classmate

The class application for classroom management

Interactive Multimedia Design Major Project *(2015 - 2016)*

Matthew Skinner
B00613595

PSG 1 – George Moore

Acknowledgements

University

I would like to thank everyone who had a part in helping and supporting me throughout the creation and completion of my major project.
Special thanks go to my project mentor **George Moore** for giving me guidance and contributing feedback throughout the year, especially in the stages of defining my project.

I would also like to thank **Peter Nicholl** for his continued support throughout the year and making himself available to give advice and guidance.

Other

Jeffery Way and his collection of screencasts on **laracasts.com** made the learning process for this project much smoother and easier.

I would like to extend my thanks to the team at **Fresh Made Media** and others, Ryan Forsythe, Jamie McCleave and Simon Fraser for the experience and skill set they provided me with during my placement year.

Finally, a special thanks to **Nathan Nelson** and **Ryan Warke**, who throughout the year have both been motivational, supportive and cooperative throughout our 4 years of university together and especially during the final year during the major project planning, design and development.

Contents

1. Introduction

1.1 Overview

Classmate is a web-based application, providing educational institutes with the capability to record and display data, based on student performance.
The primary aim of Classmate is to provide an interactive application, which will have the capability of replacing current means of recording student data and displaying the information. Because of this the application has the potential to continue to grow as long as there is data available to be recorded.

1.2 Contextualized

Within places of education there are many different tools for a teacher, providing them with the capability to collect data within the classroom. This data however is rarely stored together as different programs are used based on the type of data being collected (attendance, results, contact details etc.). For a teacher, accessing this data can become time consuming and impractical having to use multiple platforms to keep record of classroom information. Based on initial research of similar applications, I have found that a there is a gap here for an application that will gather and store all classroom data together and allow a teacher to review it as clearly displayed information.

When searching online and querying those who work in education I found that the number of high quality and reliable applications in this sector are extremely limited. However those applications that did met the criteria set by a school or education board still lacked the functionality of displaying more than one key area of information.

Existing Solutions:

Capita Business Services have created a product called SIMS, a collection of school information management programs in which a subscription is paid, depending on the program, and the teacher can carry out a number of tasks.
Sims Teacher App allows teachers to hold and display their student's personal information on any device the school would use (Desktop, tablets). Although limited to a specific amount of information, SIMS Teacher App is one application of a collection created by

CAPITA. These additional applications provide teachers with the ability to target more areas of classroom information.

1.3 Aim

The aim of Classmate is to provide an interactive application for teachers, which will have the capability to replace current means of recording student data and displaying the information. Because of this the application has the potential to continue to grow as long as there is data available to be recorded.

Another factor for the growth of Classmate in the future is that it's not set specifically to a single sector of the education timeline meaning that all schools or further education colleges can use it.

1.4 Objectives

To help me achieve my aim there are a number of objectives that I had to complete.

Useful and well-designed user interface:

As a teacher will be using this on a daily basis it is essential that the design and user experience are clean and clear, allowing the user to navigate the site with ease. Creating an overly complex system could be a factor that may persuade the user to stop using the application all together.

Account Setup & Login:

A teacher must be able to easily register their personal account. Once a teacher has registered their own account, using their school email address and with a password, they will be given access to their secure Classmate dashboard.

Profile Management:

Once a teacher has been added and logged in they will have the opportunity to create their profile with basic information such as their name, email address and name of their school. When new students are enrolled into the class, the user within their dashboard creates a student profile. This profile includes student contact, medical and general information. Each student is added and collected into a class and managed by their teacher.

Data Input:

Once a teacher has been assigned a class they will be able to input data based on that student. For attendance, this will be taken on a daily basis meaning that the means of input must be efficient and effective.

Displaying as Information:

Once the teacher has entered the data this will then be displayed using graphs and charts based on the most effective way of displaying the information the school now has. For attendance, the classes teacher and the Headmaster would be able to view a student's individual attendance, as well as the classes overall and average percentage.

Fully Functional Application:

Within the given time frame the application will be fully functional on an external server, which can host and support my chosen development languages.

Testing:

Throughout the development of the Classmate application thorough testing will be carried out for each function. Once completed and testing has resulted in success, development and testing will continue on to the next feature.

1.5 Scope

Time:

The final deadline for the major project is the 29th April 2016, meaning that the time between receiving the project brief and then must be divided to ensure attention and detail is taken for each process of the application's development.

Research:

Research will require creating a mood board for my design to provide users with the best possible user experience and display information in the most effective way. As well as design there is a lot of research into the development features that needs to be collected. Apart from development, areas such as potential competitors and data protection will also need to be researched.

Design:

After researching potential designs for a number of the sites layouts, Create an Account, Login, Dashboard, Profile, the design stage will run smoother as I know what design style works best for each page.

Paper prototyping will be the first stage of this process and once a final layout structure has been confirmed mock-up designs will be created.

Development:

The development of the site will be the most time consuming section of the project. This will include mark up, front and back end development of all the features needed to create Classmate.

Cross Browser Testing:

As this site is primarily, but not constricted to, primary schools the main browser generally is Internet Explorer. As more interactive programs and educational sites become available to teachers, browsers such as Google Chrome and Firefox will be used more to support them. Classmate can be used in browsers such Chrome, Firefox and Safari to allow teachers to use their preference on browser without complication or limited use.

During development I will be using Chrome as my browser of choice as many HTML5 and CSS3, JavaScript and jQuery functions are available on this browser.

Functionality Testing:

Once the site has been developed, time should be taken to thoroughly test every feature on the site. There will inevitable be areas of the site that could be refined or replaced and this time will allow me to see which areas would be practical to edit or replace.

Refinement Period:

The refinement period allows me to take time at the end of the project and ensure that all changes have been made and that the code is well structured and commented.

Cost:

The cost of this service will be kept to a minimum. To date the only expenditure for the site has been the domain name: myclassmate.xyz

The domain was bought to make the product distinct and memorable for the public and users.
For myself this allows me to create a workstation within my own web workspace, which provides a faster and more productive workflow than what is possible from elsewhere.
Apart from the domain name, there is no reason for the cost of the project to increase.

Resources:

In terms of resources I have a large variety of tools to help me create the end goal.
Throughout final year there will be areas covered within classes that will also help suggested tools, feedback and me from my project mentor and lecturers.
As well as resources from the university, there are also resources available within the web development studio I work in.

Quality:

During the past year and a half I have been working full time at a web development studio in Belfast. During this time I have come to realise the quality of work that is expected at a professional level and I don't expect Classmate to be any different.
The quality of the code behind the website needs to be at as high a standard as the site itself and this should be followed from start to finish.

1.6 Risks

When first deciding on the Aims and Objectives of this project there were a number of factors that I saw as potential risks to its completion. Throughout the design and development of the project I encountered more unforeseen risks than I initially had.

Experience:

My main concern throughout the projects development was my experience and skill when it came to back end development. Throughout my placement year I spent the majority of my time within the front end, dealing with JavaScript, jQuery and responsive development.

Timeframe:

Based on the scope of the project and the experience I had in back end development I felt that a potential risk would be the completion of the site within the given time period.

Data and Child Protection:

As this site is focused on student information, Data Protection and Child Protection are essential. By focusing my research and attention at teachers will help to avoid any unnecessary complications.

1.7 Target Audience

When researching my targeted audience for this application, the intention was always for focus to be completely given to teachers within a school.

According to the BBC, with a 2.1% increase in the population of primary schools in the UK, the number of pupils has reached its highest since the 1970s. With numbers increasing and class sizes increasing as a result, teachers should be provided with tools to ensure they can comfortable and efficiently manage their classroom and pupils.

2. Concept Definition and Planning

2.1 Idea Generation

The initial idea for Classmate was a website for teachers to hold their student's personal details in an easily accessible application. The system would allow a teacher to securely login to their profile and have access to their students and their personal details, exams and their results that are created within the application. The idea of a collective functions application came from initial research into the programs currently in use within education.

Speaking to a number of potential users working within schools and also those studying within a Postgraduate Certificate in Education (PGCE) course about targeted programs (those that deal with a specific issue or feature), I found that there was no real need for such an application.

This is when I began to expand on the idea of an overall classroom management system and started researching available programs for keeping track of attendance or grades as well as other key features. Finding only a limited number of programs I began building upon my original idea, giving the user more features within the application rather than a stand-alone feature product.

2.2 Requirement Specification

After speaking with my target audience and reviewing their opinions on my initial idea I created a set of requirement specifications. Each requirement has been categorized and numbered, each giving a short description, a rationale as to why it is needed within the project, any dependencies it may have and a priority level as to how critical it is to creating a fully functioning project.

Functional Requirements (The User):

There will be a single user group that will be using this product, the Standard User (Teacher). The following are a number of requirement examples that are essential for the product to be fully operational and allow the user to carry out their primary functions. The full sets of functional requirements are in ***Appendix A***.

Requirement	**#1**
Description:	The user must have the ability to register for their own Classmate account.
Rationale:	By registering their own account, a user can secure their secure account by using a unique email address and setting a secure password.
Dependencies:	This depends on the user having a browser no further than 4 versions back. Classmate will only offer support to this extent is there are issues.

Requirement	**#2**
Description:	The user must have a well structured and easy to navigate dashboard to manage their classroom
Rationale:	The main purpose of the Classmate application is to assist and equip teachers with the tools and features they need to manage their pupils. By creating a well-designed site structure and layout will only further the assistance provided to the user.
Dependencies:	This depends on the user having first registered their account and logged into the site.

Requirement	**#3**
Description:	The user must be able to view all of their students within their dashboard.
Rationale:	This will allow the user to quickly and efficiently access their student's information without searching through each student within the school.
Dependencies:	Requires the user to be logged into their account. If not their dashboard cannot be accessed.

Requirement	**#4**
Description:	The user should have the ability to create exams when needed.
Rationale:	Throughout the year class tests will be required to track student progress and growth in subjects covered in the classroom. The user has the ability to add an exam to the application which will later be used to display results of the pupils
Dependencies:	This is dependant on the teacher first having students added to their application. Without students added, exams cannot be created.

Requirement	**#5**
Description:	The user should have the ability to add their student's grades and personal details when needed.
Rationale:	For the user to manage their classroom they must first have the functionality to update the student's details and add grades. They can then view this within their dashboard as described in Requirement #2.
Dependencies:	This is dependant on the teacher having students in their class to edit their personal details and also having an exam added to their application which will then be able to store results for each student.

Requirement	**#6**
Description:	The user can create reminders to keep them focused on the upcoming tasks they must prepare for.

Rationale:	Having an all in one application becomes more efficient by also allowing the user to set reminders within their dashboard as well.
Dependencies:	Dependant on the user having an account and being logged into it.

System Requirements:

The system must have a number of requirements for the product to be operational. The following are a number of requirement examples that are essential for the system to be fully operational and primary functions to be carried out by the user. The full sets of functional requirements are in ***Appendix B***.

Requirement	**#7**
Description:	The system will allow a user to update their password if it is forgotten.
Rationale:	If a user forgets their password, they can request the opportunity to create a new one. This will save them having to struggle to remember their old forgotten password and continue managing their classroom through the application.
Dependencies:	Dependant on the user having an original account.

Requirement	**#8**
Description:	The system will allow the user to update their details.
Rationale:	By updating details, teachers can change their name, email address or school name quickly. This will be an effective feature when a class moves to a new teacher.
Dependencies:	Dependant on teacher first having created an account with details to edit.

Non-Functional Requirements:

Non-Functional requirements include the aesthetics of the product and other important aspects not associated with the already mentioned requirements.

Design

- A well-designed product that is attractive to all users.
- An updated look from current applications offering similar features.
- A relevant design to schools as this is the primary place of use for the product.
- Friendly and comfortable design, enticing the user to trust and use the product.
- A professional and modern design.
- Information should be displayed in a way that is easy to read and understand.

Usability

- Clear and intuitive navigation throughout the site.
- Easy to use for both users of any computing experience level.
- Should encourage users to use the product more.
- The product should be an effective aid rather than a hindrance to use.
- A teacher's dashboard can be personalized if they have an account.

Performance

- The product should give users access immediately.
- Records should be updated without delay.
- The product should be available to 24 hours a day.
- The product will be able to manage multiple schools at once.

Operational and Environmental

- The product is responsive, allowing it to be used on a wide range of devices such as desktop, laptop and tablets.
- The product will be cross browser compatible.
- The product will work across multiple operating systems.
- The product is available as a web application only.
- Updates to the application will not cause disruption to current users or features within the application.

Maintainability and Support

- Support will be available to all users by notifying the schools admin user. They will then get in contact via a contact form.

- All updates to the product will be made during off-peek hours i.e. evenings and weekends.
- For potential users requesting information about the product, a contact page is available on the landing site.

Security

- The system cannot be accessed unless the user has logged in via their school account.
- All data regarding students shall be protected.
- The product will only provide users with information regarding the school they belong to.

Legal

- The product shall comply with the Data Protection Act.

2.3 Methodology

To assist in the management of this project, selecting a methodology was important as this would help me to structure and plan my design and development schedule and allow deadlines to be set for each section of the project.
There are a wide range of methodologies to chose from, a small number of which I found best suited for my project:

Waterfall Model:
The waterfall model shows each phase of the project being completed one after another, allowing the developer to focus entirely on the set phase of the build at a time. The model is strict, not allowing the developer to continue with the build until the current phase has been fully completed. As clean and effective as the Waterfall model is, I feel that the models lack of flexibility would be more of a hindrance to the project than aid when blocks appear or scope changes.

Another version of this same model is the **Modified Waterfall Model**. This gives the project manager the flexibility to move from the current model phase and go back to past phases. I found this model to be very beneficial as when an unforeseen obstacle presents

itself I was able to return to a past phase and amend requirements or designs to overcome it.

Agile:

Agile Methodology is a more iterative form of project management, building individual components and deploying them over a short period of time. As each component is created quickly the overall production time is short which is what makes this model appealing, however there is no clear structured plan as to creating each component meaning that communication within a team would be essential when using this model.

Prototyping:

Unlike the Waterfall model, the Prototyping model is quite flexible. Each component follows phases of design, development and tested and repeated until it is then added to the main application. This iteration based model works well when the project requirements are known before design and development begin.
For my project I feel there are not enough requirements known in detail for this model to be used effectively.

Chosen Methodology:

After considering all of the available options I decided that the Modified Waterfall Model was best suited to my project, giving me the flexibility needed to navigate between phases of the project when issues appear.
As some features within the project are developed similarly it worked to my advantage to be able to cross between features, gradually creating each and building the project as a whole instead of working at each individual feature and moving on.

2.4 Paper Prototyping

The paper prototyping stage allowed for an effective and quick mock up user interface to be created for the product. To give myself an initial view of how many pages would be needed within the site I created a site map to help give me an idea of the scale of the application.
A full size version is available in ***Appendix C***.

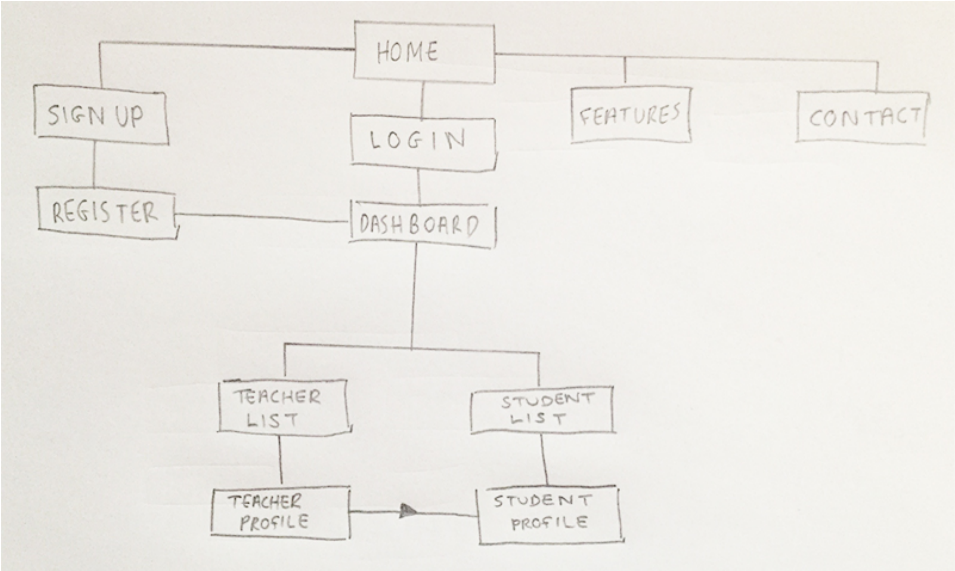


Figure 2.4.1 – Paper Site Map

The sketches are used to give an evaluation of the products idea rather than a finished presentable mockup design. By doing a paper prototype before anything else helps prevent risks appearing unexpectedly during development. The client is able to evaluate the paper mockups and provide input on the suggested layout of the application. This allows the designer to easily make small changes now rather than during the design phase which would take up more time unnecessarily.

Below is a computer built site map, refined after further development and testing of the original site map.

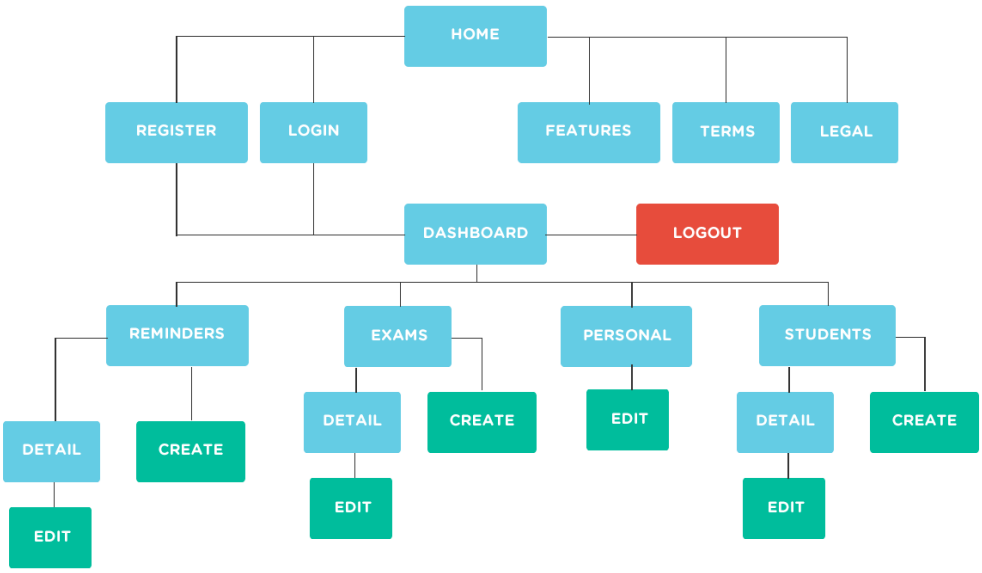


Figure 2.4.2 – Refined Site Map

6-Up:

The 6-up method includes taking a page or problem within the application and creating 6 possible solutions for it and then choosing which works best. Normally this method would be used within a team, as 6 peoples perspective on a page or problem can result in different solutions and ideas than one person could generate.

The page I have addressed with a problem in my product is the Teacher Profile page. I had originally thought that the students would be managed on a separate page, however during the paper prototyping phase I found that I would be more efficient for the user to have access to their student list from their profile.

This means that the Teachers Profile page now must display the information of the teacher, along with student's information.

To solve this issue, a small number of people were asked how they would address the issue and I created a 6-up paper prototype based on their feedback, which can be seen below along with annotations on the next page. Within each iteration the navigation Bar was pre drawn as I had decided this would stay before this process began. A full size version is available in ***Appendix D***.

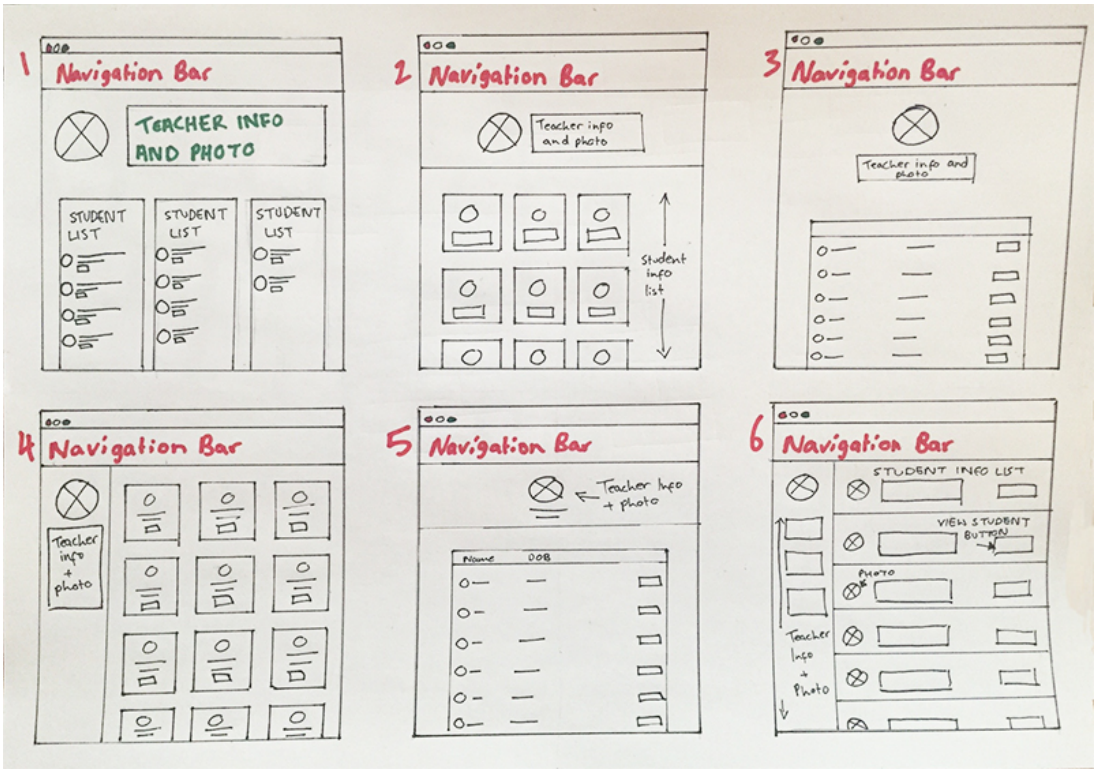


Figure 2.4.3 – 6 Up

Iteration One:

This is a very basic layout where all of the students are laid out within three grids below the teacher's information.

Although this layout would be easy to manage and develop I feel that it doesn't have the potential to look professional and would take away from the rest of the site. I feel that this iteration could benefit from a noticeable break between the teacher's information and the students information grid to give it a more appealing feel.

Iteration Two:

This iteration features the teacher's information within a section at the top of the page, clearly creating a break between itself and the student's information grid below it.

The student's information is in a simple block grid.

This would be the weakest of the 6 iterations as once the user scrolls down the page they have no easy access to their navigation, which is essential for quick navigation around the application.

Iteration Three:

This very simple layout has the teacher's information displayed centred at the top of the page. The information would be given typography structure to make this an effective structure.

The student's list below is displayed within a table, giving a clean and structured look to the page.

Iteration Four:

This design uses a fairly common sidebar for the teachers profile information and the pages content within the open right area. This immediately transforms the feel of the page, giving it more character and a somewhat more professional appearance. However I felt that this layout including the student's list was over complicating the layout for what the user would need the page for. Not enough emphasis is on this page being the teacher's personal profile.

Iteration Five:

Within iteration 5 the teacher's information has been given impact by positioning in the centre of the screen and changing the background colour of the section it's within.
The students below are within a clearly laid out table, making it easy and effective to quickly locate and read a student's information. This idea was created by combining sections of different iterations and bringing together, what we believed to be, the unique and effective aspects of them.
The finished iteration looks both professional and easy to navigate as each section of the page is clearly displayed.

Iteration Six:

In this iteration the teacher's profile information is again within a sidebar, displaying the teacher's photo and using the whole sidebar to display information. The student's information is listed, using the entire width of the remaining page to display the information.
In both the teacher and student's areas I cannot see how either will have enough information to fill these areas and make use of the space that has been given. There would be a lot of white space left and this could leave the page looking bare.
I again feel that because the teacher information is within the sidebar it would again lose its emphasis of the page being the teacher's profile.

1-Up:

After creating the iterations from the 6-up phase I took some time to review and consider each design. The iteration layout that was decided on was iteration 5, as this was constructed from a group of people pulling a collection of ideas together to create a clean and efficient layout for the teacher's profile. This iteration removes problems discovered in other iterations such as unnecessary white space, a hierarchy of information and structure within the page.
With the iteration chosen, some addition thought and detail went into the layout of the page using the 1-Up method.
I found the 6-Up and paper prototyping phase of the project to be very beneficial and resulted in creating a solution to a layout problem within the application. A full size version is available in ***Appendix E***.

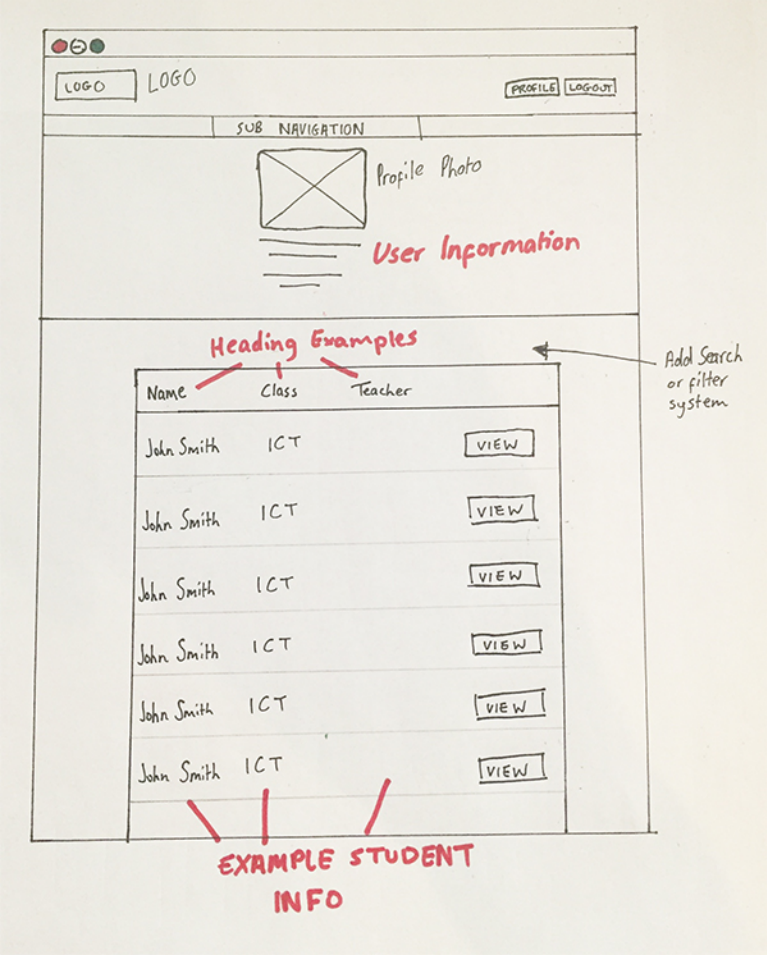


Figure 2.4.4 – 1 Up

2.5 Feasibility Testing

Project Risks:

As the project progresses and the paper prototyping had come to an end, an important stage was to access the potential risks that may occur within the project.

By creating a risk level assessment table, the risks are rated using the Traffic Light Rating System. Assigning a threat level, Green, Amber or Red, is done by multiplying the risks impact level by the likelihood of the risk occurring within the project. By scoring each of these a number is given and placed on the table.

A key for this system is displayed within ***Figure 2.5***. This helps to discover which risks are immediate threats to the completion of the application and which can be monitored without actions being taken. If a risk reaches a critical red level, development should be halted until the risk has been dealt with.

A few examples of such risks would be:

- The external server may crash, causing the product to be unavailable for an extended period of time, which is out of my control.
- The development machine may crash, causing all project files to be destroyed and lost.
- Users account is not secure and data can be accessed from an external source.
- Workflow stopped due to learning new development languages.

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

The full Traffic Light Rating System has been created and is displayed within ***Appendix C***.

3. Design

The application relies on having a design that can be implemented within each page. The use of a well-designed layout and navigation, as well as typography and colours, all contribute to the users experience within the application and will encourage them to continue returning to the product to use it.

3.1 UX Design Evolution

As I began the process of turning my initial 1 Up sketch into a wireframe I was reading an article on Medium, *Growing as a Designer*. In the article, Luke Jones states how design is much more than its face value. When designing it's not about solely on what looks nice and exciting but on what works and what are the appropriate methods to use when designing. *"Form over function is never the right approach"* Jones states, and with review and consideration for the 6 Up method I had previously created I decided to rethink my chosen layout.

As important as a well-designed site is, this also means that the user, regardless of their technical skill or experience, should have no issues when it came managing their

dashboard. The user should be able to navigate the site without hesitation as to where a function would be located or how to view exam results, giving them a confidence in the product they are using.

When showing the initial UX design to a number of people the main issue many raised was the sites navigation. It had been designed to give a different approach to the standard dashboard experience, however after consideration and reading the Medium article I rethought my approach and began to research into current dashboards users would currently use and which would make them feel most comfortable using the application. *“Asking users to adopt new behaviors or even modify their existing behaviors is very, very hard.” – Khol Vinh of Mixel*.

Based on this research carried out and conversations with those how may potentially use the application, the decision was taken to base the products layout on the current design trends of sites (social networking and shopping sites) that users would use on a daily basis, allowing them to fell comfortable when they use this application for the first time. As Classmate is not a social networking site the dashboard content would rely on designs based on the data collected and making an informed decision on the best design practice to implement these into the site.

Site Wide Design

Throughout the site a use of flat design and colours will be used to immediately give the product a place within current design trends. The use of skeuomorphism within application design has radically decreased, with flat design becoming the default theme, for example the updated operating systems for Windows and Apple have made a transitioned to using flat design within their more recent OS versions. Following this design trend will contribute to giving the user a sense that the application is well maintained and updated to fit current design trends.

The flat design colours where inspired by using flatuicolors.com and also Adobe Color CC. By selecting my primary colour from the Flat UI Colors site I was then able to create a colour palette after using a number of colour rules such as Triad, Complementary and Shades.

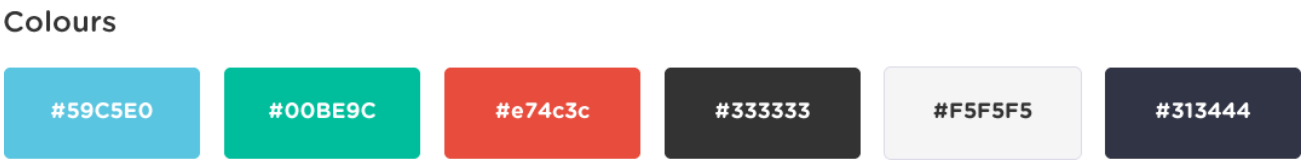


Figure 3.1.1 – Colour Scheme

Typography

When researching options available to me for adding fonts to my site I decided to use Adobe Typekit, paying for a year subscription. Typekit has over 5000 fonts which allowed me to choose fonts that were unavailable anywhere else without paying a large amount for it. I searched to ensure that I would not be able to get the required fonts elsewhere without using Typekit before paying for it.

The typography used throughout the site will be Gotham Rounded, which gives an easy to read, modern appearance to the site. My initial choice of font was Proxima Nova Soft as it had the same rounded shape to each letter. The use of the rounded letters made the site look more appealing and child friendly, as the application would be used within schools. However, when it came to added content to designs I found that Proxima Nova Soft didn't fit the appearance I hand imagined and planned to portray. Searching for similar fonts, Gotham Rounded was the best match and while testing the use of the font family within Style Tiles, shown below, it became the primary font used within the site.

Style Tiles

The style tile was created to help design visual mockups of the Classmate application. Creating this tile would gather the collected fonts and colours together and give an earlier visual representation of how they will work collectively within the application.

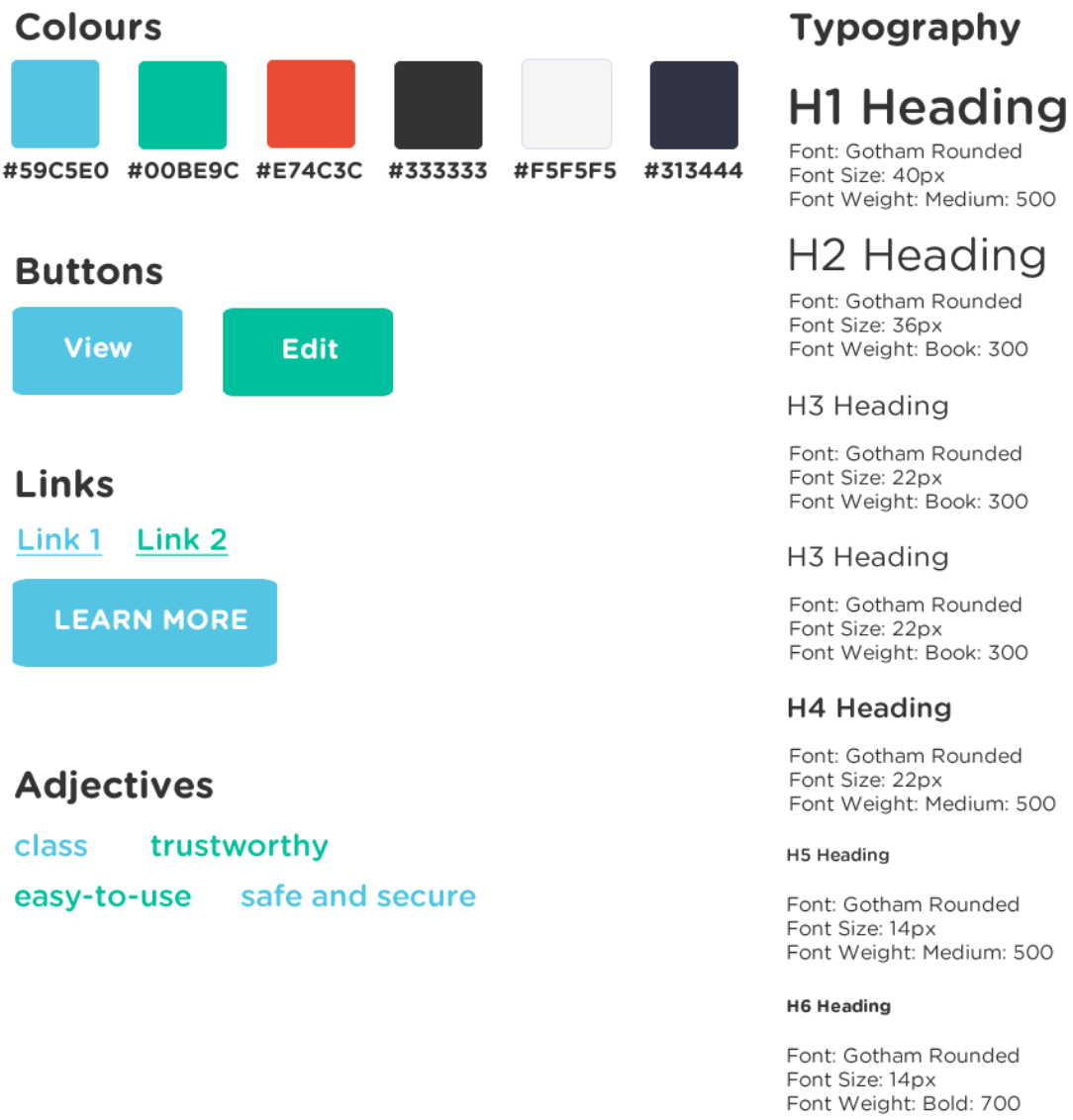


Figure 3.1.2 – Style Tile

Icon Sets:

In recent design trends the use of icons within navigation and careful consideration throughout sites has become second nature to most designers.

Within Classmate, icons will be used across the interface and carefully selected to ensure that the icon best matches the link or section of the application that it represents. With many icon sets being used within operating systems, such as Apples iOS for iPhone, icons are no longer used for only complementing navigation.

However, for Classmate icons will be used along with text within the navigation and as a visual aid for the headings or text it is representing.

To select the icon set I would be using within the site I reviewed a number of sites that produced free to use icon sets. As expected many of these were not at a high enough quality for the product I wished to create and others did not match the design that Classmate would take.
Font Awesome is a site that allows user to download a wide and expanding font set and freely use for commercial use. As all of the icons are controlled with CSS rather than JavaScript, the style such as weight and colour would be easily changed depending on its use within the site.

As well as using Font Awesome for icons within the Dashboard of the application there was a need for icons in the promotional pages of the site. These would be icons used to help promote and explain the benefits and the features available to users by using Classmate. I felt that the icons available to me using Font Awesome were not suited to this method of use and so research into Swifticons began. The icons available from Swifticons had the ability to be edited in programs such as Sketch and Adobe Illustrator, allowing me to edit specific fonts and adding the theme colours, which can be found through the site into the icons.

Figure 3.1.2 – Swifticons

3.2 Paper Prototyping

3.2.1 Mind Map:

During the initial sketching process of the application I developed a Mind Map to collect all of the different thoughts and ideas I had for creating this product. By doing this I had all of the different possible options in one place and when reviewed I could take a number of the best ideas and expand upon them in further sketches.
This can be seen in ***Appendix D***.

3.2.2 Users Journey:

Based on the Mind Map, as seen in ***Appendix D***, and implementing these ideas into the initial sketches of the site, a users journey diagram was created. This diagram would outline the users movement throughout the application and also help to towards building a site map, which is shown in ***Appendix D***.

3.2.3 Initial Sketches

After the reconsideration of our design after the 6-Up sketches, new ideas and layouts were sketched out for the main feature pages of Classmate. The various aspects of the site such as login, adding a student, inputting results and displaying results required some sketch attempts and shown in ***Figure 3.2.1*** is the thought process for them through a number of initial sketches with the rest of the sketches within ***Appendix D***.

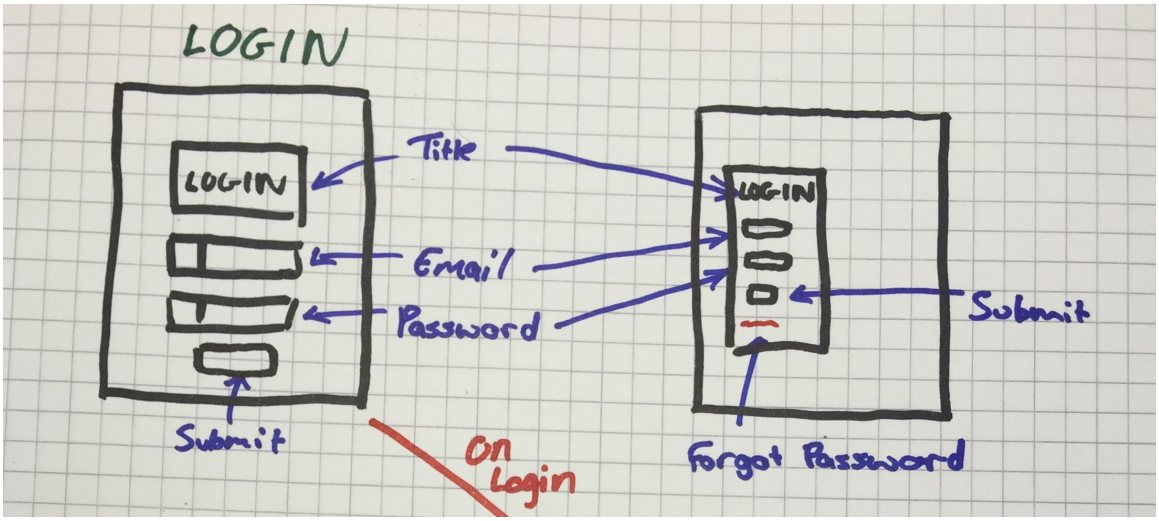


Figure 3.2.3 – Initial Sketches

3.2.4 Refined Sketches

After the initial sketches had been drawn I took the same sketches and spent time refining them, adding more detail to the functional areas of the pages. At this stage I added a small amount of colour. As I did not have the exact HEX value pen I limited the use of blue, green and red to a minimum to guide me in further designs, which would be computer based and allow the correct colour codes to be used.

Below in ***Figure 3.2.4*** are two of the refined sketches created, with the rest of the refinements available in ***Appendix D***.

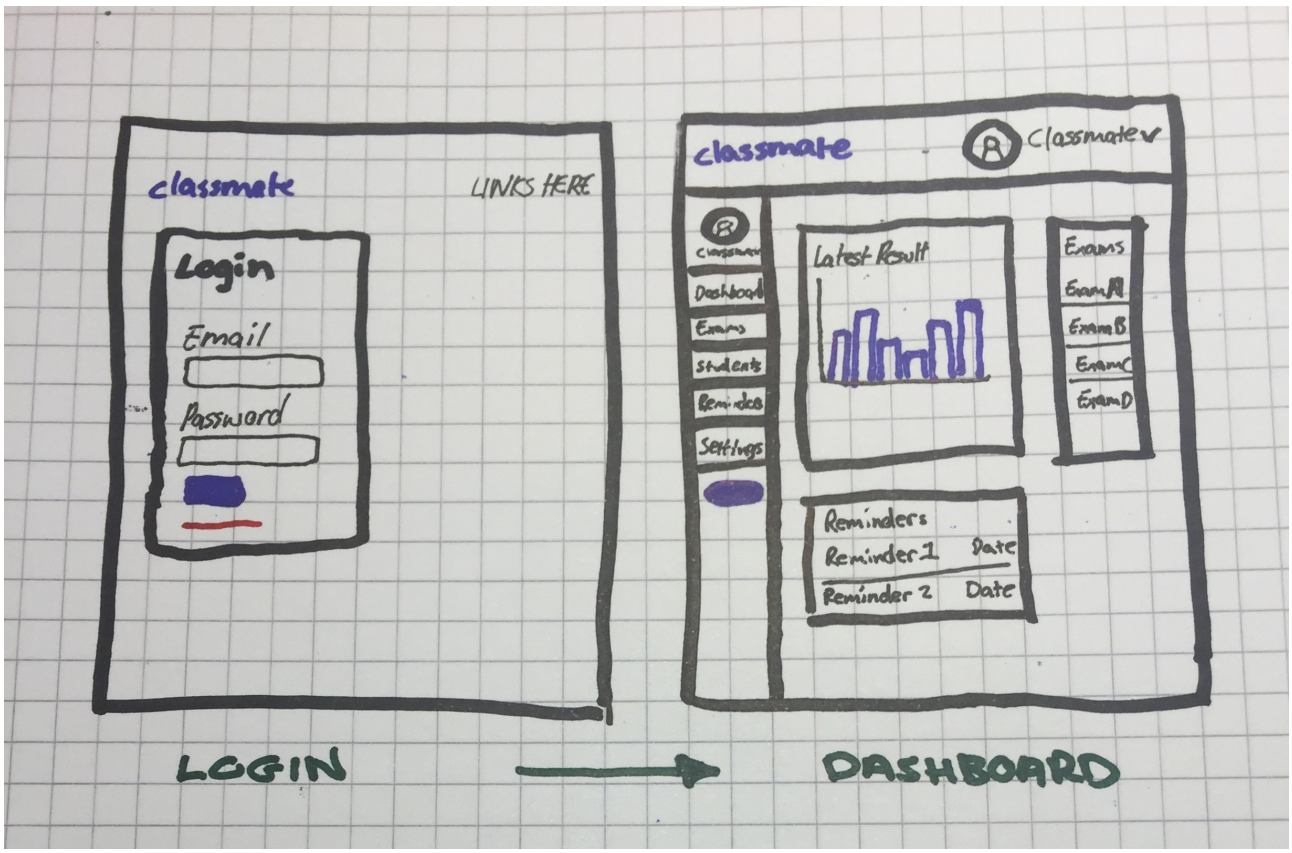


Figure 3.2.4 – Refined Sketches

3.3 Designed Mockups

Taking the initial sketched designs and turning them into mockups was an enjoyable part of the design process. By following the requirements, system design, mind maps and refined sketches, creating mockups was a much simpler task to undertake.

Homepage:

The homepage is the first page any visitor or potential user will arrive at, meaning that there is as much an emphasis on good design here as there is within the users dashboard area. The homepage offers an insight into the design style of the application, as well as offering key information on the features and usage of the product to attract people to use it.

From the homepage users have a number of links available, allowing them to navigate to the features page, register or login and also read information the legal information from the footer.

Design inspiration was taken from the Laracasts site and also from dribble shots. From Laracasts I particularly liked an icon based tile background, which gave me the inspiration

to create a tiled background for Classmate. Basing the tile design on education, a suitable tile from Subtle Patterns was found and with some editing I was able to create the tile needed.

The full home page design will be available in ***Appendix D***.

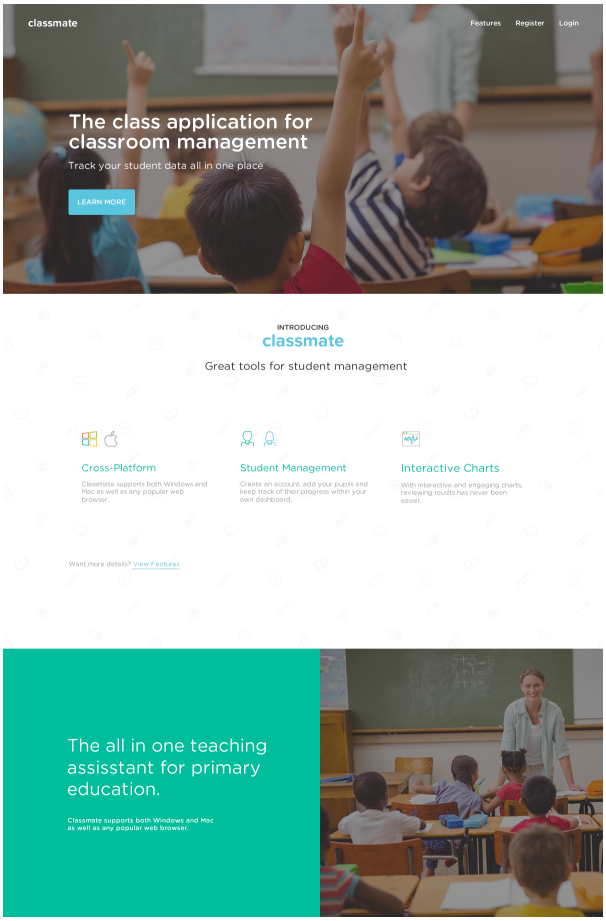


Figure 3.3.1 – Homepage

Navigation:

There are a number of forms of navigation within the Classmate site for both the landing site and within the application. From the home page there are two navigation options available; the header navigation items located at the top right of the site and also the footer navigation options.

The header navigation is kept to a minimal number of links. When a user visits the site they will see options to move to the features page, which holds more information on the available features of the site, a page to register for an account and also a page to login. When a user has an account already and has recently logged in, the navigation options will switch, now giving access directly into their dashboard due to the recent login. After a

period of time without visiting the site, the user will automatically be logged out of the application and the original navigation will be displayed again.

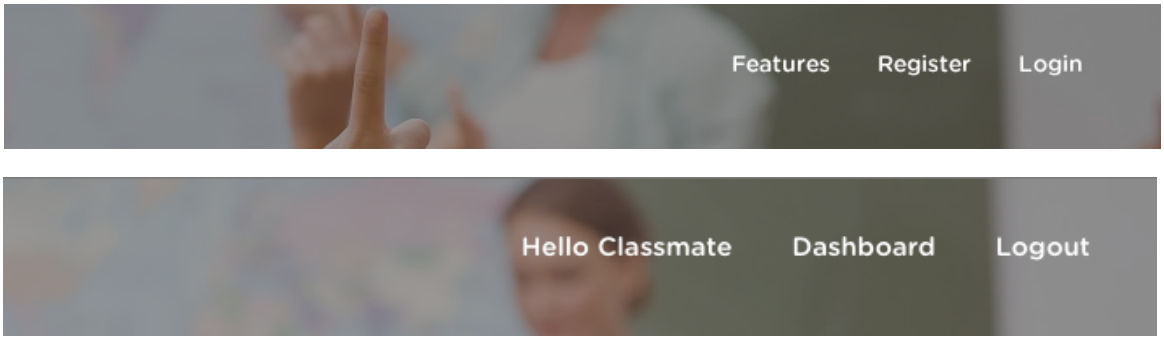


Figure 3.3.2 – Homepage Navigation

Within the footer the user is given a number of options, some being the same as the header have been duplicated to save the user having to scroll to the top of the site again to request a new page.
Additional links have been added for legal information such as the Classmate privacy policy and the terms and conditions of the site. From researching other sites such as Laracast and Dribbble this information is only ever linked within the footer. On the Facebook site the legal navigation options are only available outside of the dashboard area. Leaving the dashboard navigation to focus completely on the application. This is a navigation method I plan to implement into Classmate.

Figure 3.3.3 –Footer Navigation

Within the dashboard section of the site the primary navigation changes from a header to sidebar. This design change is due to the current and popular design trend within dashboard related applications. Having a fixed sidebar navigation allows users to constantly have easy access to each feature within the site. The links within this new navigation now only direct the user to pages within the application and not back to the homepage.

Within the header is a small secondary navigation. As mentioned before in regards to the layout of the dashboard, users feel more confident and comfortable using systems with recognizable navigation. This navigation allows the user to access the settings of the site and to also logout of their dashboard, redirecting them back to the homepage.
Also within the header secondary navigation is the logo in the top left of the page. This will direct the user to their dashboard page, as is the common practice of most sites.

Figure 3.3.4 – Laracasts Navigation

Figure 3.3.5 – Facebook Navigation

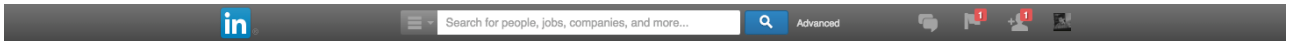


Figure 3.3.6 – LinkedIn Navigation

Having multiple navigation options for the user is convenient as the typical web user is impatient and doesn't want to overthink the layout of a sites navigation. With consistent navigation throughout the application and implementing good navigation structure the user should feel confident when using Classmate. The resulting navigation is displayed below in ***Figure 3.3.7*** and ***Figure 3.3.8***.

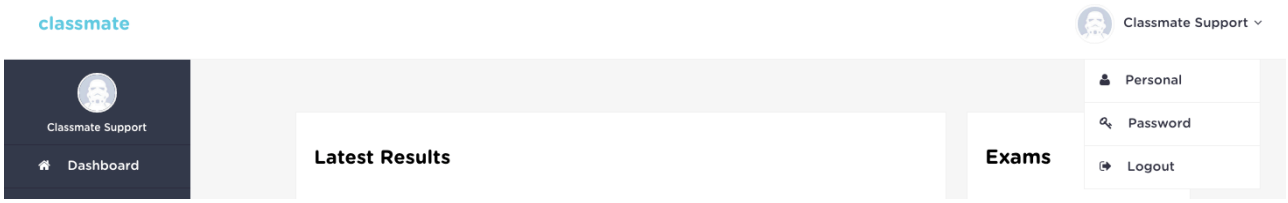


Figure 3.3.7 – Secondary Navigation

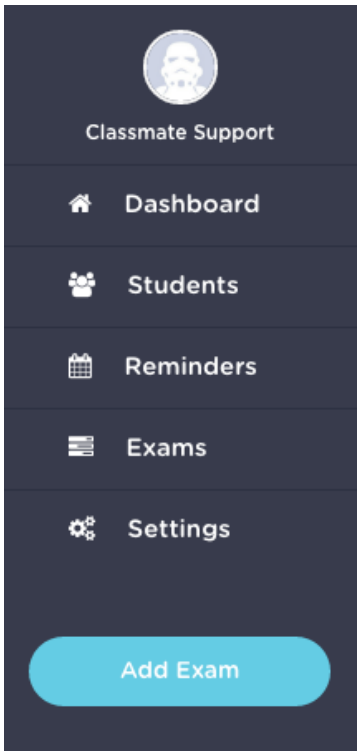


Figure 3.3.8 – Primary Navigation

Dashboard:

When a user logs into their dashboard the first thing they should see is their most up to date and relevant information. By gathering information from a number of the sites features the user can immediately be updated on current results, exams and reminders created previously.

This dashboard has been separated into 3 sections, the primary section being where student's results from the most recently marked exam are to be displayed. Secondly is the exams sidebar, displaying each exam the user has created, giving quick and easy access to this area of the application, again, preventing the user from becoming impatient by to the exams page and then locating the required page. The final content section within the dashboard is the reminders. The user can quickly keep track of any tasks they have set and navigate to the specific reminder for further details.

The hierarchy of content works well within the content area beside the sidebar and allows for clean scrolling when needed without manipulating other sections of the application. This content area remains the same for each page, giving the product uniform and structure and presenting itself as a professional application. The dashboard is shown in ***Figure 3.3.9***.

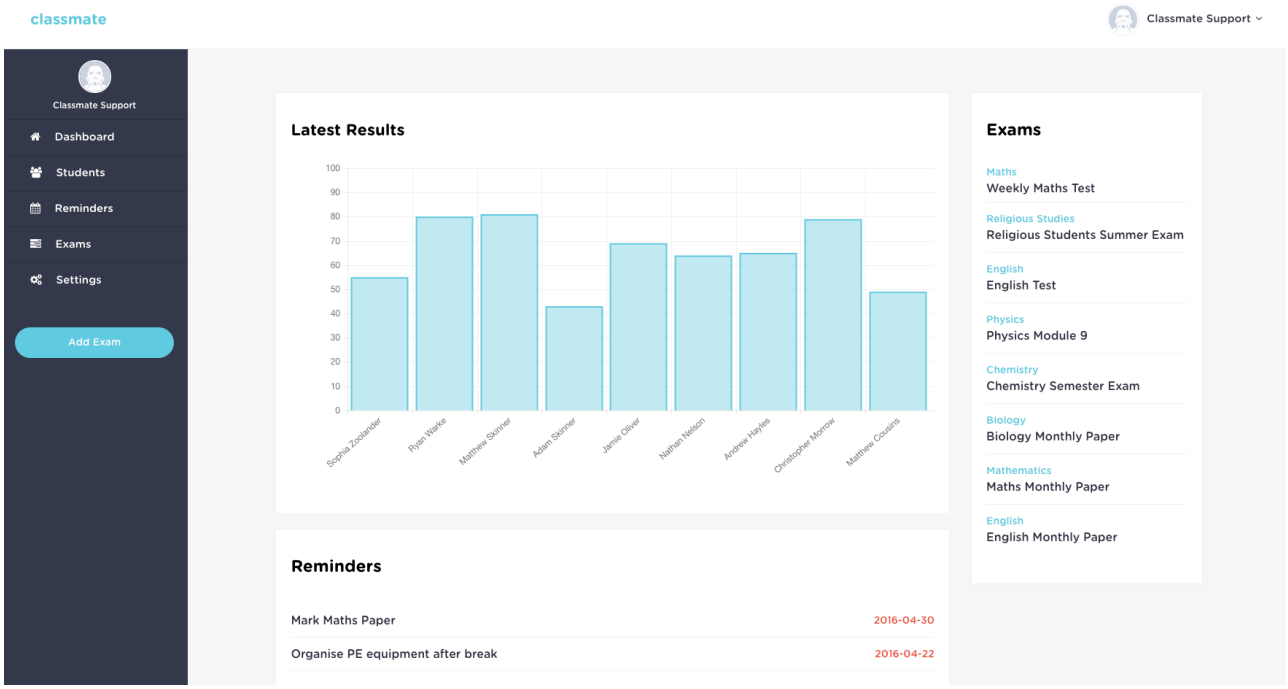


Figure 3.3.9 – Application Dashboard

Many other mockups were created for the Classmate application, displaying the different aspects of the site and areas that the user can interact with. As per the style tile in ***Figure 3.1.2*** the site is presented in a clean, modern and flat design based user interface.
The rest of these mockups can be found within ***Appendix D***.

3.4 Branding

While creating a brand, consideration of the site as a whole is required to create a persona for the application. This persona makes this application unique from any other, creating an identity around the product, which will become recognizable to users and those considering which product to use.
A strong and memorable brand can be the difference between a user selecting Classmate over another.

Logo

Initial thoughts towards the logo of the site were to be a friendly, clean and minimal single letter representing Classmate name. This was inspires by many large companies such as Facebook, WordPress, Beats and Tumblr using single letters to represent their product.

Figure 3.4.1 – Brand Logos

From these examples created a number of different logos for Classmate, using the letter C as the centre of my design. Using a single letter heavily relies on the typography and colours used to make the logo unique and recognizable. Using my style tile in ***Figure 3.1.2*** I used the styles available to create a number of options as shown in ***Figure 3.4.2***.

Figure 3.4.2 – Classmate Logos

After taking these examples and placing each into context within a site design I found that all seemed to be lacking in anything memorable that would entice a potential user to find out more about the application.

Out of the 5 possible options the fifth was closest to anything that would be used as a way of representing the site and therefore it was chosen as the favicon for the application.

As a font family had already been decided and the colour selected to keep the logo in uniform with the site, the only area to tackle for the logo was the weight of the font used as shown in ***Figure 3.4.3***.

Figure 3.4.3 – Classmate Logos Refined

After consideration the final logo to represent the Classmate application would use the font of Gotham Rounded, a font weight of Bold (700) and use the flat design blue, which is the primary colour within the site. The final logo is displayed in ***Figure 3.4.4***

Figure 3.4.3 – Final Classmate Logo

On the home page and authentication pages of the site, the logo appears over a dark background and the use of the primary colour doesn't blend together. As shown in ***Appendix D – Designed Mockups*** the logo has been altered to white along with the navigation in order to make it visible and stand out in front of the background overlay.

3.5 Responsive

With the advancement of technology and the programs available for students as a learning aid, schools are becoming more technically advanced within the use of laptops & tablet devices being integrated into the school curriculum. Due to this and the rapid growth in mobile browsing a large, if not the majority, of the targeted audience would be using devices smaller than a desktop computer to access the internet from with a school.
Based on this, Classmate was to have a responsive design with 3 breakpoint levels – desktop, laptop and tablet.
Mobile browsing of this site will, to a certain extent be supported, however with the use of charts within the dashboard that may potentially hold up to 30 students, mobile browsing will not be able to do justice to the large charts required. The sites 'landing site', consisting of pages such as the home page, features, legal and terms and conditions will be supported for mobile, allowing users to get an introduction to the site and see the features available from any viewing platform.

To achieve the fluid layout for the application, percentages were used for declaring widths instead of a pixel value. By doing this, a page may be re-sized and the content will follow the change in width due to the percentage being taken from the viewport width.
As content begins to compress and distorted, media queries are introduced into the style. These media queries allow new style properties to take the place of those for the same

element once the viewport reaches a set device width. The content width values are increased and this is repeated for lower media queries. The manipulation of the grid layout will continue to provide the user with an accessible interface with familiar controls as shown in ***Figure 3.5.1***.

Figure 3.5.1– Responsive Design

The navigation of the site will be replaced with a mobile menu dropdown once the viewport width reaches a screen size lower than an iPad.

As there are now such a wide variety of screen sizes for tablets and mobile devices, trying to accommodate all with individual media queries would be near impossible. The best practice is to implement style and layout changes, which will accommodate all device widths within set media queries.

When choosing an icon to best represent the mobile navigation I researched current popular sites and their use of the menu icon.

Sites such as Facebook and Twitter use apps for their mobile and tablet presence primarily, following a layout of positioning the menu as fixed at the bottom of the page and primarily using icons as shown in ***Figure 3.5.2***.

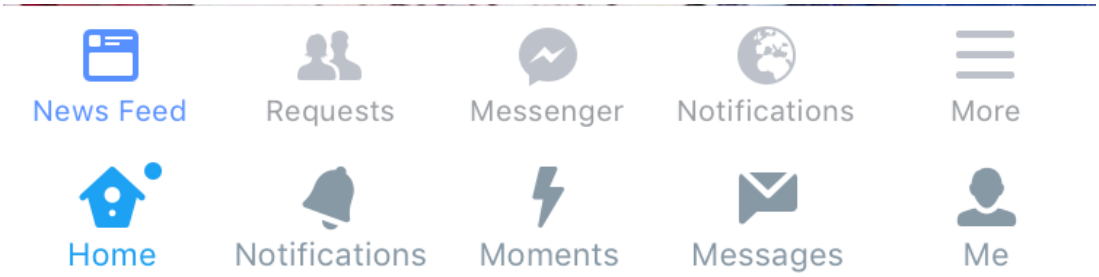


Figure 3.5.2– Social Mobile Navigation

Moving from social media to product based sites I found that mobile navigation was generally located at the top right of the page and on click showing a dropdown of the available pages to move to.

The menu icon is generally 2 or 3 lines above each other, giving the appearance of a list which mobile users have become familiar with. Maintaining this method of representing the navigation would be applied within the Classmate site as shown in ***Figure 3.5.3***.

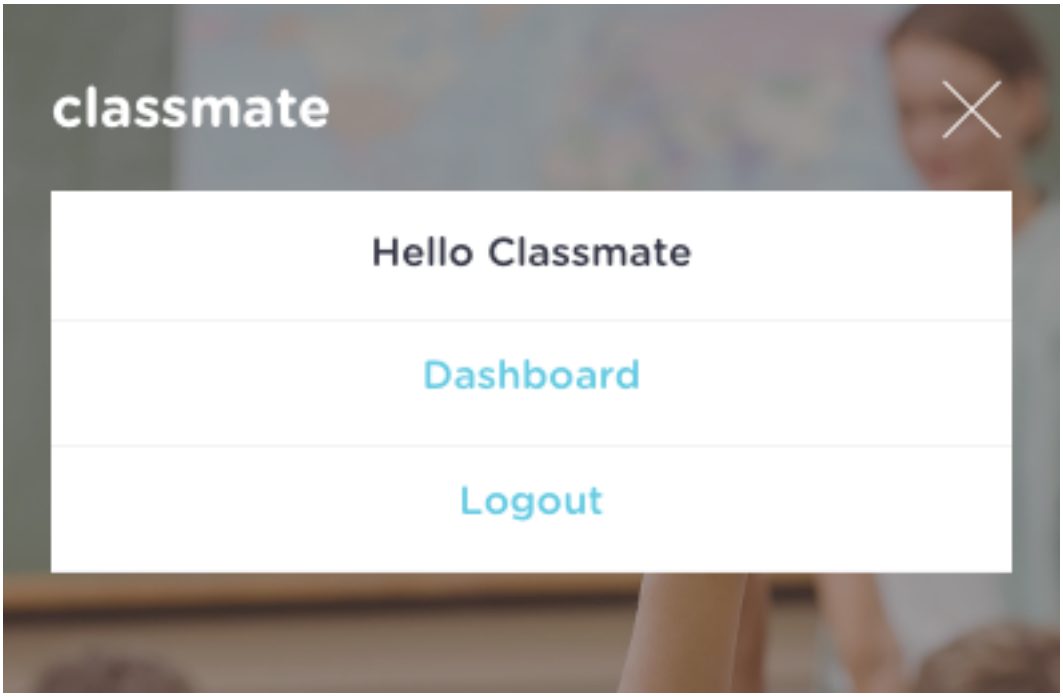


Figure 3.5.2– Classmate Mobile Navigation

3.6 System Design

The system design of any project consists of an in-depth review of how the application will function on the web and the technologies used, both client side and server side, to bring it

to life. These areas cover a wide range such as database design, to the development framework used to build the product.

Client Server Model:

The Client Server Model shows the relationship between client side and server side components. The chosen client and server components are those that I have had experience with in the past, progressing the initial setup and development of the application.

Client Side components used for the Classmate application are:

- HTML5 was used as the sites markup language.
- SASS was used for styling and was then compiled into CSS. There is a use of CSS3 transition elements throughout the site. The framework used for the site was based on Bootstrap, providing many variables and elements with style and structure already.
- jQuery was used as the products scripting language and used to create many features throughout the site. bxSlider was a downloadable jQuery package downloaded and used within the homepage of the site, creating a clean and minimal slider. Chart.js allowed the creation and implementation of interactive charts through the HTML5 element canvas. Using this JavaScript resource provided a wide range of possibilities for displaying results throughout the site.
- The displays/devices used for developing this project were a 2011 iMac with secondary Samsung monitor, a 13 inch Macbook Air and OS X El Capitan as the operating system. To test the responsive aspects for the application an iPad running iOS 9.3.1 was used. For users, all devices and operating systems will be usable.
- The site will be compatible with all modern browsers.

Server Side components include:

- PHP is used as the server side scripting language for the application.
- The development framework used is Laravel 5.1.
- The database management system used is MySQL.
- The shared hosting server is hosted by Big Wet Fish

3.7 Logic Design

Creating a logic design before development of the site began gave a clear understanding of the application and navigation. Each logic step shows where the user will interact with a form of functionality within the site and use logic to determine what to do in the scenario.

#1 Sign up using register form

When the user submits the required data within the registration form the data is validated. For unique input fields such as the users email address, the authentication model is searched to ensure no other address is the same.
The password the user adds is encrypted and stored within the database.

#2 Login using login form

If the user already has an account with Classmate they will fill out the form within the log in page. When the form is submitted the email address is again checked against know addresses within the authentication model. If there is a match the input password will be checked against the requested email address.
If both the email and password match the stored data, the user is granted access to the dashboard.

#3 Directed to Classmate dashboard

Once registered or logged in, the user is redirected to their personal dashboard. Within the dashboard, information from the exam, results and reminders models are requested and displayed for the user to view recent information. This includes the latest results marked, exams created by the user and reminders created by the user.

#4 Navigating to the Students List

Once authentication has been verified, the user will be able to navigate to the students list page. Within this page, the student model will request all students data created by the user.

#5 Navigating to the Students Details

Once authentication has been verified, the user can navigate to the students list page. Within this page, each student added will be available to click and direct the user to this students details page.

#6 Add Student using add student form

Once authentication has been verified, the using has the ability to add students to their Classmate account by navigation themselves from the students list page to the add a student page.

By filling out the form and ensuring that all required fields have been completed and submitting, the students details are added to the database and linked to the user via the user_id field. This field takes the users id and creates a relationship between the students details and the user who created them.

#7 Edit Student using edit student form

Once authentication has been verified, and the user has successfully added at least one student to their dashboard, the user can navigate from the students list page to the student details page. From here the user will be able to then redirect to the edit students details. The student model collects the requested students current details and inputs them into the form inputs.

Upon submission of the new form the model searches for any changes within each input and if any are for they replace the current data.

Each section follows a similar pattern of logic design, allowing the user to easily navigate from one section to another and request details views of each student, exam, results, student or reminder.

Shown in ***Figure 3.7.1*** is a use case and logic diagram, representing the users movement through the site.

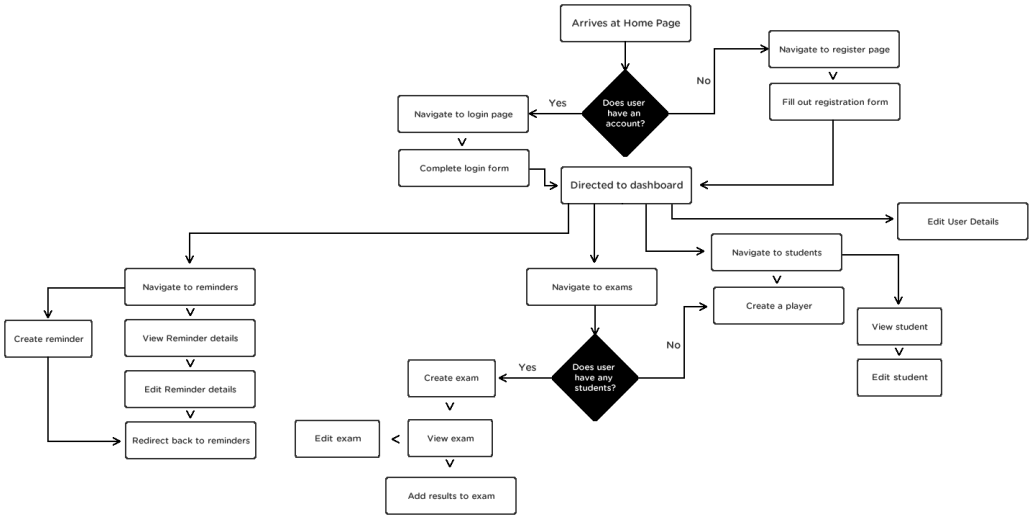


Figure 3.7.1– Use Case and Logic Diagram

3.8 Data Design

Data design is how the data input by the user is stored within the application database. With consideration before designing the relationship models, no changes were needed.

The application will use MySQL as I have had past experience using it within the university and also during stages of industrial placement.

In ***Figure 3.8.1*** the initial database relationships are sketched, showing which tables will be linked to others and what name the link shall be given.

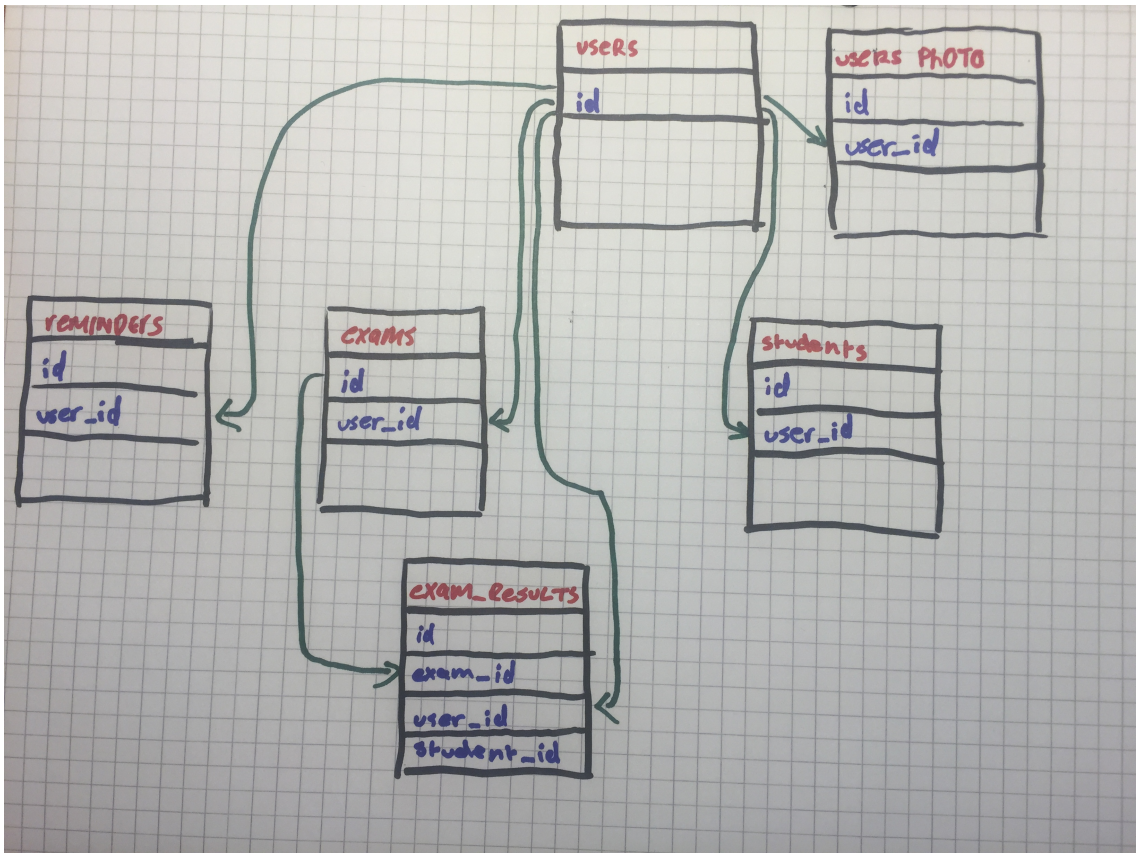


Figure 3.8.1– Database Relationship Sketches

From this initial sketch the data design was developed. Each entity required added to its model. Each entity was given a name, type and requirement status.

Shown in ***Appendix D***, ***the*** data has been added to each table in a way that allows the developer to easily locate and understand relationships between tables by carefully naming each entity.

4. Implementation

4.1 Technology Selection

Throughout the creation of this project a wide range of technologies were used, both on the client side and on the server side including scripting and mark up languages, databases, hosting packages, front and back end frameworks and external libraries for implementing key features into the site.

4.1.1 Server Side Technologies

Server Side Scripting:

When it came to selecting a server side scripting language to use for my project I found it to be an easy choice. As I prefer to focus on Front End Development I was able to remove options such as Python and Ruby as I would have to learn these languages from the beginning. As the option to learn these new languages would not fit within the given timeframe, my options were between using PHP or Node.js.

Node.js is a relatively new and lightweight language. I found this appealing as it allows for writing server side scripts in JavaScript.

PHP (PHP:Hypertext Preprocessor) is an extremely well known and widely used scripting language for web development. Using PHP would greatly reduced the risk of using a server which does not support the language as almost all servers now do support PHP.

I decided to use PHP as my server side scripting language. As I will discuss later, another factor in my choice to use PHP was the use of the Framework Laravel. Having previous experience with Laravel I found this helped direct me towards using PHP as my Server Side Scripting language of choice.

Server Side Framework:

During the early thought process for the project I had in mind to use a Model View Controller (MVC) framework as I would not need to learn from scratch, having had previous experience with this form of framework in the past. The MVC framework chosen without hesitation was Laravel. I have used this during my placement and have found it to be an effective and easy to use framework, simplifying PHP and has plenty of available resources available such as Laracasts.

The use of the built-in blade template and integration with Bootstrap would also lead me to use it over frameworks such as CodeIgniter.
Apart from the framework itself, the resources available for Laravel are extremely well detailed. Having subscribed to an account on **laracasts.com** I have found that each screencast is extremely detailed and helps me to save a vast amount of time looking for solutions elsewhere.

Hosting:

When looking at hosting, the option was available to use the university servers to host my application. Choosing this method would ease any worries of malfunction affecting my site due to it being the universities responsibility to maintain working servers.
However the consequences of using these servers would be the security measures and restricted permissions that could severely restrain the Classmate product from functioning properly.
Another option, which was available to use, was local Belfast based server provider Big Wet Fish (BWF), who advertised special offers for final year students looking to externally host their major projects. With experience dealing with BWF during my placement, I had first hand interaction with the company in regards to setting Laravel based websites up within the server and server support when issues arose.

The chosen hosting was a shared hosting plan through BWF. This server had the ability to support PHP scripting language and the team set up the Laravel framework quickly. Other advantages include not being restricted by the security measures that the university requires, and if permissions need changed a support ticket will be responded to in good time with a response.

Database:

I chose to use the most popular database system for my project, MySQL. The system is very fast and easy to use. Within the application there is the potential for hundreds of teachers and thousands of students to be added to the database. Where some database systems may have issues with this, MySQL will continue running reliably on a server.
When choosing to use Big Wet Fish to host my application, the chosen package included MySQL databases, which could be managed from within my hosted cPanel account.

This combination created a manageable development environment and easy access to the MySQL database used for Classmate.

Node Modules:

While developing my application there were a number of external JavaScript libraries that were needed to improve the workflow and efficiency of the project as well as libraries used required is key features of the site were to be implemented. The list below details a number of the node modules used within the application and how they are benefiting development.

- **Chart.js** was installed using *npm install* to allow the chart functionality to be used within the site. Without this, results would not be displayed within graphs throughout the site.
- **Gulp** is used to automate and enhance the workflow by compressing SASS files into CSS and optimizing assets. By using the command “gulp watch” within the terminal, each save of an assets file will be automatically compiled and compressed without constantly having to use the “gulp” command.
- **Gulp Autoprefixer** is used to track issues when saving and compiling assets such as SASS and JavaScript. If there is an issue within any of the asset files the Autoprefixer will display an error message and location of the error within the Terminal.
- **Gulp Load Plugins** automatically loads any gulp plugins requested into the package.json file. This save manual editing of the file and creates a more efficient installation flow for future plugins.
- **Gulp Minify CSS** allows the app.css file to be compressed and minified into app.min.css. This process removes unnecessary blank space and can dramatically reduce file sizes.
- **Gulp SASS** allows the style to be written in Syntactically Awesome Style Sheets and when partnered with **Gulp**, offers a new and more efficient styling method.
- **Sweetalert** is used to provide clean and animated notifications to the user when requested.
- **Vue** is a new JavaScript library, which creates Vue instances with HTML and JSON. Perfect when dealing with charts.

4.1.2 Client Side Technologies

Client Side Scripting:

Markup:

When selecting a language for markup, there was no research for any possible contenders over HTML. When selecting which version to use I decided that it would be beneficial to use the most recent version to create the application, HTML5.

I found the use of new elements such as <header>, <main>, <section> and <footer> to be extremely useful for markup, clearly representing the different sections of the site and its structure.

As Laravel was used as the projects back end framework, markup was different than standard .html files. Using Laravel's blade file, this powerful templating engine compiles the files into php code.

Styling:

Like HTML, I again immediately began using the most recent version of CSS as my styling language for the project, CSS3. With many new styling techniques available through this version I made use of them when necessary throughout the application. Instead of writing CSS, it was compiled from SASS. I have found that using SASS to write style is much more efficient and understandable than CSS, allowing code to be nested like HTML or jQuery would also be. The combination of using SASS nesting and the gulp watch command within the Terminal created an extremely fast and easy to use style set up which made development much more enjoyable throughout the creation of the application.

Scripting:

JavaScript (JS), being the most widely used language, is the front end scripting language I am most familiar with so it was used for this project. JS will run in the browser and allows elements within a web page to be manipulated and allows for a large amount of flexibility as to what can be done within a web page.

There are a number of libraries available, one of which is jQuery that can also be used, simplifying the writing of scripts and allows tasks to be performed easier. With these libraries and the use of JavaScript / jQuery within the side opens the flexibility and possibilities for each webpage to a wide variety of potential use.

Bootstrap:

Using a framework for any website offers the developer a wide range of additional functionality which can be used to help meet the users needs. Having had experience with a limited number of frameworks I decided that I would use one that I was familiar with.
The Bootstrap Framework used for the Classmate product was edited and development during my time within my placement. The new version of the Bootstrap Framework, nicknamed Bowman, was built around the Laravel Framework as this was the primary server side framework used for any project, meaning that the two work very well together.
As well as working with Laravel, the framework is responsive, allowing the application to work across a number of platforms and devices as required.

The development of the Classmate application would not have been possible without the use of these incredibly reliable, effective and flexible languages and technologies.

Browsers:

Being a web application, Classmate must be viewed within a web browser. With a large number of browsers available for use there are a select few that stand out immediately from others.

- Google Chrome
- Mozilla Firefox
- Safari
- Microsoft Edge
- Internet Explorer
- Opera

From studies gathered by Craig Buckler, the website Site Point released an article at the beginning of 2016 on the latest browser trends.
Based on these studies the most popular browser by a substantial rate of 53.71% was Google Chrome. Followed by Internet Explorer (all versions) at 15.16% and then closely followed by Firefox at 14.29%.
As the majority use of the IE browser is from users in China and Greenland it was not the primary browser for development and testing, as the application would not support the browser versions being used or have language support either.

Browser support the Classmate site would cover each browser stated above as well as up to 4 versions previous as development of the site cannot rely on the user constantly having their browser up to date.

Displays and Devices:

As the application would be viewed within a browser, this means that any device capable of accessing the Internet could be a potential viewport for the product. As designing and developing for all of these would be both a waste of time and effort a select group of popular products were chosen to base all design and development for. These main devices consist of a Desktop computer, a laptop and an iPad tablet.

Although Classmate is primarily created for these viewports due to the main use of the site being steered towards schools and places of education, where the use of mobile devices by teachers is not encouraged within the classroom, users may explore and investigate the site with a mobile device and use the applications dashboard on a larger viewport.

4.2 Tool Selection

Code Editor:

Throughout the 4 years experience of code editing I have used a number of different code editing programs. The initial editor of choice and for FTP would be Coda 2, allowing the user to edit files from a remote server by logging in.

However as the use of SASS was introduced I found it disappointing that there is no reliable support within the Coda 2 program to compile SASS into CSS. Gradually the use of Coda has become a tool for making small and quick text corrections to a file on the server or transferring files between the computer and the server.

The code editor, which was introduced to me during placement, is Sublime Text 2. As this is the most essential tool for a developer I was very impressed by Sublimes ability to effortlessly install new packages, themes, syntax highlighting and coding aids for any scripting or mark up language needed. With additional support for SFTP able to be installed I was able to store all development files on both my local machine and also on the server. Uploading a file when it was edited and saved.

I found the use of split screen to be incredibly useful, having a blade file open on one half of the screen and style on the other.

Project Management:

To help manage the Classmate application and ensure that all I was meeting set deadlines within the assigned timeframe I used the Trello web app. Trello creates boards for the user to add reminders, tasks, checklists and due dates to and help to track progress throughout projects.

I found this to be a good way to motivate myself to continue through development within a section of the site so it could then be checked off the Classmate Trello board. The Classmate Trello board is shown below in ***Figure 4.2.1***.

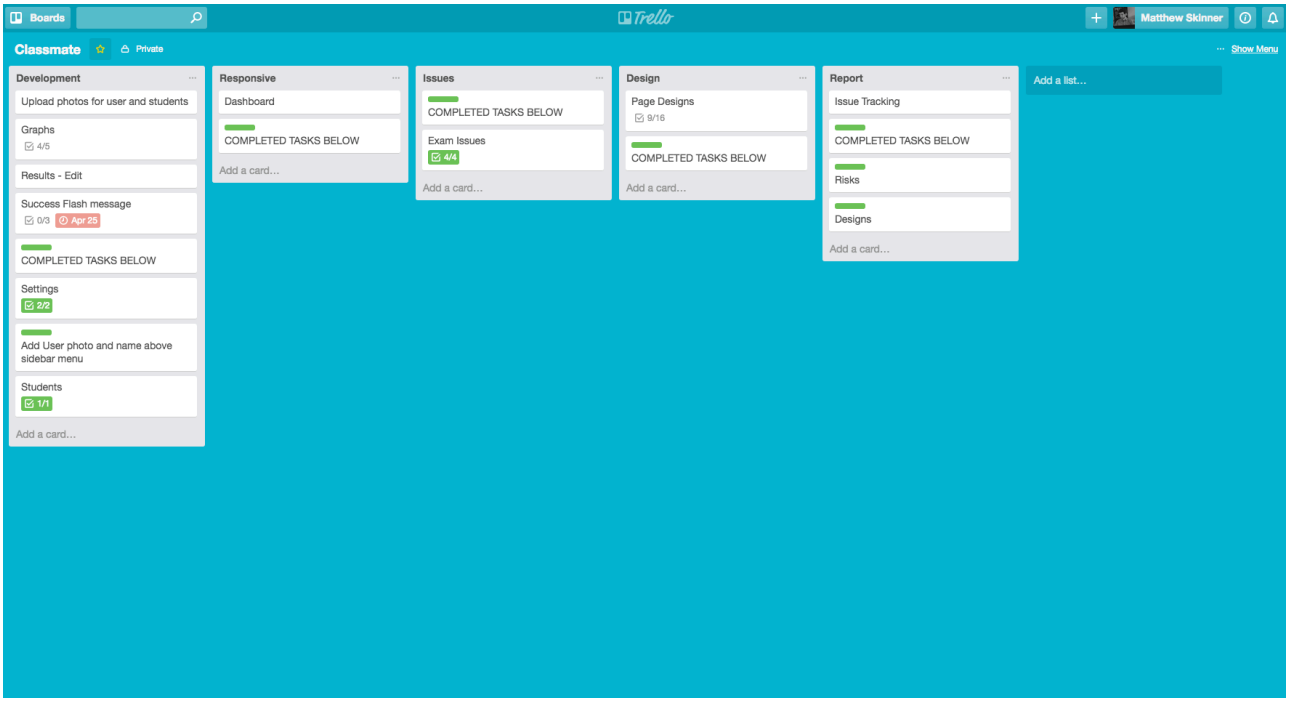


Figure 4.2.1 – Trello Board

Hardware:

Throughout the entire development of this application, everything was created using Apple's OS El Capitan on both an iMac with dual display on a Samsung monitor and a Macbook Air. The majority of development was carried out on the Macbook Air, which was connected to the second monitor due to its speed and easy navigation using the track pad. Both devices had the same programs and tools installed meaning that development could switch seamlessly between them. For testing the application within Internet Explorer and Microsoft Edge a Windows OS laptop was used, as these browsers are unavailable on Mac OS.

Backup:

Throughout the development process a number of project backups were created and stored as a new feature was added to the site. The latest backup would be stored on the server within a compressed ZIP folder along with the current live version of the site, a copy of the current site on both hardware devices and finally an archive of backups stored on a Samsun 1TB External Hard Drive.

Command Line:

For many tools that were used within the development of the site were installed and run through the command line. A number of examples where the command line was used would be:

- Using gulp watch to compile SASS into CSS
- Using the npm install function to install required node modules
- Monitor saved files for errors and display them using the node module gulp-autoprefixer.

Design Software:

I used a range of Adobe programs for the design of the application. Photoshop was primarily used for editing images and experimenting with colours and branding.
Illustrator was used for editing icons taken from the Swifticon set.
The original website designs were created using Fireworks as I found the tools and control given within the program allowed the initial sketches to be made into design mock-ups. However, as Fireworks is not the best practice to be using for web design I took up the program Sketch and found that it was an extremely effective tool for design.

4.3 Notable Challenges

Throughout the development of the application I faced a number of challenges, some of which were more challenging than others.
Although I had some experience with Laravel before beginning development on Classmate the relationships between controllers, models and view was still to be grasped. The most challenging area of the Laravel development was the use of database factories and migrations, choosing which fields were to be required and also setting any fields which were to take their data from another table.

After the Laravel development issues had been dealt with the next issue was for a key component of the site. The ability to add exam results for the students within the users class. As each student needed to be stored within a separate row in the database table, each needing to be contained inside a separate form, shown in ***Figure 4.3.1***.

```
39          <button id="submit-trigger" class="btn btn-primary" onclick="submitAll()">Add Results</button>
40          <span id="processing-results">Processing results, please wait...</span>
41
42          <script type="text/javascript">
43              function submitAll() {
44                  var formCount = $('form').length;
45
46                  var numberCount = 0;
47
48                  console.log("form count " + formCount);
49                  console.log("number count " + numberCount);
50
51                  $('form').each(function () {
52                      var that = $(this);
53                      console.log(that.attr('action'));
54                      console.log(that.serialize());
55
56                      $.post(that.attr('action'), that.serialize());
57                      numberCount++;
58
59                      console.log("test");
60                      console.log("number counting: " + numberCount);
61
62
63                  });
64
65                  if (numberCount == formCount) {
66
67                      function your_func() {
68                          window.location.href = "/exams/<?php echo $examID; ?>";
69                      }
70
71                      setTimeout(function() { your_func(); }, 5000);
72
73                  }
74              }
75          </script>
```

Figure 4.3.1 – AJAX form submission

After storing the results data, the next challenge I faced was converting the stored data into JSON and displaying it within the chart.

Once this had been completed, the next stage was to use the relation created by the student_id to display the student's full name within the chart label rather than their ID.

4.4 Notable Achievements

An overall achievement would be the completion of the product and the growth in development, design and project management throughout the set timeframe. This was done by identifying the best frameworks and JS libraries and extensions to use and combining them to create the application as it now stands.

Authentication:

The authentication process was made very easy with the use of Laravel 5.1 as it was pre built within the framework already, along with other features and functionality. As this was the first site, which I set up authentication for, I view it as an achievement that there were no critical bugs during its setup and development.

Creating Results for an Exam:

The AJAX request used JavaScript to count the number of forms that are on the page. As the number of students within the class may vary so must the number of result inputs. The function created counts the number of forms found on the page and on form submission, each form is processed until the number count matches the number of initially counting forms. On the submission button click it is disabled and a message appears, explaining to the user that the results are being processed.

Displaying student name as chart label:

The data for the chart is collected as an array and by using chart.js, can use “json_encode” to change it into a string of JSON data.

The same had to be done with the students full name by creating a string variable of both the first and last name of the student and added as the charts label shown in ***Figure 4.4.1***.

```
85
86      <?php
87          $student_name = '[';
88          foreach ($results as $key => $value) {
89
90              $user = DB::table('students')
91                      ->where('id', $value->student_id)
92                      ->first();
93
94              $student_name .= '"'.$user->fname.' '.$user->lname.'"';
95
96              $student_name .= ',';
97          }
98
99          $student_name = substr("$student_name",0 , -1);
100         $student_name .=']';
101     ?>
102
103     <script type="text/javascript" src="/js/chart.min.js"></script>
104     <script>
105
106         (function() {
107             var ctx = document.getElementById('latestExamChart').getContext('2d');
108             var chart = {
109                 labels: <?php echo $student_name; ?>,
110                 datasets: [{
111                     fillColor: "rgba(89, 197, 224, 0.4)",
112                     strokeColor: "rgba(89, 197, 224, 1)",
113                     pointColor: "rgba(89, 197, 224, 1)",
114                     pointStrokeColor: "#fff",
115                     pointHighlightFill: "#fff",
116                     pointHighlightStroke: "rgba(89, 197, 224, 1)",
117                     data: <?php echo json_encode($scores) ?>
118                 }]
119             };
```

Figure 4.4.1 – Display name as chart label

Restricting a user to data added by themselves:

When development on the application began an area for concern had been the ability to secure a users data so they were the only user able to few their students, exams, results and reminders.

With tutorials from Laracasts and the use of the Laravel framework this was made simple and works without any issues. Within each of the required controllers the following lines of code was used as shown in ***Figure 4.4.2***:

```
19      public function __construct()
20      {
21          $this->middleware('auth');
22      }
23
```

Figure 4.4.2 – Protected routes

This protects the routes by only allowing it to be access by authenticated users. When setting requirements for the data, which is to be gathered on the page functions page, one requirement is that the user_id found in the Students table must match the id of the authenticated user currently making the request on the site.

Meaning that the data is unique to the one user. This is shown in ***Figure 4.4.3***:

```
25
26      // Students
27      public function index()
28      {
29          $userID = \Auth::user()->id;
30          $students = Student::orderby('lname', 'asc')
31                      ->where('user_id', '=', $userID)
32                      ->get();
33
34          return view('dashboard.students.index', compact('students'));
35      }
36
```

Figure 4.4.2 – Where user_id equals users ID

5. Testing

The testing section of the application development focuses on using a range of different methods to evaluate the status of features against the requirement specification created at the beginning of development.
To do this, the box testing method was to be introduced along with cross browser testing and cross platform testing.

5.1 Testing Approach Selection

White Box Testing:

This testing approach reviews how the application preforms different unit tests. Mainly focusing on the functionality of the site at a code level. Security, inputs and outputs flow within the product are main areas where this testing approach can be used to an advantage to produce a good quality product.
The white box method helps to catch bugs early before further development takes place.

Black Box Testing:

The black box testing approach is opposite to the white box method. Rather than dealing with code and examining the site through its controllers and views, testing is carried out within the browser, testing the applications performance, usability and other non-functional areas of the site.

Grey Box Testing:

Grey box is a combination the white and black box methods, allowing testing be carried out both by those who have a knowledge of code and the inner workings of the application and those who don't who are able to follow the black box method alongside.

5.2 Methods

Static Testing:

Without running code or testing the site within a web browser, the application can be tested by carefully reviewing code for each section of the application. This involved ensure that layout and structures were created using concise methods, with minimal elements to prevent confusion and the increased possibility of style rules being overwritten.

As for assets such as SASS and JavaScript, a structure was created following the users flow. Global style and scripts would added first and gradually move from the home page through to each function of the dashboard.
The main aim of this method of testing is to catch any obvious bugs before browser based testing commences. This helps to improve the overall quality of the code and also development skills by catching bugs within using the console log or issue tracking.

Dynamic Testing:

Once functions begin to run within the browser, some unexpected errors will begin to appear. Within the Classmate application, an unforeseen error originated from the submission of the results form.
Once the user submitted the forms the page was immediately redirected, meaning that all of the forms within the paged did not have time to post their input data to the selected table. When the results graph was displayed only 4-5 students results would be displayed. By console logging this issue the problem was discovered and by setting a delay on the page redirect each form was given time to post and be displayed within the results chart.

Cross Browser Testing:

During the development of the Classmate application Google Chrome was used as the browser window for viewing and initial testing of the site. Chrome was chosen as it is by a large margin, the most popular web browser to date, is regularly updated, providing new features for developers and allows for informative insight into errors that occur throughout development within the browsers Inspector Element console.
However, Classmate would need to work on all used browsers rather than limit the target audience to only Chrome users. Each available browser that would be supported, listed in **4.1.2 – Client Side Technologies**, would be downloaded and tested. Testing includes style meets the designs created for each browser and the features of the site are functional for each browser.

Cross Platform Testing:

The Cross Platform testing was heavily design centred as other tests were covers during the cross browser-testing period. The devices used during this testing period were an iMac, Macbook Air, iPad, Windows tablet and iPhone 6.

The browsers tested across the tablet and mobile devices were Google Chrome and Safari as for mobile browsing these 2 browsers were the most popular. Similar tests from the cross browser tests were carried out on these devices.

5.3 Results

Fortunately I was able to have both of these methods carried out during the entire development period. With friends currently studying and preparing to study for a PGCE these users were the ideal test users as they would be experienced with using the internet and managing a profile from social media. While using the features as they were developed, the testing users gave me feedback on how they expected features to work and how they expected information to be displayed.
There were no significant changes needed from the testing users feedback on the use of the site.

For development testing results were promising with initial speed tests averaging 1.5 seconds per page. Making a user wait any longer than 2 seconds on a page loading fully would be an issue as the attention span of the average web user is extremely low and will leave the page if it has not loaded quick enough. The site also allows the users journey through the site to be fluid and uninterrupted by predicting issues that the user may run into, for example an exam cannot be given results unless the user has first added students to the application. By trying to navigate to the add results page the user will receive a message explaining the necessary requirements and a link to create them.

Overall there were no major functionality issues recorded throughout the testing process by users. A number of bugs were discovered and quickly dealt with. The application benefited from having non-developers demo the site and comment on features, design decisions and usability.
The users all gave positive feedback on the application and were impressed by the use of charts throughout the site,

6. Evaluation

6.1 Testing and User Evaluation

Each user involved in the testing of the site was asked to give honest feedback on the products design, usability and functionality and then to give suggestions for potential improvements that could be made to the application.

As mentioned within **5.3 Results** the general consensus on the site was encouraging and positive. Comments describing the site as well designed, easy to use and understandable were positive indications that the requirement specification for the site was being met.

The box testing methods were chosen as this process could be carried out during the project development, meaning bugs could be fixed when first discovered and usability changes could be implemented when needed.

6.2 Project Outcomes

The main outcome of the project was a fully functional application, which met each of the requirement specifications found in ***Appendix A***. As well as meeting the requirements set out initially, the application can now also fulfill its purpose and fill the gap in the market as it was intended to do.

6.3 Methodology

As mentioned in **Section 2.3**, the Modified Waterfall Method was used for this products development. This methodology worked well for my site and the development path followed very closely to the chosen methods.

Following this method allowed development to return to previous stages of a feature to fix bugs when needed rather than wait until development and testing was completed before doing so. This also worked well with the box testing methods that were used, so when a bug was discovered it was immediately dealt with.

7. Conclusion

7.1 Summary

The aim set was to create a classroom management application and a number of objectives were set along with a suitable methodology to guide the planning, designing and development process.

From this stage, requirements were created, giving detailed descriptions on what each section of the application should do.

From the requirements, initial sketches were created and refined. Designed mockups were then created. Creating the user interface included taking into consideration the users flow and journey throughout the site.

Once designs were completed and finalised development began. Throughout development there was continuous testing taking place for usability and catching bugs early.

All of this came together to create the final product.

7.2 Project Reflection

Having the freedom to chose scale and criteria of the university major project ensured that I would enjoy the topic selected and be determined to see the aim and objectives of the application met. From selecting the projects topic the scale was then assessed.

Speaking with the major project mentor assigned lead me to focus on developing a fully functioning feature within the application rather than trying to add multiple and finishing none. I found this helpful advice and stood by this throughout the design and development stages. With careful planning and time set aside to learn new required skills, the project was able to be completed within the set time frame.

7.3 Self Review

Reviewing my self-performance on this products design and development journey, I feel that I have grown over the process, learning the importance of time management and planning. The roles covered within this time frame included project manager, user interface designer, front end and back end developer.

Beginning the process of planning the product was an area I did not have experience so reflecting on the journey taken I feel that the journey has helped to grow management skills and qualities.

Following the Modified Waterfall methodology helps a great deal during development of the site and helped to guide me from one task to another.

7.4 Future Improvements

There are many features and ideas that were not added to the application at this stage due to the tight time frame and development involved in creating the site as it now stands.
Over the coming months the next objective of the application is to begin planning and designs of possibly new features and implement them into the site.
Other possibly objectives would be to create an iOS application for Classmate and further the development skills gained from creating the product so far. This would require learning new scripting languages or programs such as Xcode or PhoneGap.

Whatever method is chosen I am excited to see where the Classmate application goes from here and that the skills developed from this project management experience are grown and improved.

References

Capita, SIMS Teacher App, SIMS Discover & SIMS InTouch:

SIMS Management Information Systems from Capita SIMS | Capita SIMS. 2015. SIMS Management Information Systems from Capita SIMS | Capita SIMS. [ONLINE] Available at: http://www.capita-sims.co.uk/. [Accessed 23 October 2015].

SIMS Teacher app | Capita SIMS. 2015. SIMS Teacher app | Capita SIMS. [ONLINE] Available at: http://www.capita-sims.co.uk/products/sims-teacher-app. [Accessed 23 October 2015].

SIMS Assessment Manager and Reporting - Primary Schools and Academies | Capita SIMS. 2015. SIMS Assessment Manager and Reporting - Primary Schools and Academies | Capita SIMS. [ONLINE] Available at: http://www.capita-sims.co.uk/our-products/sims-assessment-and-reporting-suite-primary-schools-and-academies. [Accessed 23 October 2015].

SIMS Discover for primary schools and academies | Capita SIMS. 2015.SIMS Discover for primary schools and academies | Capita SIMS. [ONLINE] Available at: http://www.capita-sims.co.uk/our-products/sims-discover-primary-schools-and-academies. [Accessed 23 October 2015].

SIMS InTouch for primary schools and academies | Capita SIMS. 2015.SIMS InTouch for primary schools and academies | Capita SIMS. [ONLINE] Available at: http://www.capita-sims.co.uk/our-products/sims-intouch-primary-schools-and-academies. [Accessed 23 October 2015].

Target Audience:

Primary school pupil numbers soaring - BBC News. 2016. Primary school pupil numbers soaring - BBC News. [ONLINE] Available at: http://www.bbc.co.uk/news/education-33094304. [Accessed 25 April 2016].

Data Protection:

How much does it cost? | ICO. 2015. How much does it cost? | ICO. [ONLINE] Available at: https://ico.org.uk/for-organisations/register/cost/. [Accessed 09 October 2015].

Data Protection Act 1998. 2015. Data Protection Act 1998. [ONLINE] Available at: http://www.legislation.gov.uk/ukpga/1998/29/section/7. [Accessed 09 October 2015].

Education | ICO. 2015. Education | ICO. [ONLINE] Available at: https://ico.org.uk/for-organisations/education/. [Accessed 09 October 2015].

Methodology:

Waterfall model - Wikipedia, the free encyclopedia. 2016. Waterfall model - Wikipedia, the free encyclopedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Waterfall_model. [Accessed 07 January 2016].

Agile management - Wikipedia, the free encyclopedia. 2016. Agile management - Wikipedia, the free encyclopedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Agile_management. [Accessed 07 January 2016].

Modified waterfall models - Wikipedia, the free encyclopedia. 2016. Modified waterfall models - Wikipedia, the free encyclopedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Modified_waterfall_models. [Accessed 07 January 2016].

Design inspiration:

Webdesigner Depot. 2016. How to recycle design patterns for UX success | Webdesigner Depot. [ONLINE] Available at: http://www.webdesignerdepot.com/2014/08/how-to-recycle-design-patterns-for-ux-success/. [Accessed 20 January 2016].

Luke Jones. 2016. *Growing as a Designer — Interactive Mind — Medium*. [ONLINE] Available at:https://medium.com/interactive-mind/growing-as-a-designer-b2ab9b553e44#.cvfz61d3w. [Accessed 20 January 2016].

Laracasts:

The Best Laravel and PHP Screencasts. 2016. The Best Laravel and PHP Screencasts. [ONLINE] Available at: https://laracasts.com/. [Accessed 07 January 2016].

Laravel:

Laravel - The PHP Framework For Web Artisans. 2016. Laravel - The PHP Framework For Web Artisans. [ONLINE] Available at: https://laravel.com/. [Accessed 07 January 2016].

Server Side Scripting:

PHP - Wikipedia, the free encyclopedia. 2016. PHP - Wikipedia, the free encyclopedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/PHP. [Accessed 08 January 2016].

Node.js - Wikipedia, the free encyclopedia. 2016. Node.js - Wikipedia, the free encyclopedia. [ONLINE] Available at: https://en.wikipedia.org/wiki/Node.js. [Accessed 08 January 2016].

Traffic Light Rating System:

Leadership Thoughts. 2016. How to Use RAG Status Ratings to Track Project Performance • Leadership Thoughts Blog. [ONLINE] Available at: http://www.leadershipthoughts.com/rag-status-definition/. [Accessed 29 March 2016].

White Box Testing:

tutorialspoint.com. 2016. *White box Testing*. [ONLINE] Available at:http://www.tutorialspoint.com/software_testing_dictionary/white_box_testing.htm. [Accessed 13 April 2016].

What is White Box Testing? Webopedia Definition. 2016. *What is White Box Testing? Webopedia Definition*. [ONLINE] Available at: http://www.webopedia.com/TERM/W/White_Box_Testing.html. [Accessed 13 April 2016].

Black Box Testing:

What is Black Box Testing? Webopedia Definition. 2016. *What is Black Box Testing? Webopedia Definition*. [ONLINE] Available at: http://www.webopedia.com/TERM/B/Black_Box_Testing.html. [Accessed 13 April 2016].

Appendices

Appendix A – Functional Requirements

Requirement	**#1**
Description:	The user must have the ability to register for their own Classmate account.
Rationale:	By registering their own account, a user can secure their secure account by using a unique email address and setting a secure password.
Dependencies:	This depends on the user having a browser no further than 4 versions back. Classmate will only offer support to this extent is there are issues.

Requirement	**#2**
Description:	The user must have a well structured and easy to navigate dashboard to manage their classroom
Rationale:	The main purpose of the Classmate application is to assist and equip teachers with the tools and features they need to manage their pupils. By creating a well-designed site structure and layout will only further the assistance provided to the user.
Dependencies:	This depends on the user having first registered their account and logged into the site.

Requirement	**#3**
Description:	The user must be able to view all of their students within their dashboard.
Rationale:	This will allow the user to quickly and efficiently access their student's information without searching through each student within the school.
Dependencies:	Requires the user to be logged into their account. If not their dashboard cannot be accessed.

Requirement	**#4**
Description:	The user should have the ability to create exams when needed.
Rationale:	Throughout the year class tests will be required to track student progress and growth in subjects covered in the classroom. The user has the ability to add an exam to the application which will later be used to display results of the pupils
Dependencies:	This is dependant on the teacher first having students added to their application. Without students added, exams cannot be created.

Requirement	**#5**
Description:	The user should have the ability to add their student's grades and personal details when needed.
Rationale:	For the user to manage their classroom they must first have the functionality to update the student's details and add grades. They can then view this within their dashboard as described in Requirement #2.
Dependencies:	This is dependant on the teacher having students in their class to edit their personal details and also having an exam added to their application which will then be able to store results for each student.

Requirement	**#6**
Description:	The user can create reminders to keep them focused on the upcoming tasks they must prepare for.
Rationale:	Having an all in one application becomes more efficient by also allowing the user to set reminders within their dashboard as well.
Dependencies:	Dependant on the user having an account and being logged into it.

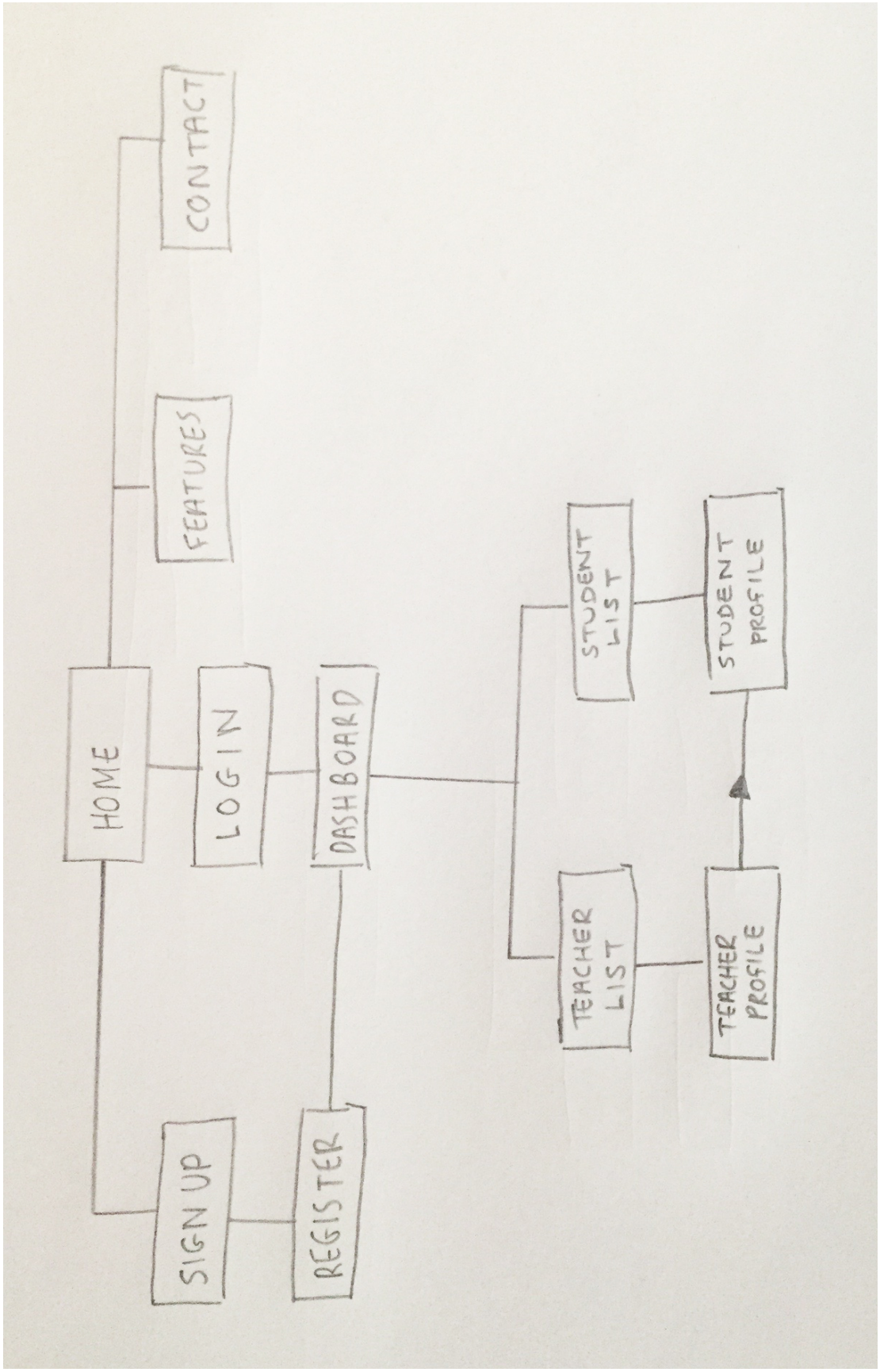
System Requirements

Requirement	**#7**
Description:	The system will allow a user to update their password if it is forgotten.
Rationale:	If a user forgets their password, they can request the opportunity to create a new one. This will save them having to struggle to remember their old forgotten password and continue managing their classroom through the application.
Dependencies:	Dependant on the user having an original account.

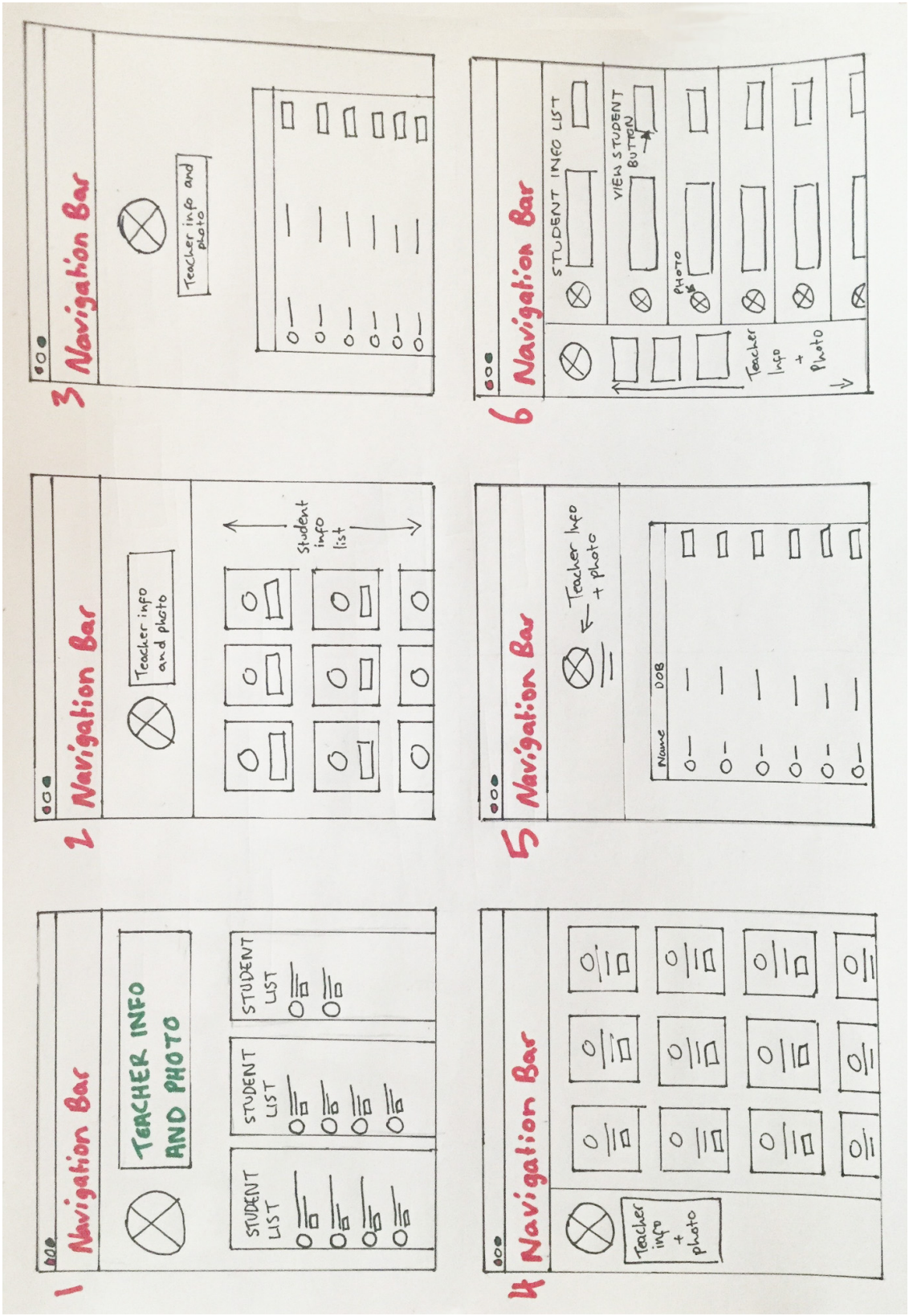
Requirement	**#8**
Description:	The system will allow the user to update their details.
Rationale:	By updating details, teachers can change their name, email address or school name quickly. This will be an effective feature when a class moves to a new teacher.
Dependencies:	Dependant on teacher first having created an account with details to edit.

Requirement	**#9**
Description:	The system will automatically log the user out.
Rationale:	For security measures, the user will be logged out automatically if there has been no interaction with their dashboard within a set time. This will help prevent non-users gaining access to student details.
Dependencies:	Dependant on teacher first having created an account and logged into their dashboard.

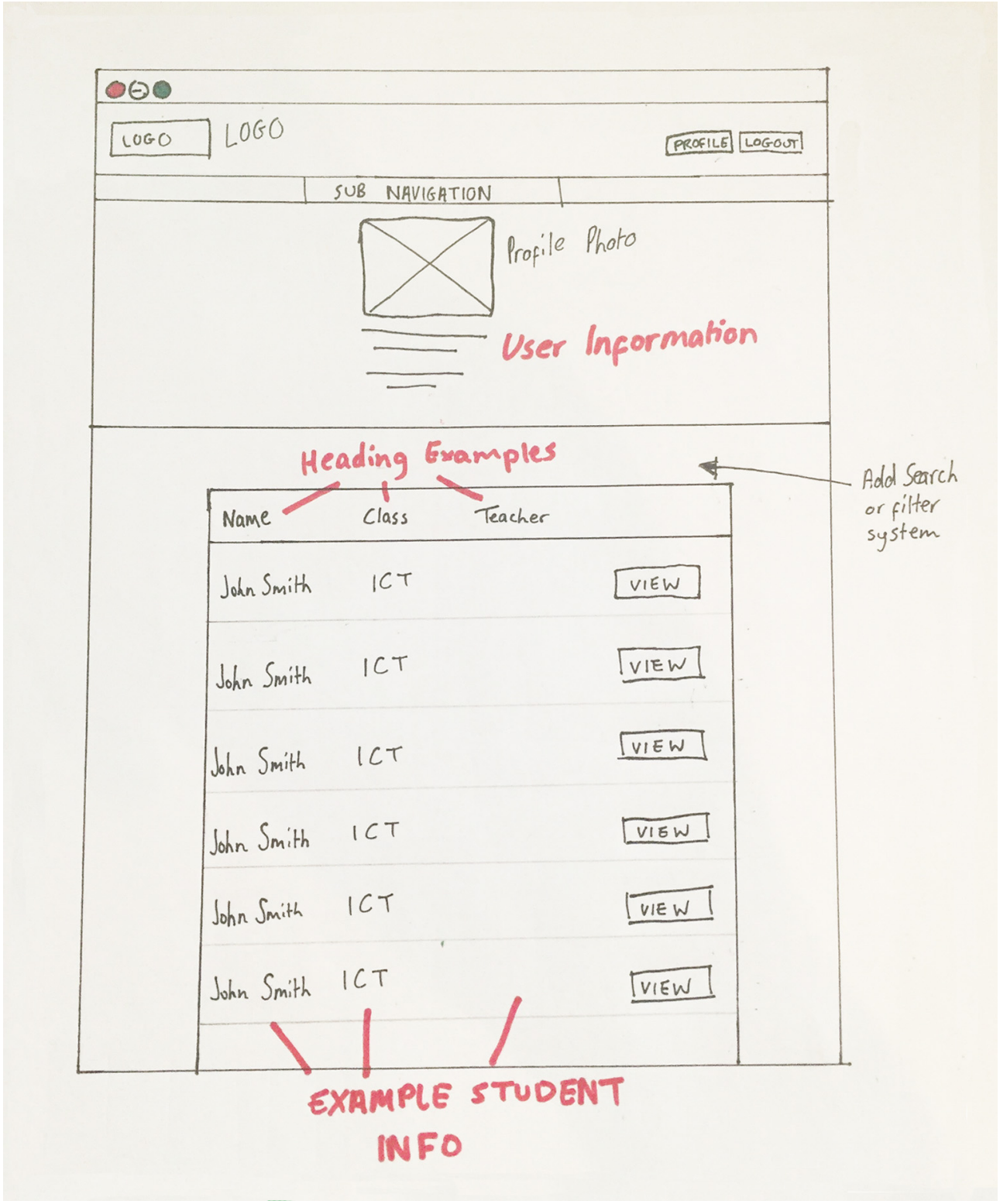
Appendix B – Site Map



6-Up


1 Navigation Bar
TEACHER INFO AND PHOTO
STUDENT LIST
STUDENT LIST
STUDENT LIST
2 Navigation Bar
Teacher info and photo
Student info list
3 Navigation Bar
Teacher info and photo
4 Navigation Bar
Teacher info + photo
5 Navigation Bar
Teacher Info + photo
Name
DOB
6 Navigation Bar
STUDENT INFO LIST
VIEW STUDENT BUTTON
PHOTO
Teacher Info + Photo

1-Up Paper Prototype

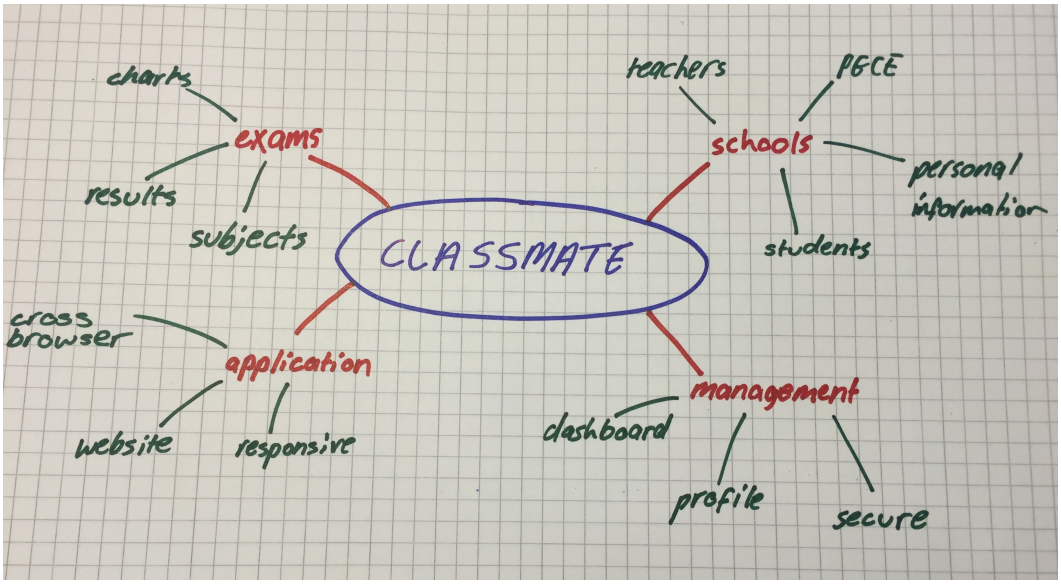


Appendix C – Traffic Light Rating System

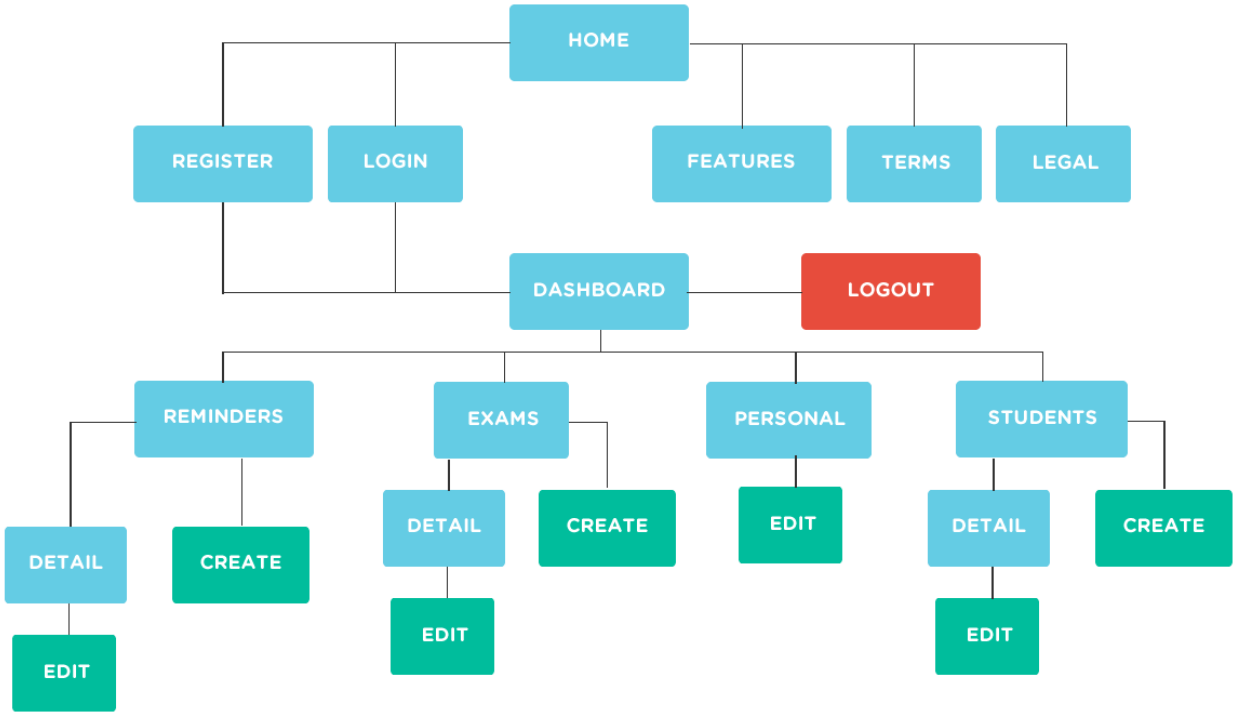
Risks	Impact (1-5)	Likelihood (1-5)	Initial Risk Level
The external server may crash, causing the product to be unavailable for an extended period of time, which is out of my control.	5	2	**Amber**
The development machine may crash, causing all project files to be destroyed and lost.	5	3	Amber
Users account is not secure and data can be accessed from an external source.	5	1	Green
Workflow stopped due to learning new development languages.	2	4	Green
Knowledge needed to create site is beyond my skill level.	4	3	Amber

Not meeting the timeframe set for project.	5	1	Green
Potential users not willing to move from competitive programs to try Classmate	5	3	Amber
Competitive product released at the same time as my product	4	3	Amber
Personal problems or illness preventing development.	5	1	Green
Database failure resulting in the loss of client's information.	5	3	Amber

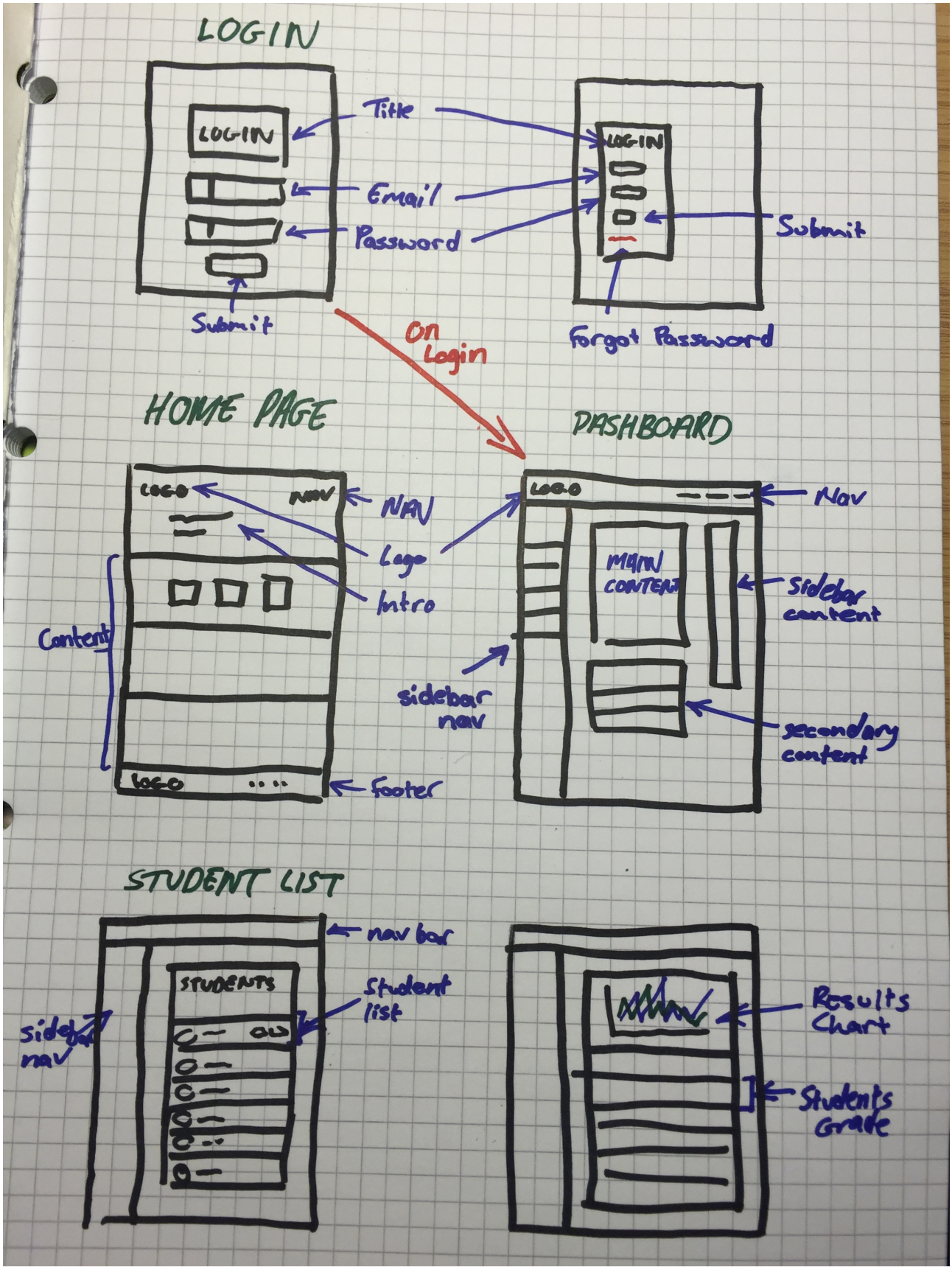
Appendix D – Mind Map



Users Journey / Site Map



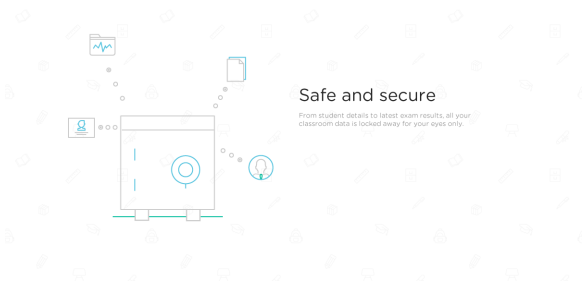
Initial Sketches

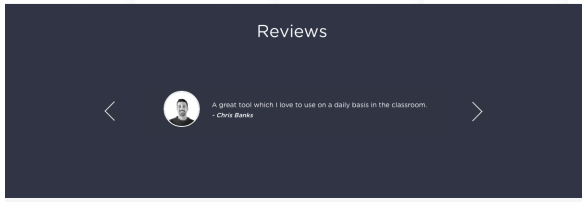


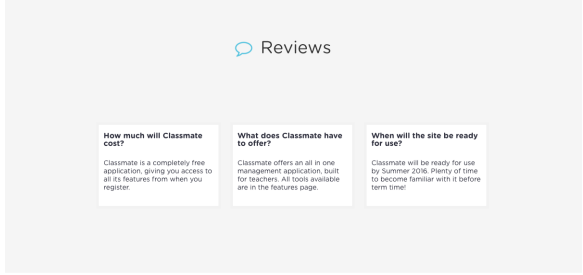
Refined Sketches

Designed Mockups

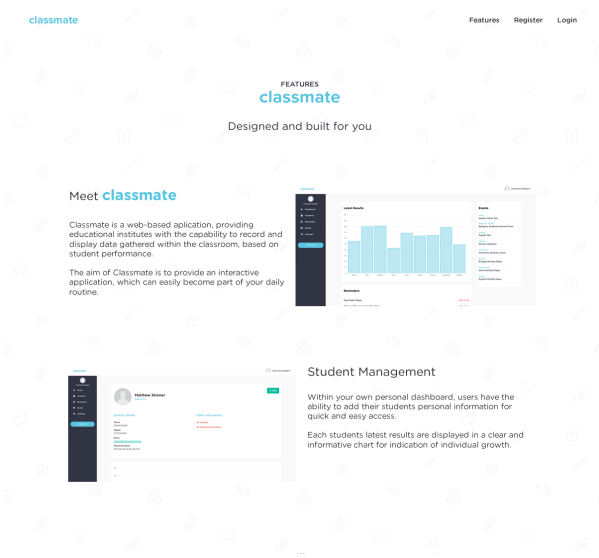
Home

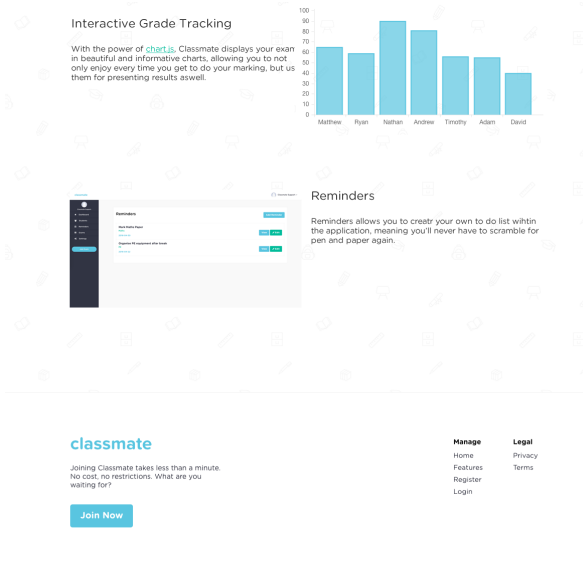




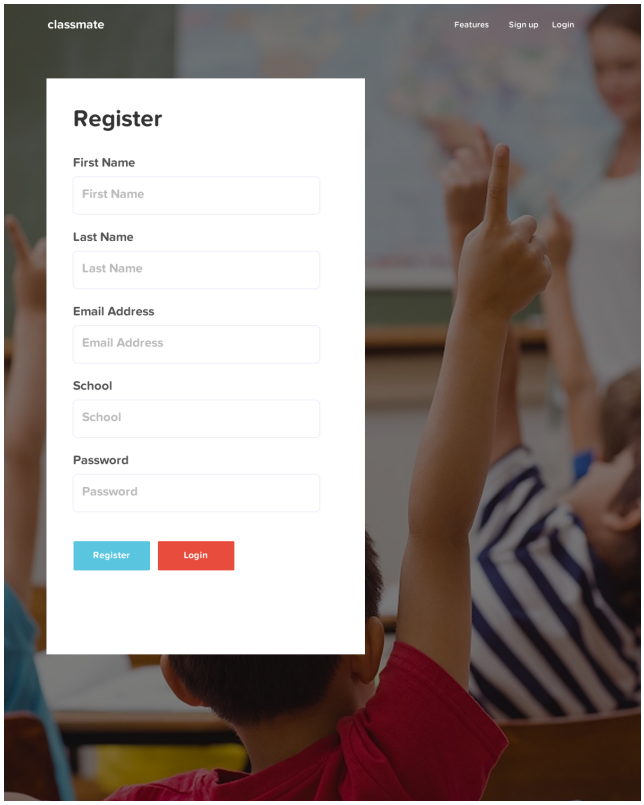


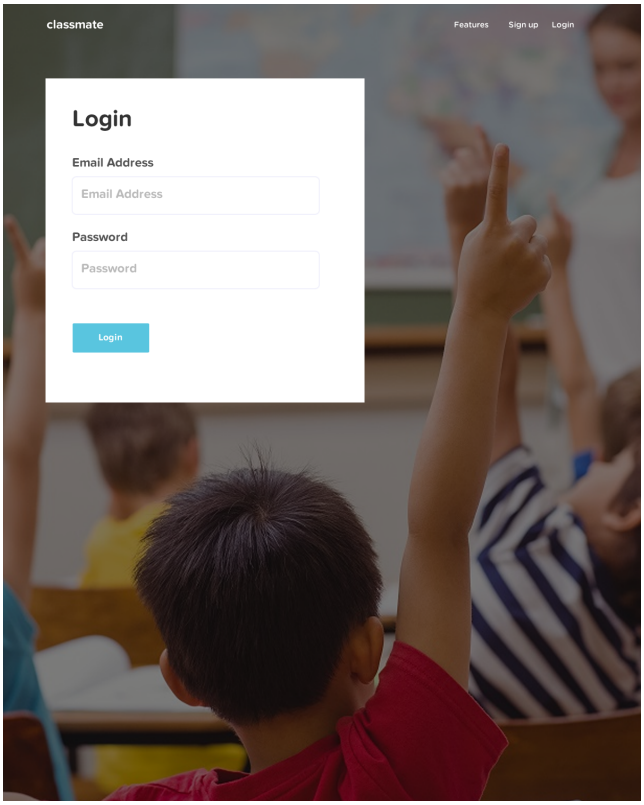
Features



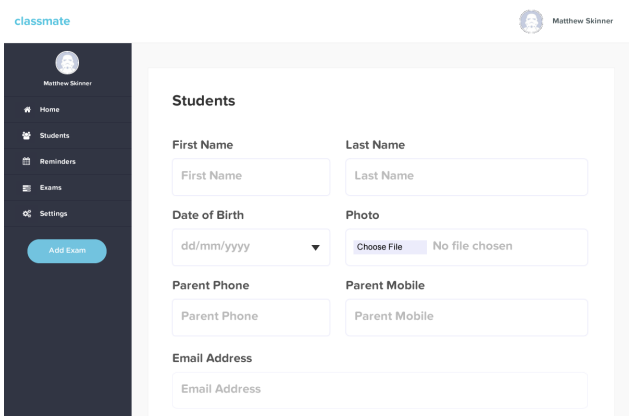


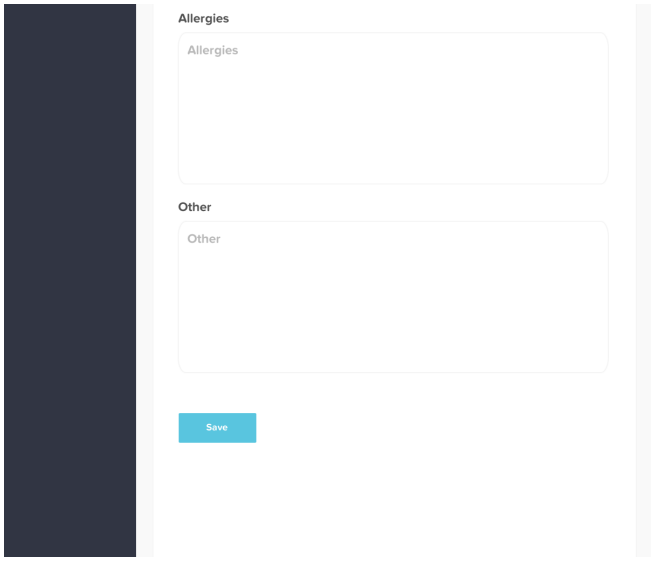
Register Form



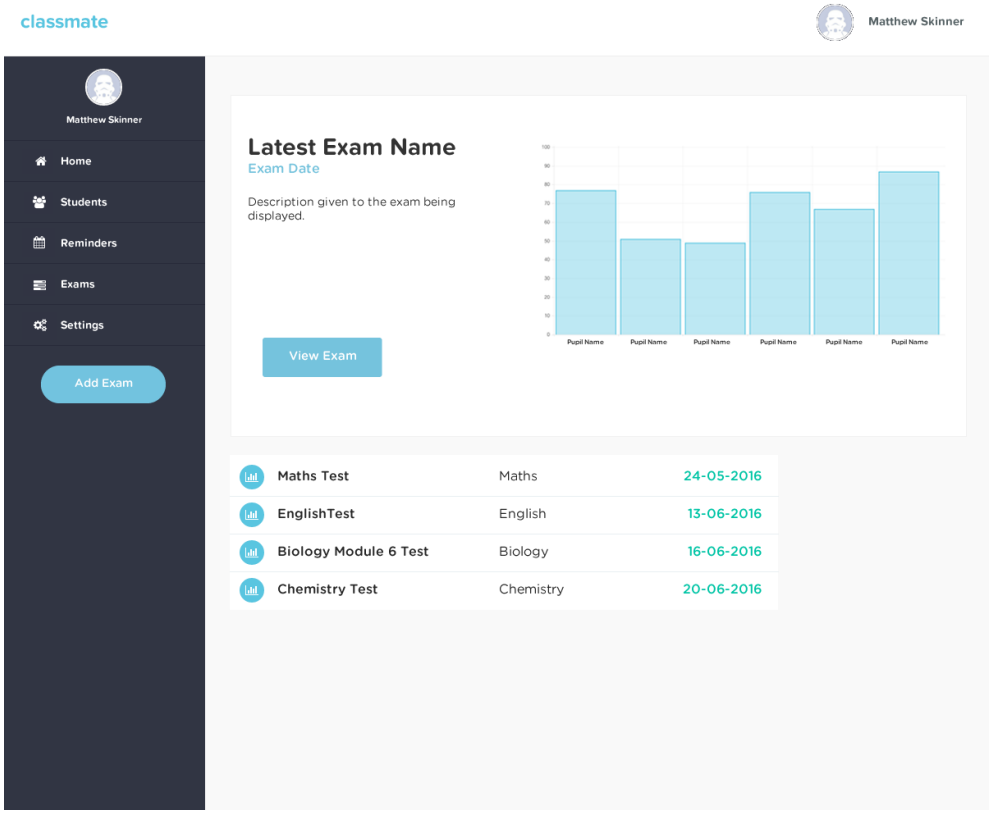


Dashboard Form





Exam



Data Design

User Table		
Field	Type	Allow Null
Id	Int	No
Fname	varchar	No
Lname	Varchar	No
Email	Varchar	No
School	Varchar	Yes
Password	Varchar	No
Photo	Varchar	Yes
Created_at	TIMESTAMP	No
Updated_at	TIMESTAMP	No

Students Table		
Field	Type	Allow Null
Id	Int	No
Student_id	varchar	Yes
User_id	mediumint	No
fname	Varchar	No
lname	Varchar	No
photo	Varchar	Yes
year	Varchar	Yes
dob	varchar	No
Parent_phone	varchar	Yes
Parent_mobile	varchar	Yes
Parent_mother	varchar	Yes
Parent_father	varchar	Yes
allergies	varchar	Yes
other	varchar	Yes
Active	mediumint	Yes
Created_at	TIMESTAMP	Yes
Updated_at	TIMESTAMP	Yes

Exam Results Table		
Field	Type	Allow Null
Id	Int	No
User_id	Int	No
Exam_id	int	No
Student_id	int	No
results	Varchar	Yes
Subject_name	Date	Yes
Updated_at	Timestamp	Yes
Created_at	Timestamp	Yes

Reminders Table		
Field	Type	Allow Null
Id	Int	No
User_id	Int	No
Title	Varchar	No
Subject	Varchar	Yes
Description	Varchar	Yes
Set_date	Date	Yes
Updated_at	Timestamp	Yes
Created_at	Timestamp	Yes

Exams Table		
Field	Type	Allow Null
Id	Int	No
User_id	Int	No
Title	Varchar	No
Subject	Varchar	Yes
Description	Varchar	Yes
Set_date	Date	Yes
Updated_at	Timestamp	Yes
Created_at	Timestamp	Yes

This is to certify that

Matthew Skinner

has successfully completed

Research Integrity (taught courses)

on

04 April 2016