BTECH 451 – MID YEAR REPORT

SHANE SINGH | ssin924@aucklanduni.ac.nz

ABSTRACT

Imagine a place where anyone and everyone can come together to achieve what they did not think was possible, communicate with people who are experts in the areas they want learn about. Freedawn is a platform currently in development that envisions providing a place where users can go to achieve whatever is possible and look at self-development and personal growth. My project with Freedawn is to work alongside their current technical developers and improve their present usability and continually evaluate usability improvements.

This report contains a current evaluation of the processes I have undergone in my time working with the Freedawn project and learning the methods of development with a large team of developers in a GitLab environment. It also focuses on the scope of the project set by my industry mentors.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to thank a large number of people for their continual support throughout the time I have put towards the completing of my Bachelor of Technology (Honours) and my BTECH Project. The people I would like to mention and again, thank for their support are:

My parents – Prem and Gyan Wati Singh

My partner – Georgie Crispe

My academic supervisor – Dr. Ulrich Speidel

My academic co-ordinator – Dr. S. Manoharan

My industry mentors – Eddie Freeman and David Noble

INTRODUCTION

Have you ever wanted to learn something or do something that you were not sure whether you had to the time for or the ability to do? Something you thought was out of reach; a hopeful ambition that you would love to achieve. Freedawn is the platform that will allow you to do so.

Freedawn is a platform currently under development in Auckland, New Zealand that will allow anyone to communicate about similar goals that they would like to achieve, to develop themselves into better people and to learn new skills and talents along the way. It will also give individuals the ability to personalise their own goals and to have a tangible medium to monitor progress and help reach their full potential.

The development and surge in popularity in social networks has been dramatic. In January 2015 there were 2.078 billion recorded active social media accounts with the population at the time recorded at 7.210 billion. These statistics reveal that an equivalent of 29% of the world's population have active social media accounts. (Kemp, 2015)

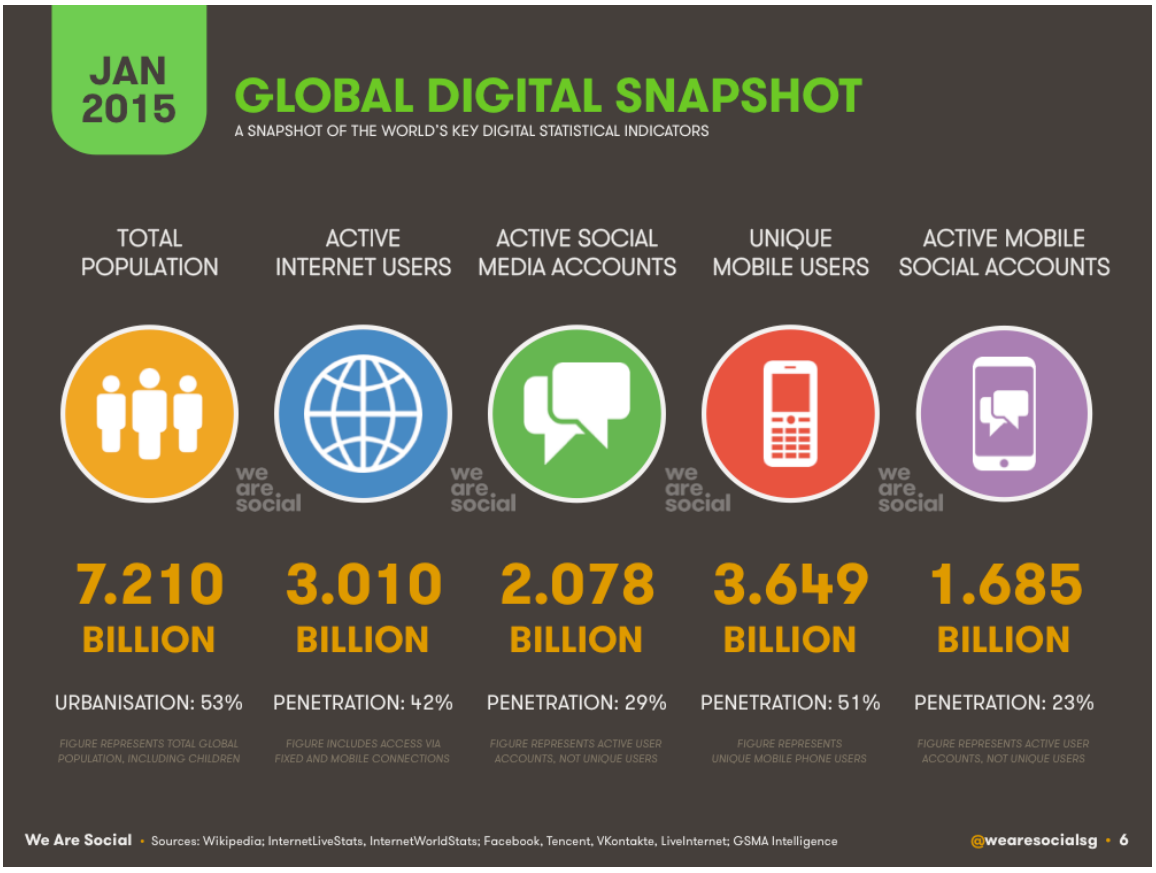

FIGURE 1 - GLOBAL DIGITAL SNAPSHOT (KEMP, 2015)

As Freedawn's development progresses, hidden behind its social media platform also lies a feasible business model – the ability to allow users to communicate with Mentors and Experts in particular areas and pay them for their knowledge and mentorship.

Freedawn is currently in a private-beta stage, which means that there are a select few users who have been handpicked to test Freedawn's functionality as it is being developed and implemented until the platform is at a stage that it is ready for public testing. At present, the baseline functionality is existent, however incremental changes to each module are currently being carried out to progress the platform further. With the particular nature of Freedawn, it is a platform that will undergo continuous development to suit the ever-changing needs of its targeted audience, and it will change dynamically over time instead of having a final goal.

PROJECT & PROJECT SCOPE

This project with Freedawn follows its progression from the opening stages to the development of the site. The basic platform for the site has been chosen, with the ideas and modules identified in the business model having already been proven to be key components for a successful social platform for self-development and personal growth. The project scope focuses on the usability of the platform in its entirety from first-time users all the way to advanced users as well as the development of module integration.

As modules are custom built for Freedawn, the implementation to the current structure should be seamless and testing should be undertaken to ensure that functionality is not altered when a module is implemented. Testing and modification to these modules is a critical part of this project to ensure that in the future the scalability of Freedawn will not be hindered by the modules and that the modules provide a stable solution to the business needs of Freedawn.

DEVELOPMENT ENVIRONMENT

As the business model and idea for Freedawn targets a global market, the initial infrastructure must be extremely stable, scalable and efficient. It must be able to provide what users want quickly and easily. There are three fundamental cloud service models: Software as a Service, Platform as a Service and Infrastructure as a Service (Appcore, 2015). Freedawn operates on the Amazon Web Services Infrastructure as a Service system.

Infrastructure as a Service (IaaS) encompasses Cloud Platform Services, which are used for a number of applications. The use of a cloud-based infrastructure means that there is no need for a server and data to be hosted on-site. Essentially, the cloud contains virtualized hardware, with the hardware resources being physically utilized across a large number of servers and data centres. There are a number of benefits as to why using an IaaS and a cloud-based infrastructure is quite ideal for a business.

- **Scalability** - Resources are available on-demand and there are no delays in expanding capacity or the wastage of unused capacity.
- **Minimal upfront investment** - The physical hardware that is the backbone of the IaaS service is setup and maintained by the cloud provider, in this case Amazon, and this saves hugely in time and money.
- **Utility style costing** - As a user, the service can be accessed on demand and only the usage that is actually consumed is charged.
- **Location Independence** - The service can be accessed from any location, as long as there is an internet connection and sufficient security privileges
- **Physical security of data centre locations** - Services available through a service provider are secured and maintained by the provider themselves and hosted within a data centre.
- **No single point of failure** - If one server or networking hardware was to fail, the broader service would remain unaffected due to the ability to use many physical resources and redundancy configurations. This maximises uptime and minimizes downtime, which can affect a business especially a globally accessed website.

The choice of Amazon Web Services (AWS) as the Infrastructure provider was for many reasons – the ability to have an infrastructure that is stable, scalable and cost-effective are all necessary requirements for a successful business model. The idea of scalability for a social platform is also largely taken into consideration for service provision. The ability to easily add resources and capacity to Freedawn as the user population base grows will be vital. AWS provides an easy-to-use interface, both for managing instances as well as organising critical backups making it a very reliable service. At present, there is no utilization of a Content Delivery Network (CDN), which is a network of distributed servers across multiple data centres with the ability to serve content to end-users with high-availability and performance. This would be ideal for when Freedawn grows its user base and international population on the platform increases – fast internet access is widely considered to be a productivity-enhancing factor as well as a usability factor. The maximum number of seconds a user is willing to wait on average before abandoning a web page is 8.6 seconds (King, 2003). Load speed has a massive impact on conversions for ecommerce platforms. The impact on Freedawn may not be as relevant, but it would heavily affect usage traffic and user decisions about reusing the website.

Another reason why the AWS infrastructure was chosen is due to the fact that it is a secure platform; in AWS privacy of personal data comes under extreme scrutiny. The requirement for data protection, especially since recent events and leaks have come to light, is even greater. In 2013, the US Retail Chain Target suffered a crippling data breach leading to both the loss of customer credit card information as well as data including names, addresses, email addresses and phone numbers. The number of records stolen totalled to 70 million. The percentage drop in profits at Target in the fourth quarter was 46% in comparison with the year before (BBC, 2014). The gravity of importance that this brings to business success and the success of a platform is quite high. Although financial data is not entirely relevant to the current model of Freedawn, it can be related to the number of users and the usage rate of users.

The foremost security concern is physical security of the hardware. AWS mitigates this risk by having all their servers and resources in Data Centres that are monitored 24x7 with user's access authorised on a least privileged basis. Multiple geographic regions and availability zones allow users of the AWS services peace of mind that there is redundancy if a particular link is to fail. The AWS infrastructure is continuously monitored and protected by an extensive network of security protocols to ensure that there are basic security measures in place for Distributed Denial of Service (DDoS) protection and brute-force detection on AWS components. Automation of tasks especially security tools is utilized to manage

credentials, monitor server and network usage as well as application usage. These tools are developed primarily by the AWS security team and the use of automation affects the velocity at which intrusions and security risks can be detected. (Amazon Web Services, n.d.)

DISTRIBUTED REVISION CONTROL

There are many open-source distributed revision control mechanisms that can be used to set up version control for software development. When there is a large group of developers there are occasionally overlaps of development for tasks that each developer is completing. The ability to merge these changes from each developer, while independently ensuring that their development tasks are complete and integrated seamlessly is a highly sought after feature

Freedawn has utilized GitLab as a method for distributed revision control. GitLab works in quite a simple manner and its revision control is utilized by Freedawn in quite a clean yet extremely effective way. There are many developers that are based internationally with experience in modifying the particular platform Freedawn is based on and they all make their changes on their own independent machines providing them with their own development and testing instance.

A basic concept of how the model works is shown below:

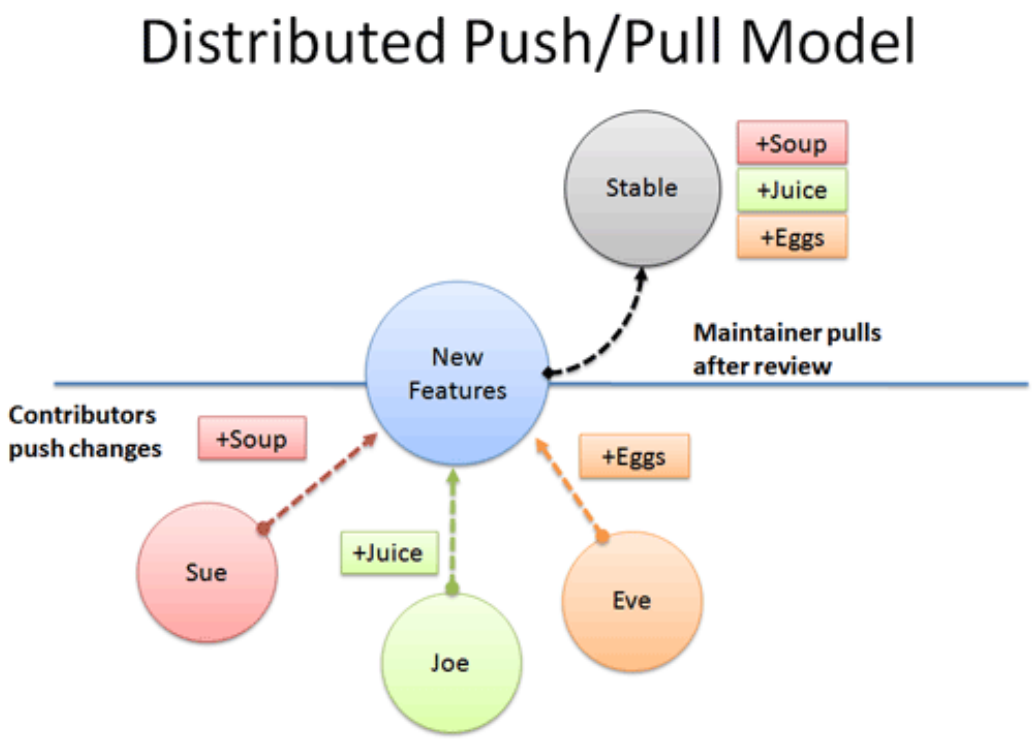

FIGURE 2 - ORGANISING A DISTRIBUTED PROJECT (AZAD, N.D.)

The developers in this case are Sue, Joe and Eve. Sue contributes “Soup”, Joe contributes “Juice” and Eve contributes “Eggs”. These features are all pushed to a management repository that contains all the features. In this case, the features from each developer are merged in the form of “New Features”. This means that, in between, the development that has been completed can be reviewed and tested prior to pushing it into the live environment. The revision management software used by Freedawn called GitLab

is used to manage revision history. This is ideal to manage any errors made and to make sure that there is leeway to roll-back to previous versions quickly and easily with minimal downtime.

At joining the Freedawn project, each developer is provided with their own environment to work in – this way all development testing can be completed in an independent environment before it is pushed for review. The benefit this brings is it ensures the quality of the work and also ensures that if other developers are working on tasks simultaneously my testing and changes do not interfere until my edits are committed and pushed to the GitLab repository. Each developer is provided with their own instance of an AWS machine however all these one development machines are linked to the same Development Database. This way all the content is consistent across the different instances, however the underlying code of phpFox is independent.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
Freedawn Blog	i-1388cadc	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn	i-f4842028	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn	i-6bcc59b7	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn LIVE	i-f73ae52b	m3.medium	ap-southeast-2b	running	2/2 checks...	None
Freedawn	i-e23eb93e	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn Shane	i-b1340d7e	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn	i-a1ab187d	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn	i-f1971e2d	t2.micro	ap-southeast-2b	running	2/2 checks...	None
Freedawn Testing	i-0fb881c0	t2.micro	ap-southeast-2b	running	2/2 checks...	None
GitLab	i-2f3d04e0	t1.micro	ap-southeast-2b	running	2/2 checks...	None

FIGURE 3 - FREEDAWN INFRASTRUCTURE ENVIRONMENT

In Figure 3 we see how Freedawn has their development server's setup, with a number for the developers currently working on the project. The GitLab instance hosts all the changes once made by the developers, once the developers have completed a task the industry mentors pull the updated content on the Freedawn Testing environment, where as part of my project I begin to dissect the developed code, test for any errors and bugs as well as merge with the existing modules and phpFox.

The Instance Type which is commonly for the Development servers are "t2.micro" servers which provide 1 vCPU with a capability of 2.5GHz using Intel Xeon processors as well as 1GB of RAM. The reason for such a low capacity of technical capacity is that the instances are put to quite low use for a majority of their development-life. One of the benefits the Amazon AWS infrastructure brings is that on the fly, any of the particular instances can be upgraded to a much larger and much more resourceful hardware to undergo testing for load and scalability. This can be done in a matter of minutes, with no downtime from the switchover. One of the business benefits this brings is that the cost of using the more powerful instances is only charged based on the consumption then the instance can be scaled back to the "t2.micro".

PHPFOX

phpFox is a platform developed by Moxi9, that is specifically designed for social networking. The basic idea of this platform provides the fundamental base of Freedawn. The version used by Freedawn is version 3, which is based on PHP and utilizes the smarty engine template. phpFox is based on a module structure using the Model View Controller (MVC) model. MVC is a software architectural pattern for implementing user interfaces dividing a software application into three interlinked segments (Reenskaug & Coplien, 2009). The controller component sends commands to the model to update that state of the model as well as commands. The model stores data retrieved by the controller and displays it in the view. The view requests information that it uses to generate an output representation to the user.

MVC framework is broken down into the following components:

- **Model** - Model objects are parts of the application that implement logic for the application data. Often, model objects retrieve and store the state of the model in a database. For example, they retrieve a goal name from the Goals table in the database, operate on the record and write the updated information back to the Goals table.
- **Views** - These are the components that display the application user interface. This is created from the model data. For example, when you edit a goal it will populate the screen with the drop down boxes and different text boxes with the appropriate data in a format based on the state of the object.
- **Controllers** - These are components that handle the user interaction, work with their model and render the display in the UI. The controller will handle and respond to their interaction.

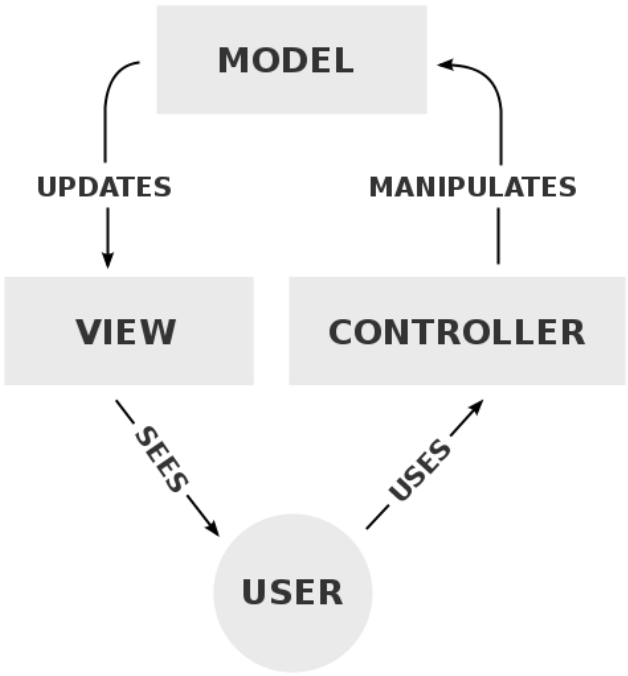

FIGURE 4 - MVC VISUAL REPRESENTATION

CONTROLLERS

phpFox uses three controllers:

1. **AJAX Controller** - This controls and replies to all AJAX requests from the Client (Web Browsers).

2. **Block Controller** - This controller requests data from the service and assigns the values for the view.

3. **Main Controller** - This gets all the responses from actions done by the user and communicates with the end users.

phpFox uses modules to integrate much more content with the system in order to create an adequate Social Networking Solution. The modules introduced bring in more features to the site so it transforms it from a Content Management System (CMS) to a fully-fledged platform that has much more capability.

phpFox uses the Smarty Template Engine, which essentially facilitates a manageable way to separate application logic and content from its presentation. This is best described in a situation where the application programmer and the template designer play different roles. In a platform like Freedawn, which does require efficiency and speed as the user base increases, we can see that the use of a template engine like Smarty is particularly effective. Its main features are that it is quite fast, especially since the PHP parser does the majority of the work. It is also quite “smart” in the area of recompiling as it only re-compiles the templates that have changed. Smarty provides developers to create custom functions and variable modifiers so that you can extend the engine to suit your needs and further develop the platform. (Smarty Template Engine, 2015)

With large websites, the use of a template engine can highlight the importance of separation. Smarty provides clean tag-based syntax as well as a variety of tools to manage presentation. In cases where efficient template management is quite a vital aspect to the platform development tag-based syntax is extremely helpful especially if the project scales to hundreds of templates. A use case of Smarty is when there are platforms, like Freedawn, with one or more PHP developers and one or more web designers. Web designers who generally focus on the layout and graphical design may potentially be unfamiliar with PHP. The use of a template engine that is tag-based gives them the ability to understand, modify and maintain.

```
{if count($aHabits)}
{if $aView == 'my' && Phpfox::getUserBy('profile_habit_id')}
<div class="message">
    {phrase var='habits.note_that_habits_displayed_here_are_habits_created_by_the_habit' global_full_name=$sGlobalUserFullName|clean profile_full_name=$aGlobalProfilePageLogin.full_na
</div>
{/if}
{foreach from=$aHabits name=habits item=aPage}
<div id="js_habits_{$aPage.habit_id}" class="js_habits_parent {if is_int($phpfox.iteration.habits/2)}row1{else}row2{/if}{if $phpfox.iteration.habits == 1 && !PHPFOX_IS_AJAX} row_first
        <div class="row_title">
            <div class="row_title_image">
                <a href="{$aPage.link}">{img server_id=$aPage.profile_server_id title=$aPage.title path='core.url_user' file=$aPage.profile_user_image suffix='_50_square' max_width='5

                {if Phpfox::getUserParam('habits.can_moderate_habits') || $aPage.user_id == Phpfox::getUserId()}
                <div class="row_edit_bar_parent">
                    <div class="row_edit_bar_holder">
                        <ul>
                            {template file='habits.block.link'}
                        </ul>
                    </div>
                    <div class="row_edit_bar">
                            <a href="#" class="row_edit_bar_action"><span>{phrase var='habits.actions'}</span></a>
                    </div>
                </div>
                {/if}

                {if Phpfox::getUserParam('habits.can_moderate_habits')}
                <a href="#{$aPage.habit_id}" class="moderate_link" rel="habits">{phrase var='habits.moderate'}</a>
                {/if}
            </div>
            <div class="row_title_info">
                <a href="{$aPage.link}" class="link">{$aPage.title|clean|shorten:55:'...'|split:40}</a>
                <div class="extra_info">
                    <ul class="extra_info_middot"><li>{$aPage.category_name|convert}</li>{if $aPage.habit_type == '1'}<li><span>&middot;</span></li><li>{if $aPage.total_like > 1}{phra
                </div>
```

FIGURE 5 - HABITS CONTROLLER CURRENTLY UNDER DEVELOPMENT AS PER PROJECT WORK

MODULES

The file structure for phpFox modules is as follows:

`/module/` - This is where all modules are contained.

`/module/newModule/include/` - The logic for a module called “newModule” is stored inside this directory.

`/module/newModule/include/component/controller/` - This is where the controller for the module “newModule” are stored, which are the components handling the user interaction.

`/module/newModule/include/service/` - This is where the services for the pure logic is stored, which is the model component.

`/module/newModule/template/` - This is where the layout and user interface of the particular module “newModule” is stored.

A visual representation of the module “freedawn”:

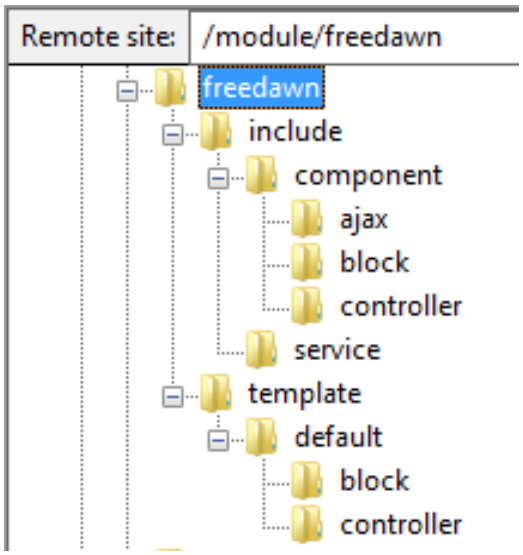

FIGURE 6 - PHPFOX MODULE STRUCTURE

DATA SECURITY

The base version of phpFox, which can be purchased for a one-time fee, is an extremely secure website. It is the further development of the platform that can cause security vulnerabilities. This can be due to different coding standards especially when using a variety of different developers.

There are also many basic principles that can be followed to ensure that data can be secured. At present, there are no extra data security measures in place. However, recommendations for the following could be taken to improve data security and integrity as the platform grows.

- **Use of Two-Factor Authentication (2FA):** This uses your password as well as another device at time of login, such as your cellphone to check your identity. 2FA works by having an independent code generator or a code being sent by SMS, which means that the website will match up with the code as well as check that the passwords match. This reduces the ability for people to either use a brute-force attack or use social engineering to gain unauthorised access to user accounts. (SecurEnvoy, 2015)

- **Least Privilege Principle:** This is part of the abstraction layer where with each module a particular user can only access the information and resources that are necessary for its legitimate purpose (Rouse, 2008). In essence, if a user has not paid for the premium content, they should not be able to access a portion of the premium part of the platform. The use of authentication and privileged user levels on their accounts will ensure that particular users are unable to access places including the Admin Control Panel unless the privilege is granted. This helps mitigate the risk based on community and user trust and ensures that there is no unwanted abuse of the site.

Freedawn is currently in the beta stages and is essentially running off the bare phpFox platform, no significant changes have been made that compromise the initial security and integrity that phpFox provides. However, as the scale of development increases so should the time on security development. The loss of private information on a global scale is potentially crushing to a site's reputation.

As Freedawn is a start-up, a recommendation to development would be to ensure focus on security at all aspects of development when adding modules and custom-development to suit the needs of the site. This way at every level a security policy is implemented to ensure further down the project it does not creep up and become a substantial task to complete.

USABILITY

The main core of my project focuses on the usability of each area, the functionality and compatibility of each module as well as ensuring that each module works on a scalable basis. There are many parts to working on the usability of the website, especially with Freedawn aiming to be a social media platform as well. This encompasses a more modern usability approach where those that have been found to have a minimum design, simplicity, accessibility, consistency, feedback and provide forgiveness to the user are very driven to increase the usability of the user and assist with their performance on the website. There are ten general principles for interaction design, which are "heuristics" as they are broad rules of thumb and not specific usability guidelines. The 10 Heuristics established by Jakob Nielsen for User Interface Design are:

1. **Visibility of System Status** - Visibility requires users to stay informed about where they are in relation to other pages in the site. Providing a mental sitemap to easily navigate across the website is essential. The classic definition is that "the system should always keep users informed about what is going on, through appropriate feedback within reasonable time" (Nielsen, 1995). Freedawn has a built-in XML sitemap that is generated by phpFox, and it also assists users with their visibility by supplying a Breadcrumb at the top of each page in the header bar. A Breadcrumb is basically one line of text showing a page's location in the hierarchy of the site. This Breadcrumb is active for each page and also provides an excellent mental indication of where the user is located throughout the website. The Breadcrumb helps increase navigation efficiency for users who launch into different pages especially from referrals such as Google Searches.

   Another way that Freedawn has provided visibility to the user of system status is with the loading circle, when something is loading there is an animated loading circle to show the user that there is currently something in progress instead of users waiting and being unsure as to what the outcome of a particular click will be. To assist with the improvement of the usability of Freedawn, a timeout function on the loading circle should be implemented. This is done so that if there is an error with loading a particular request, it should inform the user that the request has been aborted within an appropriate wait time and ask the user to Try Again or request something else.

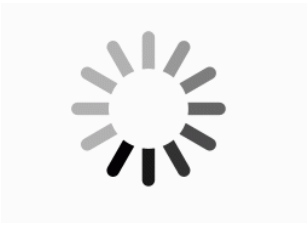

FIGURE 7 - FREEDAWN LOADING CIRCLE

2. **Match between System & Real World** - This deals with the importance of a website being readable and in formats that cater to a user instead of using unconventional formats and creating user difficulty. “The system should speak the users’ language, with words, phrases and similar concepts familiar to the user rather than system-oriented terms” (Nielsen, 1995). As Freedawn is a social media platform, it will utilize current popular social media usability to ensure that users transitioning and using the website have similar usability to decrease learning time to navigate the website.

   For Freedawn, this particular area will be important when it comes to navigation and usage of particular functions. As a social media website, the similarities between other social websites will be quite high, but in order to make Freedawn successful the key features that make it different from other social platforms must be able to be integrated easily with as little difficulty for the user as possible.

   An example would ideally be the navigation button, in the beta the navigation button is currently a generic menu icon. However, it is only activated on-click as opposed to on-hover. From my research on other websites, the majority of header buttons are activated on-click as well, but on-hover they have a highlight function that changes the colour of the buttons. This may lead to user confusion as they may not be sure if the menu icon is a clickable feature of the website.

3. **User Control & Freedom** - “Users often choose system functions by mistake and will need a clearly marked ‘emergency exit’ to leave the unwanted state without having to go through an extended dialogue” (Nielsen, 1995). This ability to take control of wherever the user goes to on the website should allow them to easily undo/redo changes they have made. Should a user become lost in the content or unable to return back to where they were previously, it will cause user frustration and the user will most likely leave the site altogether. A well-designed interface ensures that users are comfortable with their travel decisions within the site and provides easy avenues of access. This can be seen with Freedawn where the Flame in the header takes users directly back to their Public Profile. The ability to always have that header on every page ensures that there is that avenue of an “emergency exit”.

   A substantial number of websites that are quite popular based on the Alexa Top 500 Websites, which include Google, Facebook, Youtube and Yahoo, have both their logos and icons that represent their brand as clickable icons that take the user back to the index of their website. As the development of the Freedawn Beta continues, this would be essential to provide ease of use to users (Alexa, 2015).

   The use of Breadcrumb navigation has also been found to be increasingly useful. It essentially provides secondary navigation to the website. This is another one of the “emergency exits” for users that assist with taking them through the higher levels that they would have travelled into the site.

4. **Consistency & Standards** - Keeping a website consistent makes the user feel more comfortable. As users are most likely going to travel through a variety of different pages, they should essentially retain the same layout throughout and look uniform. It can become quite confusing and increase user frustration when the same terms are used to define different things, when similar content is placed in different places depending on the page and when there is no uniformity like the header or footer. Users should not have to wonder whether different words, situations or actions mean the same thing.

   The consistency applies to both navigation and layout as well as terms that are relative to Freedawn, such as Feedback Groups, should always be referenced and called Feedback Groups, Goals and Habits as well. This will ensure a greater consistency throughout the site.

   With the Freedawn Beta, at present there are only three options from the Navigation Button. These options are always the same, therefore maintaining navigational consistency to ensure that users always know where they are going and where they are.

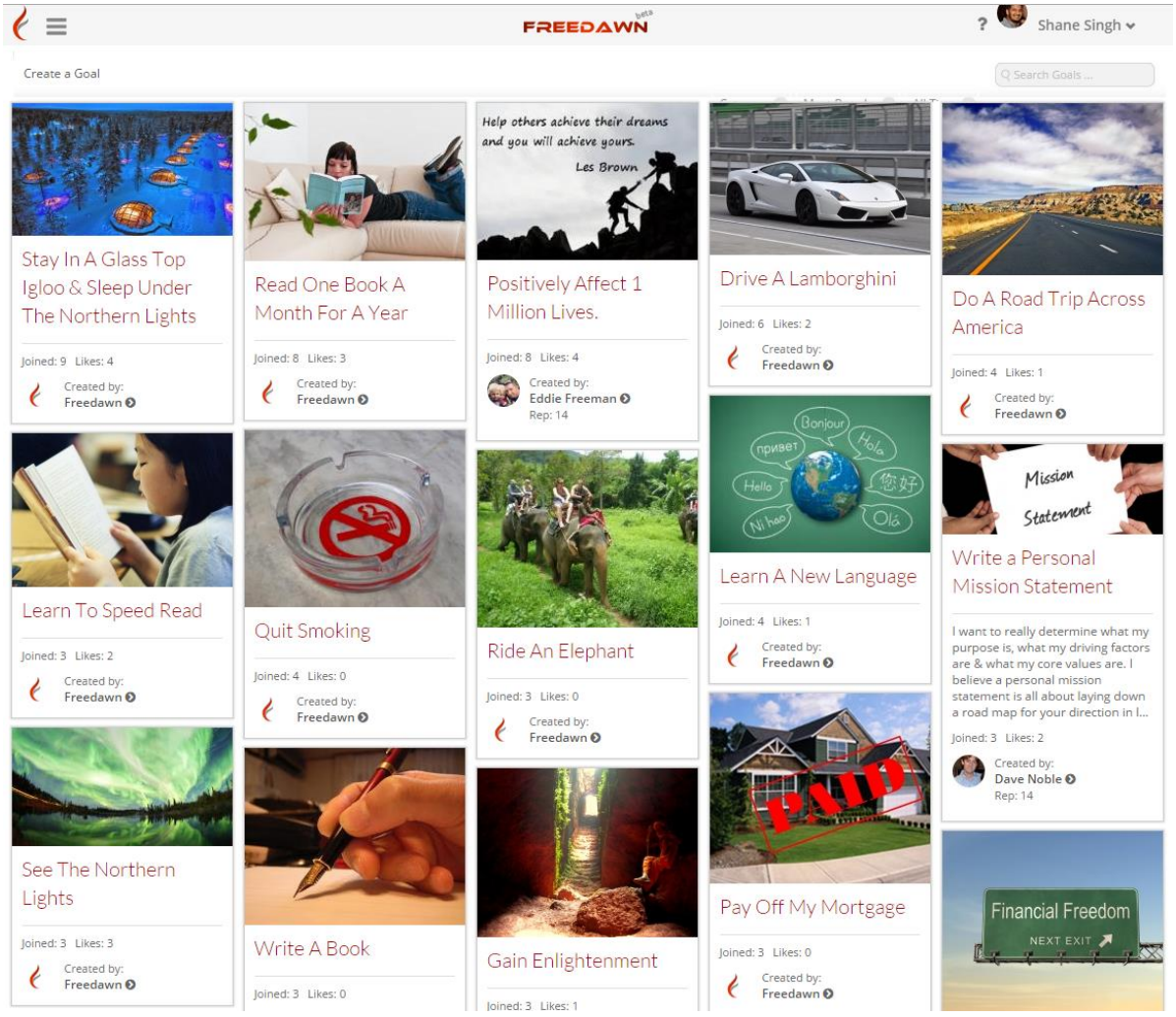

FIGURE 8 - FREEDAWN GOAL MINE

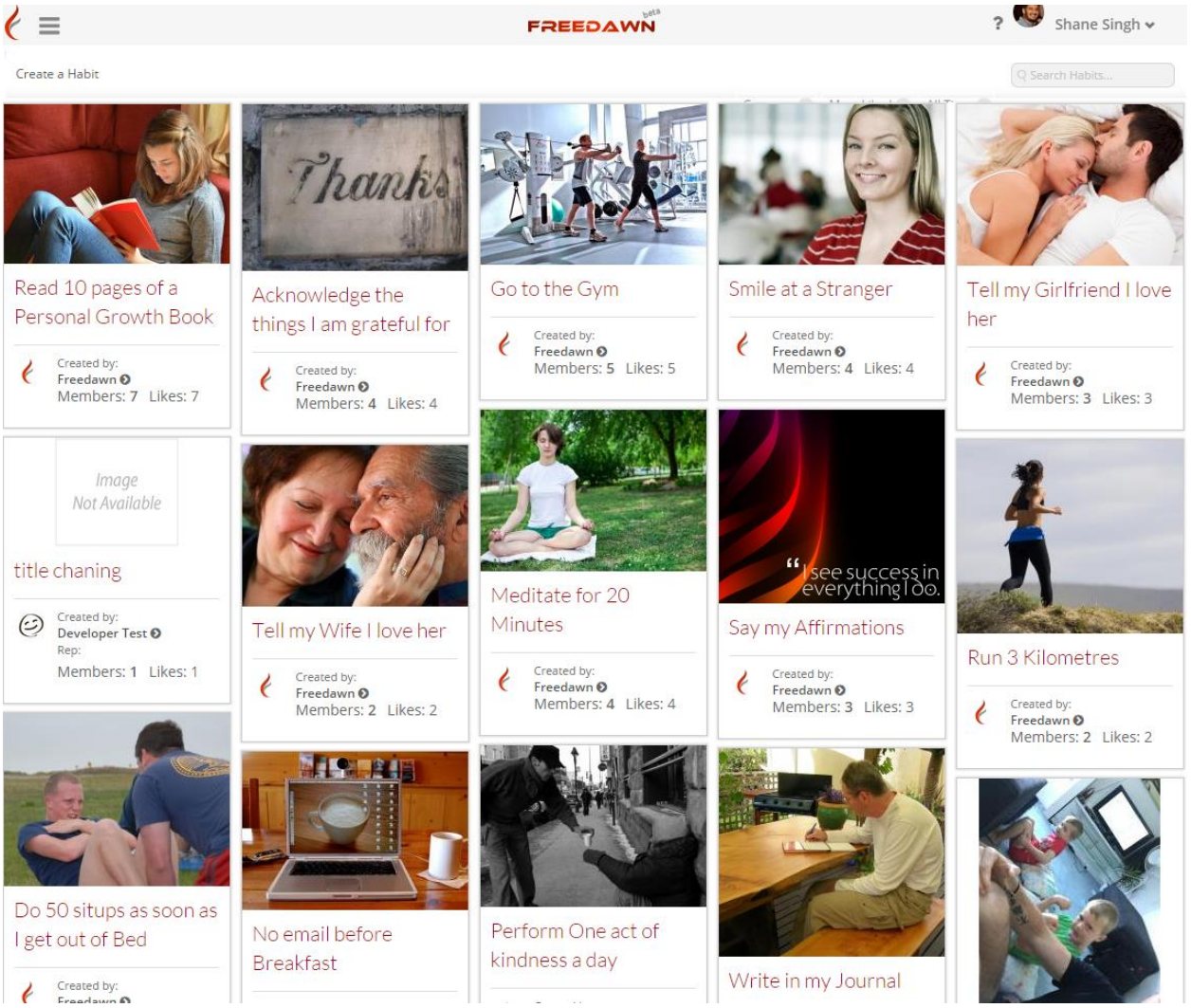

FIGURE 9 - FREEDAWN HABITS

5. **Error Prevention** - "Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action." (Nielsen, 1995)

   With Freedawn, the concept has been discussed with the Development Team at the ability of incorporating content from various different sources such as Youtube and Vimeo as well as text-based sources such as PDF documents. Hyperlinks have the ability to potentially distract users from not realising that they are leaving the website if they click on them. An important tool to ensure users do not make the error of opening up another website or document entirely by mistake is to ask users "You are about to leave Freedawn - are you sure you wish to do so?" or something along those lines to confirm the user would like to navigate away.

Another common method of error creation is by using forms. With Freedawn, forms are potentially going to use AJAX and Java to ensure that they are validating on-the-fly instead of after users fill in an entire form, submit it and then are thrown back to the data entry page with a list of validation errors. The ability to prevent this inconvenience with on-the-fly validation will help increase efficiency on the website and reduce user frustration.

6. **Recognition rather than Recall** - “Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.” (Nielsen, 1995)

   Memory recognition is performed more easily than memory recall. This idea ties in with usability designs where users should not be expected to remember what page they are currently on or how they got there. This is another aspect where the use of Breadcrumb navigation brings a further benefit to the site as it is clearly a navigational path the user has taken to get to where they are.

   Recognition is promoted in User Interfaces by ensuring that the three main components — the interface including all the buttons, navigation and page elements — are there to help the user achieve their goal and not distract them or cause frustration. Also, the content needs to be relevant to their needs and conducive to them achieving their goal. For example, with Freedawn the icons in the header bar are easily recognised aside from the thunderbolt that represents Notifications. The ability to tie in recognition can be utilised by introducing labels to things that are not automatically recognised or do not require prior knowledge to know what the connection is. Freedawn has a “Like” function built in, however to differentiate the icon is:

   

   with the number indicating the number of people who have “Liked” particular content.

   A poor example of recognition is shown below where a Mail application developed for iOS has a group of icons at the top right above the main email content, however they do not promote easy recognition aside from the ‘X’, which is a fairly common icon. A label would be ideal for these buttons as they do not give the user an indication of what they represent unless they are tapped on.

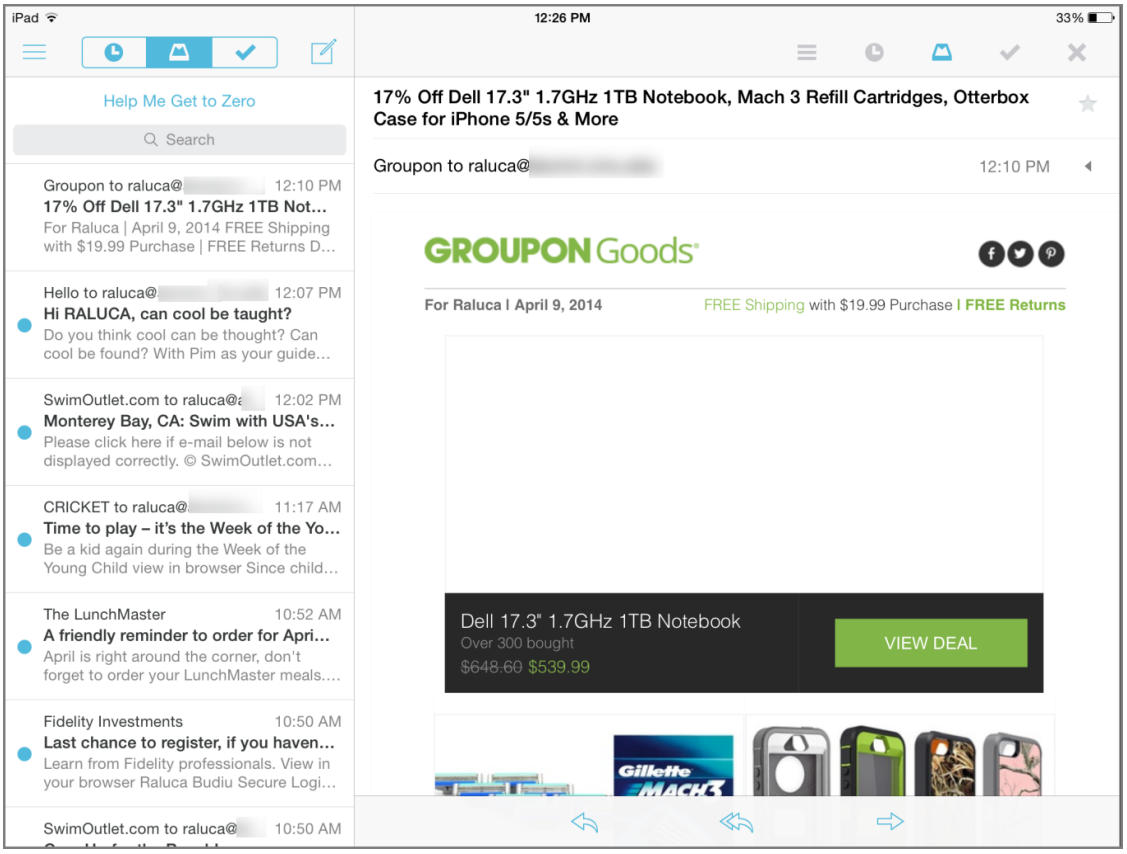

FIGURE 10 - EXAMPLE OF POOR BUTTON RECOGNITION

7. **Flexibility and Efficiency of Use** - "Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions." (Nielsen, 1995)

   This is a necessity, especially with a modern site that would ideally want day-to-day use to be flexible and efficient for a wide range of skill-sets. This means that the site caters to advanced users who can quickly navigate and use the website as well as to basic users who require a bit of guidance to get to grips with the nature of the site. Ideally, personalised delivery, especially with a social network, would ensure that users find it quite quick and easy to use without having to absorb a whole lot of technical information to use it.

   Freedawn currently has the concept of introductory Daily Habits that users will account themselves to by adding them as habits they claim to do every day. It is also provides a way to check off the completion of the habit every day. Being a daily task and also keeping in mind the advanced users, the checking off should be made an extremely fast process so that even though users have to log in and check these habits off, it can still be done in a manner of ease and with no strain on user time. If a user had to go through a long process to check off the habit completion, this would mean that particular module would become unpopular over time as users would become more and more accustomed to completing habits without going through the lengthy process of checking them off on the website.

8. **Aesthetic and Minimalist Design** - "Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility." (Nielsen, 1995)

   This essentially means that everywhere that information is provided to the user must be aesthetically appealing. It must not only keep in line with the theme of the overall site, but it must also contain relevant and minimal information. User frustration will increase if a small process such as adding a goal requires users to read a manual that is multiple pages long to achieve what should be an easy outcome.

   Progressive levels of details should be provided to users, as and when they wish to learn more information. The supplying of this information should be through further links and investigation. However, on initial display, it should not provide the user with all detailed information. There is however a limit; users should not have so many sub-links that it becomes a daunting task to learn the detailed information. Ideally, it should be just as easy to access the detailed information as it is to access the basic information.

   A guideline for this heuristic is that the site should look powerful, but the design elements should not obstruct the overall functionality of the site, which would detriment the user. A vital part of this heuristic would also be the articulation of delivery for content that the user does want to see. There is little use in creating a page meant to show a user's goal, when the main viewing window only shows the user comments or related works.

9. **Help users recognize, diagnose and recover from errors** - "Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution." (Nielsen, 1995)

   With large-scale platforms, users encountering errors is inevitable. The ability to recognize these errors and allow users to recover from them is quite an important component.

   The ability to help users recover from errors can be utilized in many areas, a common error for some sites is if pages are removed then the website may direct the user to a 404 error. A generic 404 error is as follows:

FIGURE 11 - BASIC 404 ERROR EXAMPLE

This can lead to user frustration as this is usually a default page that does not provide users with any help or an option to recover from this error other than the back button. An ideal “404 Not Found” page would be a custom page that contains the theme of the site, as well as potentially a simply delivered user-friendly error message such as “This page does not exist however maybe these pages help….”, which would provide links to commonly used pages across the site or provide a search function that may assist the users recovery into finding the accurate location of the information they want.

An example of a “404 Not Found” page that follows this heuristic, allowing users multiple avenues to find themselves and not completely throw them off the website is the Facebook example shown below:

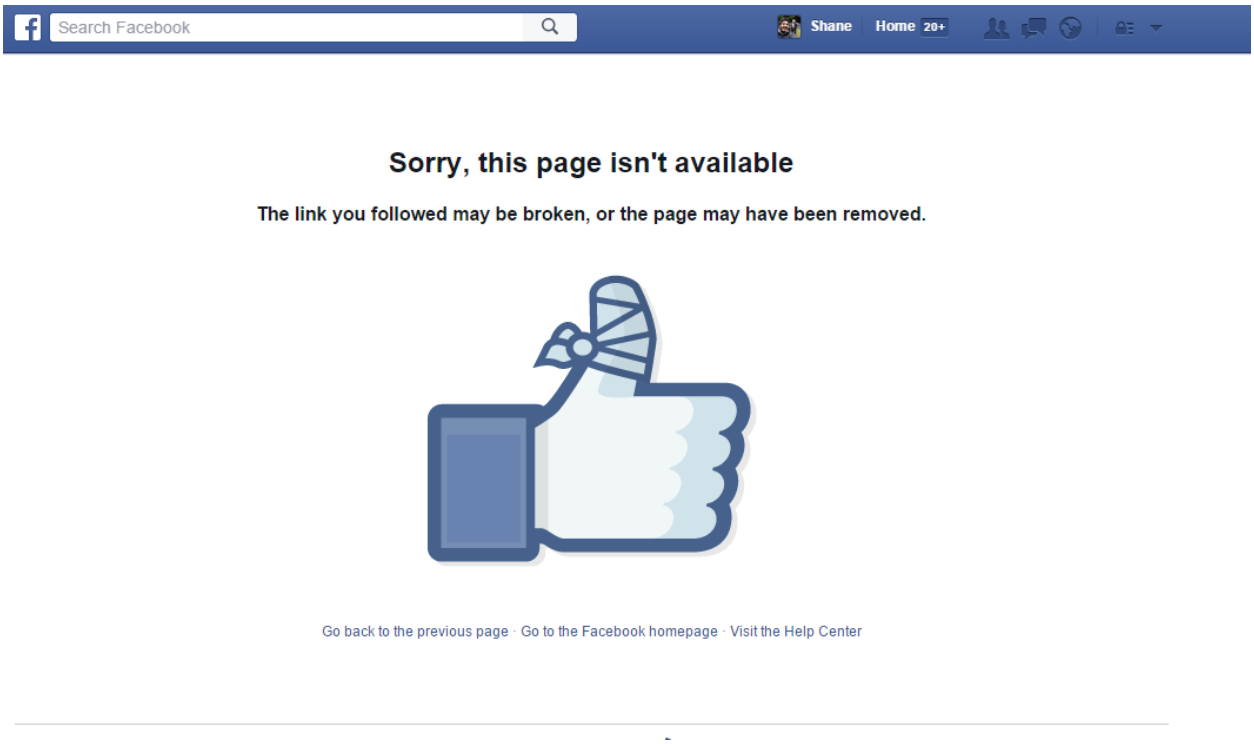

FIGURE 12 - USER-DRIVEN 404 ERROR PAGE

Another strong area of user error is form entry, as this is quite a common method of data entry on websites, users can be quite prone to error however informing them off the error is much more important. For example, a contact form where a user has to enter a numerical phone number has a letter or symbol accidentally entered when entering text into the form is submitted. An error that does not assist the user simply displays “Error - unable to send form. Try again” whereas an error that allows a user to recover easily identifies the particular area of the area and displays an appropriate error message such as “Error - Invalid phone number”. This validation and data entry can apply to multiple fields in forms and ultimately increase the overall usability of the site if applied correctly.

10. **Help & Documentation** - "Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large." (Nielsen, 1995)

    Essentially large websites, with many aspects to them, should provide a basic library of documentation with information relating to completing tasks users wish to achieve on that particular site. Ideally, this information should be easy to search for and locate, focus on the task the user wishes to achieve, list the steps of what the user needs to do to achieve the task and should not be too lengthy. This is what is ideally required to meet this heuristic and improve overall usability for a website.

    Many websites contain a Frequently Asked Questions (FAQ) section or a wiki that contains all the common tasks that are completed on the website, instructions on how to achieve the task and basic information potentially raised by users over a period of time of use or after beta-testing.

Nielsen's Heuristics links to ultimate usability by promoting the ease of use of a website and making a site user-driven and wholly user-benefit focused. The idea of a user-driven design is that the website is one that users will be able to use efficiently and effectively in order to achieve what they wish to without frustrations or hindrances in performing tasks. With Freedawn, this idea will stem as being quite important, visitor retention is essential to making a social network platform succeed. Without a returning common user base the site will be unable to succeed. The benefit of successful usability of the site outweighs the cost in terms of implementation as a usable site will result in improved performance of functionality, increased exposure in terms of increased and return traffic, and will improve the overall credibility of the site.

RELATED WORKS

As with the boom of the online social networking environment, there are many alternatives to phpFox that can be chosen. A few that stand out based on my research in terms of potentially being used are Oxwall and SocialEngine. Both of these platforms base themselves on being customizable and easy-to-adapt to your needs. The common factor with all these social platforms is the fact that they are PHP based. When I first joined the Freedawn project, the platform was already identified and had been broken down into stages to begin development. Alternatives identified show the huge market for social networking platforms where the foundations are already built.

Oxwall is a very similar platform to phpFox where it uses the same MVC model to base its core structure as well as it uses the Smarty Template engine. However, the one major factor that would cause security concerns is that Oxwall is open-sourced therefore anyone is able to download and potentially locate exploits in the baseline code and use it to exploit social networks that use it (Vadalasetty, 2003). At present Oxwall is currently being developed by a small team with limited core development. Spam vulnerabilities and options for users are extremely limited which in essence diminishes the chance of Oxwall being selected as a viable platform for Freedawn (WebSprank, 2014).

After a bit of further research looking into Oxwall the majority of content that comes as standard with phpFox is generally a plugin feature that either has to be custom developed afterwards or an aftermarket developed plugin purchased. These features include Media Sharing, Messaging and creating Social Groups. As this is quite substantial functions to a social network, as phpFox provides these functions as part of its developed package it makes phpFox a much more viable option.

SocialEngine is based on the MVC model as well, but it uses a different framework called Zend Framework. The Zend Framework is an open-source MVC framework that seems to have picked up in popularity and now has relevant development documentation. SocialEngine does have a number of advantages such as experienced developers who can custom develop new functions for the site. However, there have been a significant amount of security threats that have recently been found in the Zend Framework. In an enterprise environment and on a site where users' data and privacy must remain secure, SocialEngine would not be the ideal solution for Freedawn to use a base platform (Zend, 2015).

In comparison with phpFox SocialEngine does not also include the ability to share media as part of the basic package, the plugins must be purchased separately and individually for each type of media such as photos, music and videos. The same also applies to the Messaging system, Event Calendar and Tagging system. All of these baseline functionalities may not be a part of the Freedawn business model to be used in the website except to have the ability to implement the functions without any extra development or an added cost benefits the site in the long run as it could potentially be features used in the near future.

After comparison with the larger social networking software systems that are available out of the box we can see that phpFox is quite a viable and valid solution for Freedawn's requirements. These brief comparisons prove the business benefit that the phpFox software can bring to the platform is they decide to utilize a few more features that are common in other social networks like tagging and show that is a viable platform to continue developing on top of to further expand to suit the needs of Freedawn.

PROGRESS AND FUTURE WORK

As the scope of my project is focused on the growth and development of the Beta, I will look at ways to continually improve and focus on the usability and placement of core modules that are in progress. My project work will entail more rigorous testing and will potentially undergo usability evaluations in the future as modules come closer to completion in line with the business model.

At present, the development is in its initial stages with my progress only focusing on the layout and usability of functions determined by phpFox to suit the concept of Freedawn but as the user base grows, the ability for all the functions to be scalable will be important.

I will work closely alongside all concept development and look and research potential ways to ensure that the scalability of Freedawn will be suitable for the business model and suit the needs of my Industry Mentors as well as be academically sound.

The majority of the time from the first semester has seen time and effort being placed towards establishing a suitable development environment and understanding the business model and awaiting completion from developers. In the second semester as development is completed, I will look at focusing my project on integration of the developed modules and implementing usability solutions to further benefit Freedawn.

REFERENCES

Alexa. (2015). *The top 500 sites on the web*. Retrieved from Alexa: http://www.alexa.com/topsites

Amazon Web Services. (n.d.). *AWS Security Features*. Retrieved from Amazon Web Services: http://aws.amazon.com/security/aws-security-features/

Appcore. (2015). *3 Types of Cloud Service Models*. Retrieved from Appcore: http://www.appcore.com/3-types-cloud-service-models/

Azad, K. (n.d.). *Intro to Distributed Version Control (Illustrated)*. Retrieved from Better Explained: http://betterexplained.com/articles/intro-to-distributed-version-control-illustrated/

BBC. (2014, February 26). *Target profits plunge 46% after holiday security breach*. Retrieved from BBC News: http://www.bbc.co.uk/news/business-26358556

Kemp, S. (2015, January 21). *Digital, Social & Mobile Worldwide in 2015*. Retrieved from We are social: http://wearesocial.net/tag/sdmw/

King, A. B. (2003). *Speed Up Your Site: Web Site Optimization.* New Riders Press.

Nielsen, J. (1995, January 1). *10 Usability Heuristics for User Interface Design*. Retrieved from Nielsen Norman Group: http://www.nngroup.com/articles/ten-usability-heuristics/

Reenskaug, T., & Coplien, J. O. (2009). *The DCI Architecture: A New Vision of Object-Oriented Programming.* Artima Developer.

Rouse, M. (2008, September). *Principle of least privilege (POLP)*. Retrieved from TechTarget: http://searchsecurity.techtarget.com/definition/principle-of-least-privilege-POLP

SecurEnvoy. (2015). *What is 2FA?* Retrieved from SecurEnvoy: https://www.securenvoy.com/two-factor-authentication/what-is-2fa.shtm

Smarty Template Engine. (2015). *What is Smarty?* Retrieved from Smarty Template Engine: http://www.smarty.net/about_smarty

Vadalasetty, S. R. (2003). *Security Concerns in Using Open Source Software for Enterprise.* SANS Information Security Training.

WebSprank. (2014, June 11). *Top 5 Best Social Networking Software*. Retrieved from WebSprank: http://websprank.com/top-5-best-social-networking-software/

Zend. (2015, April). *Vulnerabilities reported against Zend Framework, and recommendations for mitigation*. Retrieved from Zend Framework: http://framework.zend.com/security/advisories/