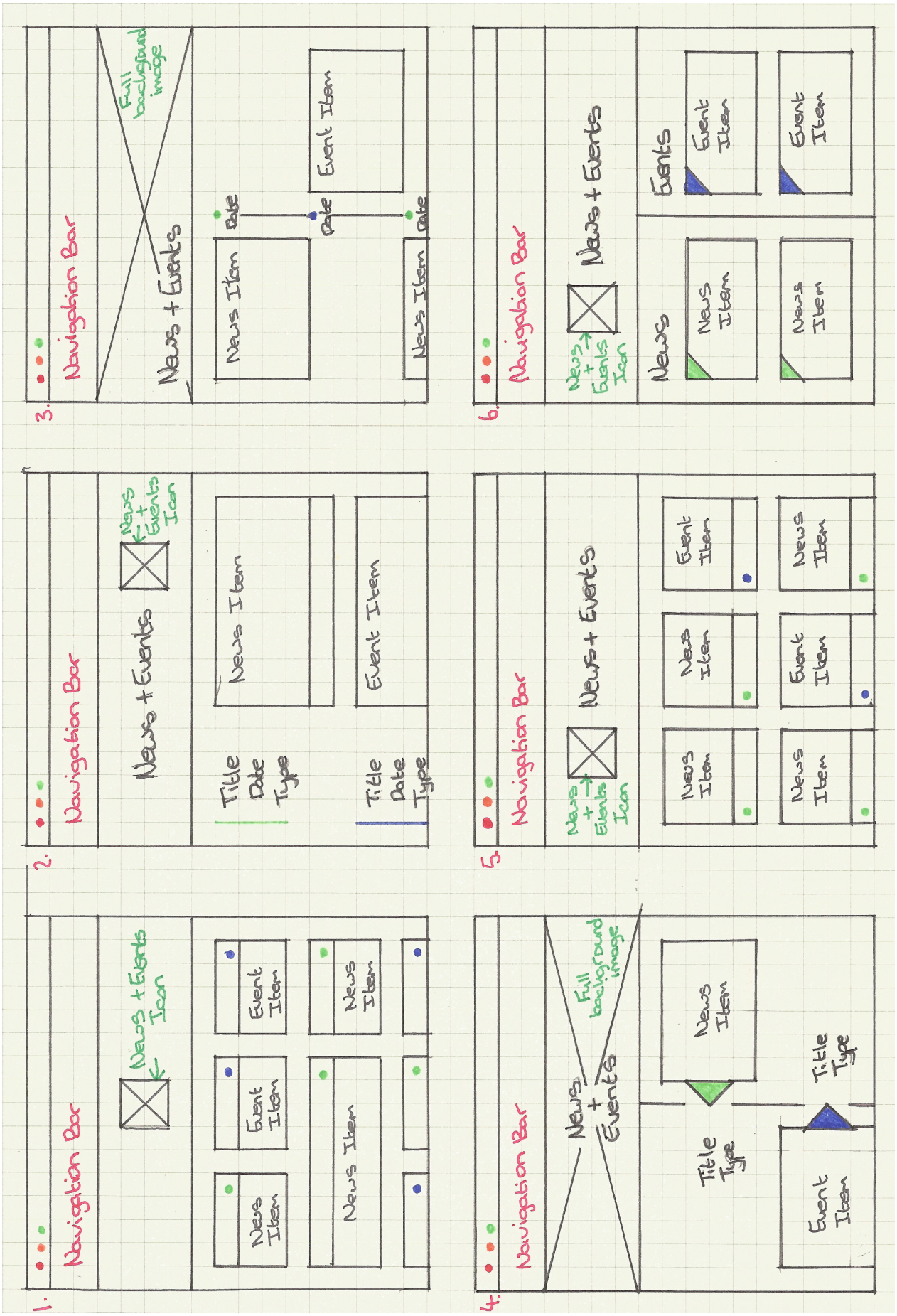
Requirement # : 16

Description : The system shall only provide users with access to information from the company account they belong to

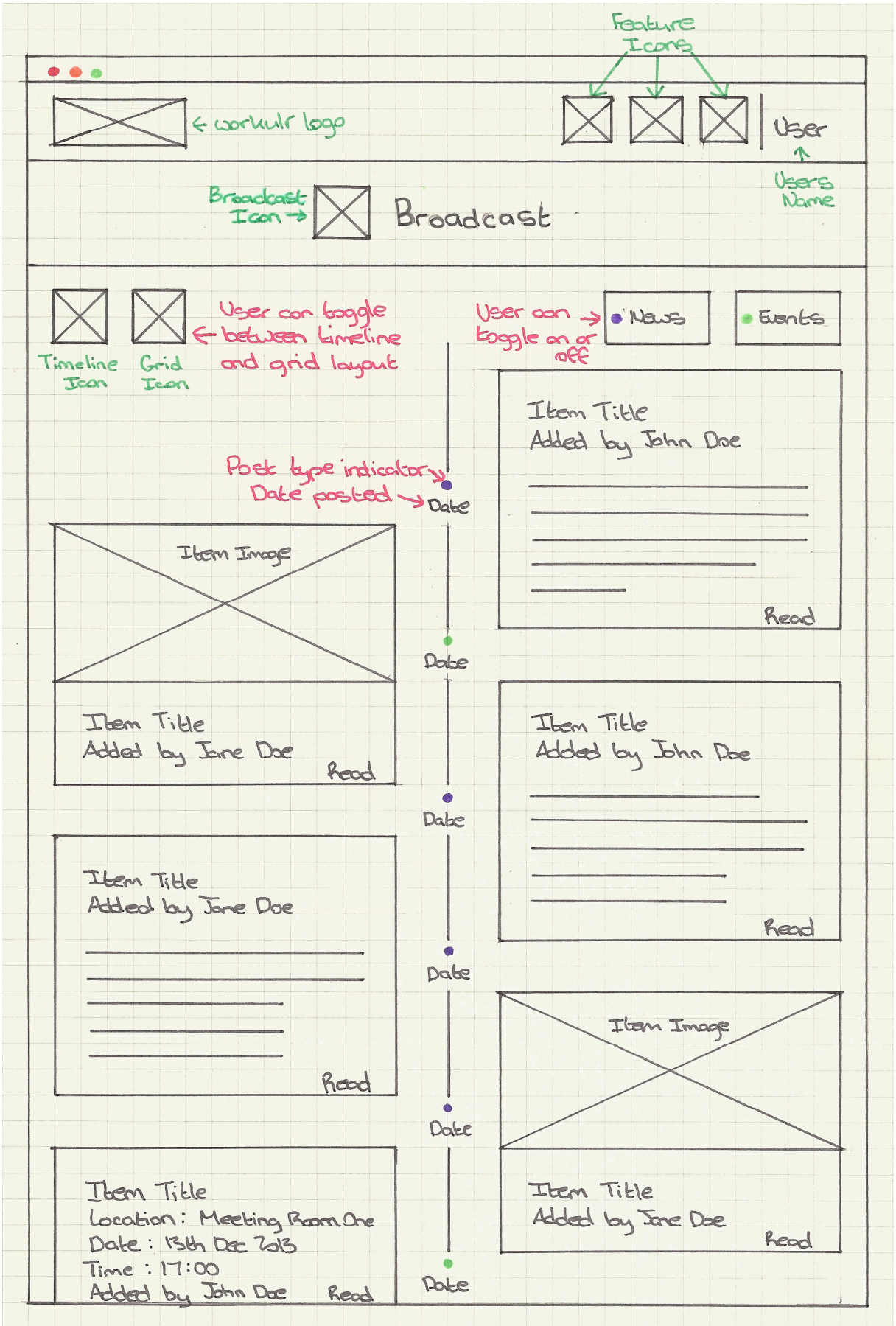
Rationale : Limit data provided to being only that which is useful. Protect data which users should not have access to.

Dependencies : Appropriate logical links between user data and company data in the backend system, controlling what is displayed by the front end system

Appendix E - Six Up



Appendix F - One Up



Risk	Risk Description	Impact (1-5)	Likelihood (1-5)	Initial Risk Level	Mitigation	Post Mitigation Impact x Likelihood	Residual Risk Level
#1	Development machine crashes fatally, losing all project files.	5	3	Amber	Development machine will be backed up automatically to an external hard drive every hour on the hour. Bitbucket will be used to hold a Git repository for the project, regular pushes will be made from development machine to Git repository while development is active.	2 x 3	Green
#2	Development may be slowed down by need to learn new development languages.	3	3	Green	Using languages that the developer has a good understanding of and are widely documented should an issue occur. Assign extra time to the development stage of the project.	2 x 3	Green
#3	Setting up hosting for the project that supports chosen languages may be beyond my skill level.	5	2	Amber	Use hosting such as Heroku, Nodejitsu, or Stackful.io that readily support Node applications.	3 x 2	Green
#4	Hosting server may crash.	5	2	Amber	Choose a reliable hosting service with good uptime reports and 24/7 support.	3 x 2	Green
#5	Illness or other personal problems.	5	1	Green	No mitigation possible, risk accepted.	5 x 1	Green

Risk	Risk Description	Impact (1-5)	Likelihood (1-5)	Initial Risk Level	Mitigation	Post Mitigation Impact x Likelihood	Residual Risk Level
#6	Security risk for personal data stored on server.	5	2	Amber	Ensure all software is patched to latest versions. Ensure hosting environment has a high security level.	5 x 1	Green
#7	Slow/no feedback from potential end users.	3	3	Green	If necessary introduce a new group of potential end users.	1 x 3	Green
#8	Data from one company's account is exposed to members of another company's account.	4	2	Green	Design the system with appropriate security and authentication measures.	4 x 1	Green
#9	Database on server suffers from a failure leading to one or all companies account information being lost.	5	3	Amber	Take regular backups of the database from the server.	1 x 3	Green
#10	Competitive product released before the completion of workulr.	4	3	Amber	Very little mitigation except to develop my product to a standard which ensures its competitiveness.	2 x 3	Green

Appendix H - Style Tile Version One

Colours

#277ACA #242D47 #6BAE44 #E2E2E2

Gradient

#277ACA to #2D3858, 120deg

Buttons

Get Started

Link Button

Save

Action Button

Save

Disabled Button

Adjectives

Trustworthy

Simple

Friendly

Helpful

Easy-Going

Knowledgeable

Typography

H1 Header

Font: Open Sans Light - Google Webfonts
40px

H2 Header

Font: Open Sans Light - Google Webfonts
32px

H3 Header

Font: Open Sans Regular - Google Webfonts
24px

H4 Header

Font: Open Sans Semibold - Google Webfonts
16px

Paragraph - Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Font: Open Sans Regular- Google Webfonts
16px

Link Text

Appendix I - User Test Cases

Company Administrator

Test #	Test	Achieved (True/False)	Comments
1	Sign up for an account		
2	Receive confirmation email		
3	Confirm email and login		
4	Access the dashboard		
5	Add a company to the account		
6	Edit and save company information		
7	Add a location to the account		
8	Add a person to the account		
9	Add a broadcast news item		
10	Add a broadcast event item		
11	Edit a location		
12	Edit a persons information		
13	Edit a broadcast news item		
14	Edit a broadcast event item		
15	Delete a location		
16	Delete a person		
17	Delete a broadcast item		
18	Log out		

Standard User

Test #	Test	Achieved (True/False)	Comments
1	Receive an invitation		
2	Confirm email and login		
3	View available information on company page		
4	View all the people who belong to the same company as you on the people page		
5	View an individual persons profile		
6	View all the broadcast items added for the company account you belong to		
7	Edit and save your profile details		
8	Change your avatar		
9	Change your password		
10	Log out		