UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

Virtual Team Software Development System VC01

Ben Smith

14160668

Declaration

The work described in this document is, except where otherwise stated, entirely that of the author and has not been submitted in any part for a degree at this or any other University.

Signed: _____

Dated: _____

Acknowledgements

I would like to express my deep gratitude to Dr. Val Casey for his valuable and constructive suggestions during the research, planning and development for this project.

I would also like to thank my friends and former colleagues for their help and valuable technical support during this project.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

Contents

Declaration 1
Acknowledgements 2
Introduction 6
Objectives 6
Motivations 7
Research 8
Other products 8
HCI 9
Agile 11
Event-driven architecture 13
Website application vs desktop application 14
Database Language 14
Server-side language 15
GSD 16
Requirements Analysis 18
Function Requirements 18
Implementation 21
Agile 21
Detailed Design 22
HCI 22
Colour & Icons 22
Security 24
Problems with Implementation 25
Product Features 26
Dashboard 26
People 26
Profile 27
Groups 28
..... 28
Offices 29
Documents 29
Calendar 30
Messaging 31
Future Work 33
Results and Evaluations 33

Conclusion.....34
References.....35
Bibliography 38
Appendices.....39

Summary

This project is focused on Global Software Development (GSD) and how it can be improved. This will be achieved by the development of specific software to better facilitate GSD. This project will involve researching all aspects of GSD and developing a software tool that will be of benefit to those companies who choose to utilise GSD.

Global Software Development is when a software development team is spread across multiple offices, often in different countries and time zones. In many instances team members come from different cultural and ethnic backgrounds. This can have a lot of benefits such as a reduction in development time, as you can have people working on the product 24/7 for continuous implementation. More skilled staff can work on the product because people and their expertise are assembled from around the world. Development expenses can, be reduced because aspects of the work can shift to low-cost countries.

However, to reap the rewards of GSD a company must get past a lot of the negatives and common traps these include, hidden delays, cultural issues, poor documentation, lack of synchronization, poor-quality communication and cooperation. Whilst that list may seem daunting they can be avoided by focusing on the right areas.

For this project a web application-based tool will be developed to help companies who have a globally distributed work force. The tool will have instant messaging, calendar, document sharing, will integrate a team management component inspired by Trello and will have a biography for each employee. The tool will be built using HTML 5, CSS 3, Node.js, JavaScript, jQuery and SQL.

Introduction

Whilst the advantages of GSD are easy to understand and have been clearly highlighted in the literature (Paulish 2004) it is still not without its problems (Casey et al. 2009). The goal of this project is to develop a tool to help specifically with GSD. To achieve this, we must look closer at the negatives that can arise with GSD, so they can be successfully addressed and improved.

Hidden delays, distributed teams tend to be slower than collocated ones, this is due to communications overhead, the limited communication window when work is carried out in different time zones (Paulish 2004). Cultural issues, GSD requires close cooperation of individuals with different cultural backgrounds. While some may find this enriching, they can also lead to serious and chronic misunderstanding (Casey 2010). Knowledge management, poor documentation can also cause ineffective collaborative development (Paulish 2004). The resistance to documentation among developers is well known, but in GSD environments the value of documentation is very important to clarify unspoken assumptions and ambiguity, and to support maintainability (Richardson et al. 2009). When teams are working across sites, the lack of synchronization can be critical, there is a need to assure commonly defined milestones and clear entry and exit criteria for all tasks. Typically, risk management does not consider possible impacts of the dispersion, diverse cultures, time and attitudes. Communication the most critical component of any effective team, communication between virtual team members is normally electronic with limited opportunities for synchronous contact, depending on temporal difference (Casey et al 2008). You can't share your thoughts as easily over the internet as you can face to face. Coordination, visibility, communication and cooperation are all negatively impacted by distance (Casey et al 2006).

Objectives

As stated the tool to be developed will be designed specifically to address the issues associated with GSD and enhance its effective operation. It must adapt to provide the required level of functionality and efficiency. The appropriate architecture must be identified and then implemented, so to must the software development methodology. The system must incorporate the best of Human Computer Interaction (HCI) such as being clear, concise, efficient and familiar. This system will be used by companies, so it is essential that it is secure.

Pursuing greater knowledge and experience in how to plan, design, develop, test and deploy a website and to do so comprehensively is a big opportunity to learn end to end web development. Also, to expand my knowledge of SQL, HTML, CSS and JavaScript, with a project as large as this, knowing there will be problems and elements that I have not encountered before that must be overcome. Time will have to be dedicated to solving and overcoming these problems but in the end, they will be of great benefit to the project and to my professional development.

Due to the size and development time of this project a correct development strategy is crucial, having had little experience with development strategies in the past, this is a great opportunity to learn about and gain experience with my chosen development strategy, which, will be the Agile methodology. The project will be completed iteratively and in an incremental fashion, validating and verifying each step of the way. This is to ensure that the software is of high quality, meets its specification and user needs.

The research conducted has highlighted specific requirements needed for the tool to achieve its objectives.

It must incorporate instant messaging both person to person and group messaging, all communication must be secure.

The calendar must be individual to each user, sending a meeting request to any users should show any conflicts with each user's schedules.

The document sharing and control, must only allow approved users to access certain files and folders, some users may only be allowed to view, while others can edit. Deleted files should go to the recycle bin for a period before being erased permanently. This file system must also hold previous versions of each file.

Employees will have their own personal profile pages with a summary of their personal details which will list birthday, work email, work phone. Their personal social media and general interests. Behind a button will also list their emergency contact details.

The project management system (White Board) will be a visual tool to help people and guide them through their tasks. A project board and its tasks will be created by team lead(s). People will be attached to certain tasks. This component has been inspired by Trello.

Motivations

The motivations for doing this project where clear, GSD is becoming increasingly prevalent. Due to globalisation, GSD is often the default choice for companies when developing software, but as discussed correctly implementing a successful GSD program is not easy. This is something I experienced first-hand during my co-op, there was two offices (Germany and Malta) which meant two IT teams. Trying to co-ordinate projects across both locations was often a struggle, because of poor communication and because of that planning, co-ordination and cohesion suffered. Due to this experience I believe I could create a capable tool specifically to address the problems I encountered.

As mentioned, GSD together with globalisation is growing extremely quickly, the opportunity to develop a tool to help address the problems associated with GSD and to conduct in-depth research into this growing and increasing relevant area is something that I am very interested in doing. Researching in this area will also be of great benefit to me when I go into the workplace.

Web development is an area which is also highly intriguing, with a web application no software must be downloaded and so long as a user has access to an internet connection it

can be used anywhere. Web applications are easily scalable and there is a wide range of technologies to choose from to enhance the tool. The web development community is highly active which is great when learning and researching new technology. Creating a web application is also the perfect opportunity to work on and research Human Computer Interaction which will be vital to this tool.

Research

I was lucky to have a supervisor that has spent many years researching the topic of global software development from this I had a great body of work to look through. Other papers were found using google scholar, IEEE Xplore Digital Library and the ACM Digital Library utilising these resources I was able build and expand my knowledge of this area.

Other products

While defining the objectives and requirements for this tool research was done by looking at other products already available. Looking at what features they incorporated, where they were strong and weak so to better understand what made a good product.

The products looked at included Bitrix24, Trello, Microsoft SharePoint.

Bitrix24 (Britrix24) whilst primary a Customer Relationship Management (CRM) does have useful features for staff organisation, on the home page it shows upcoming birthdays, the tool itself is easy to use though it can get a bit tricky for admins due to the abundance of buttons. Employees can clock in and out showing their online status to other employees. Bitirx24 includes a calendar too, so meetings can be scheduled, and people can be notified of important dates. However, while Bitrix24 is a very complete and powerful tool it is clear that it has not been designed with GSD in mind, there is no way of communicating real time with other staff, the file sharing component is lacking too. The employee's information is also limited to work information this means the opportunity to bond with colleges is limited especially if they are not in the same office. The clocking in and out feature is a very good one and since this project is focused globally with teams spread across the world, when people are leaving one office and just entering another this could be a very useful feature to have if someone wished to make a real time query.

Trello (Trello) is one of the leading project management application, it utilizes the concept of boards (which correspond to projects) and within boards, there are cards (which represent tasks). The cards contain lists which can be used to track the progress of a project or to simply categorize things. You have read that I am looking to incorporate something similar to Trello in this project organising your teams in GSD is crucial and Trello does a fine job of this. But, like Bitrix24 is hasn't been designed for GSD there is no way to communicating with other team's members and it lacks employee information and locating

files is tricky because they are attached to cards, there is no edit or view control over them either.

Microsoft SharePoint (SharePoint) is a Microsoft product specialising in document management. It can be used for personal use but is mainly used by small to large corporations. Using out-of-the-box functionality, you can facilitate various business processes and enable users to fulfil their daily tasks faster and with less effort. Whilst SharePoint does come with a lot of built in features, enabling all of them can lead to a very messy tool that can end up hindering a user's activities. The features aren't customized to your specific organizational needs, so you will have to tailor your SharePoint to your specific needs. This customisation requires considerable time and investment.

There are common themes between all of these products. They are all web-based applications, it is not necessary to download any software to your computer so long as you have an internet browser you can access them. Each of these products have features that can implemented and can be improved upon. Such as the Trello work flow management, Bitrix24's clock in and out and SharePoint file management system will be able to provide a great inspiration as to what is needed in an enterprise file management system. However as mentioned we can learn from and improve the current products. Both bitrix24 and SharePoint have Graphic User Interfaces (GUI) that can be hard to follow and can become so crowded you can't find anything on the page, when dealing with important documents and dates the last thing you want is confusion.

Whilst there are tools on the market already that deal with file, staff, date and workflow management none of these tools are designed specifically with global software development in mind. None of the tools discussed provide in-depth ways to get a message across to someone on the other side of the world who may not be in the office for another 12 hours or longer if the countries celebrate a different holiday. In each tool there is a barrier to communication and having done a great deal of research on this topic by reading various case studies, communication is the key to making global software development work for a company.

HCI

Human-Computer Interaction (HCI) is a multidisciplinary field of study focusing on the design of computer technology and, in particular the interaction between humans and computers. HCI is a broad field which overlaps with areas such as user-centred design, user interface design and user experience design. The difference between HCI and User Experience (UX) design is that HCI is more academically focused and involved in scientific research and developing empirical understandings of users. Where UX designers implement the finding from HCI and implement them into industry.

A well-designed user interface should be eight things (Dmitry 2009): clear, concise, familiar, responsive, consistent, attractive, efficient and forgiving.

Clear, clarity is the most important element of user interface design. The entire purpose of the user interface is to allow people to interact with your system in an effective manner. If people cannot figure out how to use the tool they will get confused productivity will drop because unnecessary time will be wasted.

Concise, clarity in a user interface is great, however over-clarifying brings with it, its own set of problems. It is easy to add definitions and explanations, but every time you do that you add mass and the interface grows. Add too many explanations and users will have to spend too much time reading through them. If something can be explained adequately in one short sentence instead of several then one sentence should be used. This will save valuably time for the users. Even though making a user interface both clear and concise takes a lot of time it will add a great deal to the effeteness of the tool.

Familiar, familiar is something that is intuitive, when designing an interface there is no point in 'reinventing the wheel' a button should do something that the user expects and when users are familiar with something, they know how it behaves – they know what to expect.

Responsive, responsive means fast, the interface, if not the software behind it, should work fast. Waiting for things to load is frustrating seeing things that load quickly greatly improve the users satisfaction with the tool. Begin responsive also means providing feedback to the user such as a loading screen or a progress bar to let the user know that the software is responding to his or her request.

Consistent, consistent interfaces allow users to develop usage patterns this also ties in with familiarity they want to know what a certain button or tab does if the design changes this lack of consistency will hurt the users experience. With consistency users will also learn how certain things work and will be able to work out how to operate new features quicker, extrapolating from those previous experiences. Take the adobe creative cloud as a prime example of this. The user interfaces are all very similar will the same colour and layout, the icon for the brush in Photoshop and Illustrator are both the same because there is no need for variation.

Attractive, when a user interface is attractive is can make the User Interface (UI) all that more enjoyable to use. However, aesthetics should be used in moderation and to reinforce function. Simply splashing the tool with fancy fonts, colours and pictures will not make the tool enjoyable or easier to use in fact it will do the exact opposite.

Efficient, a user interface could be viewed as a means to an end. The end being a specific function in the tool. A good UI should enable you to get to that function a quickly and an efficiently as possible using the least number of clicks. To do this the key features of the tool must be identified and the buttons to access or execute those features be put in the correct place adhering to the guidelines mentions above.

Forgiving, people will make mistakes and handling those mistakes is an important indicator of software quality. If someone adds or deletes information that they didn't mean to, that action should be easily reversible and not punish the user for their mistake.

Trying to achieve the perfect UI is an incredibly difficult task that will require many iterations and a lot of time and even then, it still won't be perfect. But that should not undermine the fact that good HCI is a vital component in developing productive and useable tool that users can enjoy.

Agile

The Agile development methodology is rooted in adaptive planning, early delivery, continuous implementation and testing, verifying and validating along the way in an iterative and incremental fashion.

Verifying is checking whether the software is meeting the correct requirements and validation is checking whether we are building what the customer wants (Sivakumar et al 2011).

Agile software development refers to a group of software development methodologies based on iterative and incremental development verifying and validating along the way, where requirements and solutions continuously evolve. Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals. (WHAT IS AGILE METHODOLOGY)

The Agile Manifesto was developed by a group fourteen leading figures in the software industry and reflects their experience of what approaches do and do not work for software development.

They outlined four important values of the agile manifesto:
1. Focus should be more on individuals and interactions instead of processes and tools.
2. Working software is more important that comprehensive documentation.
3. Customer collaboration is more vital than contract negotiation.
4. The process should respond to change rather than follow a plan.

There are also twelve principles behind the Agile Manifesto (Agile Alliance):

1. The highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a

development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Before going through the benefits of agile project management first let's compare the traditional & the agile development. In software development, we often talk about the traditional model which refers to the Waterfall Model. Very different to Agile method specially because it's not iterative, Waterfall is more about a process where you can see the progress flowing through the difference phases (Moore, S., Barnett, L. 2004). It's a sequential model usually going from requirement analysis, design, implementation and then testing. Agile project management addresses perfectly customer's needs. During the whole cycle, user involvement is encouraged, providing visibility & transparency, showing the actual progress of projects. As mentioned earlier, Agile method is all about iterative planning, making it very easy to adapt when some requirements change. The fact that there is continuous planning and feedback through the process means that we start delivering business value from the beginning of the project (Continuous Feedback in Agile Teams).

Whereas Waterfall includes several testing and bug fixing cycles before releasing a product, Agile is far more collaborative and iterative. One of the biggest differences is that Waterfall calls for heavy documentation early on. This documentation makes it harder to change features as the process goes on, this assists with the unpredictability of writing software. You can think of Agile like many "mini waterfalls," since requirements are well defined at the start of each sprint and shouldn't shift within it. The difference is that detailed requirements for the next sprint are not set months in advance. (Agile Methodology: The Complete Guide to Understanding Agile Testing)

There are a lot of benefits to using agile for both the developers and the customer. The customers find that the vendor is more responsive to development requests. High-value features are developed and delivered more quickly with short cycles, than with the longer cycles favoured by classic "waterfall" processes.

Team members enjoy development work and like to see their work used and valued. Agile benefits team members by reducing non-productive work (e.g., writing specifications), and giving them more time to do the work they enjoy. Team members also know their work is valued, because requirements are chosen to maximize value to customers.

When using Agile the software is created in small chunks this increases the modularity of the programme because of this the software can be easily modified or extended since in theory the increased modularity creates a lose coupling between various bit of code with is

of huge benefit whether the software be small or large. Since the requirements are well defined before each sprint this also provides very little opportunities for scope creep.

By breaking down the project into manageable units, the project team can focus on high-quality development, testing, and collaboration. Also, by producing frequent builds and conducting testing and reviews during each iteration, quality is improved by finding and fixing defects quickly and identifying expectation mismatches early.

The development methodology used in this project will be a type of Agile with the supervisor paying the role of the customer. Every week a sprint will start, and a module will be created and implement in to the already existing software and then tested, at the end of the week I will meet with the customer to verify and validate that the software is correct.

Event-driven architecture

An event-driven architecture (EDA) is a framework that orchestrates behaviour around the production, detection and consumption of events as well as the responses they evoke.

Many programs spend most of their time waiting for something to happen. This is especially true for computers that work directly with humans, but it's also common in areas like networks. Sometimes there's data that needs processing, and other times there isn't.

The event-driven architecture helps manage this by building a central unit that accepts all data and then delegates it to the separate modules that handle the particular type. This handoff is said to generate an "event," and it is delegated to the code assigned to that type.

Programming a web page with JavaScript involves writing the small modules that react to events like mouse clicks or keystrokes. The browser itself orchestrates all of the input and makes sure that only the right code sees the right events. Many different types of events are common in the browser, but the modules interact only with the events that concern them. This is very different from the layered architecture where all data will typically pass through all layers. Overall, event-driven architectures are easily adaptable to complex, often chaotic environments, they can scale easily and are easily extendable when new event types appear (Event-Driven Architecture: A Primer).

The architecture does have caveats though, testing can be complex if the modules can affect each other. While individual modules can be tested independently, the interactions between them can only be tested in a fully functioning system. Error handling can be difficult to structure, especially when several modules must handle the same events. When modules fail, the central unit must have a backup plan. Messaging overhead can slow down processing speed, especially when the central unit must buffer messages that arrive in bursts. Developing a system wide data structure for events can be complex when the events have very different needs. Maintaining a transaction-based mechanism for consistency is difficult because the modules are so decoupled and independent.

Event-driven architecture is best for asynchronous systems with asynchronous data flow, applications where the individual data blocks interact with only a few of the many modules and user interfaces. (event-driven architecture (EDA))

Website application vs desktop application

Over the last five years there has been a huge decline in the number of local desktop applications being created and there is good reason for this. With web applications there is nothing to download or install. It is a lot easier to convince people to use your application if they can use it immediately. There is also the benefit that the website can be updated instantly and there is no interruption to the user's experience. However, the main benefit of using a web application is you don't have to develop an application to a specific operating system, for example a native application may need both a windows and mac version built which drastically increases development time. However, a web application will sacrifice speed for this compatibility.

Because of the huge benefits and limited drawbacks along with my experience already creating websites, creating a website application was the clear choice.

Database Language

The thing that must be looked at when choosing a database language and its management system is the difference between SQL vs NoSQL.

SQL (Structured Query Language) databases are primarily called as Relational Databases, whereas NoSQL (non-relational Structured Query Language) database are primarily called as non-relational or distributed database. SQL databases are table-based databases whereas NoSQL databases are document based, key-value pairs, graph databases or wide-column stores. This means that SQL databases represent data in form of tables which consists of n number of rows of data whereas NoSQL databases are the collection of key-value pair, documents, graph databases or wide-column stores which do not have standard schema definitions which it needs to adhere to.

SQL databases have predefined schema whereas NoSQL databases have dynamic schema for unstructured data. SQL databases are vertically scalable whereas the NoSQL databases are horizontally scalable. SQL databases are scaled by increasing the horse-power of the hardware. NoSQL databases are scaled by increasing the databases servers in the pool of resources to reduce the load.

The database structures MySQL (SQL) and MongoDB (NoSQL).

MySQL is a full-featured open-source relational database management system (RDBMS). It stores data in tables that are grouped into a database, uses Structured Query Language (SQL) to access data and such commands as 'SELECT', 'UPDATE', 'INSERT' and 'DELETE' to manage it. Related information can be stored in different tables, but the usage of JOIN

operation allows you to correlate it, perform queries across various tables and minimize the chance of data duplication.

MongoDB is a popular open-source document-oriented database based on NoSQL. With MongoDB, documents are created and stored in BSON files, Binary JSON (JavaScript Object Notation) format, so all JS types of data are supported. That being the case, MongoDB is often applied for Node.js projects. Besides of that, JSON enables transferring data between servers and web applications with the use of the human-readable format. It is also a better option, when it comes to storage capacity and speed, as it offers greater efficiency and reliability (MongoDB).

Even though MongoDB is often used together with Node.js. Learning MongoDB together with everything else is just too big a task. Node.js also has great support for SQL so there will be no problem with compatibility.

Server-side language

PHP vs Node.js

PHP (Hypertext Pre-processor) is a general-purpose scripting language that quickly became the de facto server-side language of choice for web developers. Today, most sites on the web run on PHP, due in large part to its popularity as the language of choice for content management systems (CMS) like WordPress.

JavaScript is a scripting language that typically runs in the browser and makes web pages dynamic and interactive, but ever since the release of Node.js, it became possible to perform asynchronous coding with JavaScript on the back end. Node.js is a development and runtime environment with a multitude of available frameworks that run on top of it.

The runtime environments while both JavaScript and PHP can be embedded directly into HTML, they both need an interpreter to run. PHP has long been readily straightforward to install and use on the server-side and is powered by the Zend engine using apache. Node.js is a runtime environment for JavaScript on the server side, powered by Google's V8 JavaScript engine but you must create your own server for node.js to work.

Node.js leverages JavaScript event loop to create non-blocking I/O (Input/Output) applications that can easily service multiple concurrent events. Using JavaScript built-in asynchronous processing, one can create highly scalable server-side solutions that maximise the usage of a single CPU and computer memory while servicing more concurrent requests than conventional multithreaded servers. This functionality makes Node.js a great fit for asynchronous, data-driven applications and heavy I/O-bound workflows such as RTAs (Real-time Applications) or SPAs (Single-page Applications), where Node ensures excellent runtime performance compared to PHP which is far slower than Node.js (Comparison: Node.js vs. PHP) (Node.js) .

Because of the concurrency benefits of Node.js and the fact that I already have experience in PHP and I would like to learn something new Node.js was the clear choice.

GSD

Having talked about the GSD and its barriers and problems at the start let's have a look at case studies.

This case study is from Global Software Development in Practice Lessons Learned by Rafael Prikladnicki, Jorge Luis Nicolas Audy and Roberto Evaristo (Prikladnicki, R., Nicolas Audy , J.L., Evaristo , R. 2003).

This case study discusses a GSD undertaking between offices in Brazil and in the United States. When the company was asked why they choose to use GSD they cited cost reduction, quality focus, competitiveness, the creation of centres of competence, having each unit with specialization in specific skills and/or technology.

This case studied four projects each of varying size some being led by Brazil others in the USA or both.

The results 'We concluded that difficulties, solutions and critical success factors involve three dimensions: technical, nontechnical and hybrid (both technical and nontechnical). Technical factors are those related to technical knowledge used in software development, whereas nontechnical (social, cultural, languages, behaviour, and so on) factors are the ones that include knowledge of related areas needed for the software development activity. The hybrid factors include both technical and nontechnical knowledge.'

According to the interviews, the difficulties with GSD are related to the lack of standards in the activities between distributed teams, the difficulty of sharing information, and the lack of a well-defined software development process. Difficulties concerning language barriers and communication, cultural differences, context sharing and trust acquisition between teams were also found. The case studies identified the lack of a formal and consistent knowledge management process in both organizations, emerging as a great obstacle to sharing information. Trust was also a problem, mainly the necessity of a distributed trust acquisition. Finally, communication and language misunderstandings were motivated by the cultural differences between the dispersed stakeholders.

In a second paper Uncovering the Reality Within Virtual Software Teams by Valentine Casey and Ita Richardson. It looked at two case study both with companies bases in the United States, Ireland and the Far East (Casey, V., Richardson, I.).

Several problems were noted one was to do with communication over email. 'While e-mail was used to communicate, it was also used as a weapon to publicly attack fellow team members. The practice of copying senior management on minor problems, which were caused by team members at the other location, was widespread. One respondent stated "Developer A (at the other location) would always copy. If you asked them what time it was they would copy the reply to your manager." Both groups were equally guilty of employing this tactic.' This was observed to created enormous mistrust towards the other team and even though the above tactic was aimed to one individual it was perceived to be an attack on the entire group.

Another problem noted the difference in technical ability between two offices ‘To facilitate knowledge transfer Irish-based mentors were provided for Far Eastern team members. In reality given the time zone difference and the limited use of synchronous communication tools this had its limitations. A trainer travelled from Ireland to the Far East to provide formal training. Remote online training was also introduced. The trainer’s assessment was “A lot of them had very little experience and were just fresh from college or fresh from another job in a different sector or different company.” In the Far Eastern organisation there was limited technical knowledge and linguistic and cultural problems were encountered. The knowledge of English within the Far Eastern organisation was very varied. The ability to communicate effectively is key to team-based success. The ability to understand the trainer’s language is paramount. It was also very difficult to assess the effectiveness of the training provided, as there was very little feedback received from the Far Eastern participants.’

There was also a problem of huge mistrust between the two offices even though they had worked together in the same office for 6 months ‘People who worked together very successfully while co-located were now actively obstructing and blaming each other for all the problems that arose during projects. It was obvious that team members were now aligned by geographical location and there was a very clear “we verses they“ culture.’

Having read through these case studies there is clear similarities, lack of communication being the main one. In the case studies you get the impression that everyone involved presumes that processes, methodologies and technologies involved are just known to everyone and if someone doesn’t know they will find out by osmosis. If one person doesn’t like someone else for whatever reason either cultural issues or language barriers it isn’t discussed, and it just gets worse.

Nowhere in the cases studies was it mentioned how the people in different offices where introduced to each other, so we must presume that they weren’t which I believe is a crucial oversight.

Simply communicating with people provides a much greater understanding of what the other person is going through and will increase co-operation and motivation for all involved in the project (Casey et al 2008).

Developing software is a highly socio technical activity, being good at just the technical isn’t good enough for GSD even though offices may be half the world away communication and collaboration is even more important.

With the lessons learned from these studies although global software development is inherently difficult given the complications introduced by time and distance, projects that are planned and designed for distribution can be successful. Global projects that centrally control the requirements and architecture design, then organizing the distributed development as component developments, using small sub teams would have a much greater chance of succeeding.

Requirements Analysis

Defining requirements for this project was done by first looking at the project title "Virtual Team Software Development System", what does it mean and what tools are needed to manage development teams across the globe? Then I had a detailed look at the literature and the specific requirements that it highlighted for any such tools. I then review other similar products observing what they did right and what they did wrong with their features and their user interfaces. I also interviewed my former colleagues from my Co-op to discuss what features would be helpful when managing development teams. This resulted in the compilation of a list of relevant features which could be used as the basis for defining my requirements (See Figure 2, 3 and 4).

This list was then brought to my supervisor Dr. Val Casey. Dr. Casey is a lecturer and researcher in the University of Limerick. He has published widely in a number of different areas of Software Engineering including Global Software Development and the successful operation of Virtual Teams. In both these areas he is an internationalized recognized expert and has published 30 international publications which include a book, numerous peer reviewed book chapters, journal papers and international conferences. Each of which dealt with different aspects of globally distributed software development. (Val Casey)

My supervisor and I discussed each feature which I proposed in an objective and critical manner. The goal was to determine which features should become functional requirements and why such features were necessary. These functional requirements were then documented. During this process additional requirements were identified and potential problems with implementation were discovered. During my FYP presentations other supervisors suggested features they would expect to see in such a system and these were incorporated.

Function Requirements

FR (Functional Requirement) Number	Title
FR 1	Sign-in
FR 2	Register
FR 3	Logout
FR 4	Add User
FR 5	Profile
FR 6	Edit-Profile
FR 7	People
FR 8	Groups
FR 8.1	Create Group
FR 9	Offices
FR 9.1	Add Office
FR 10	Calendar
FR 10.1	Calendar Add Event
FR 10.2	Calendar Remove Event
FR 11	File Management

FR 12	Messaging
FR 12.1	Messaging Create Group

FR 1	Sign-in
Description	User will provide username and password, if they are correct the user will be sent to the dashboard.
Notes	If the username and password is incorrect an error message will be displayed.

FR 2	Register
Description	User gives company name, email, first name, last name and password. This will register the company on a new database and user provided will become the admin.
Notes	If password doesn't meet the security criteria an error message will be displayed.

FR 3	Logout
Description	Will log the current user out of the system
Notes	

FR 4	Add User
Description	Admin will type in email, office and first and last name. The user is then added to the system, so the user can no log in will the details provided by the admin.
Notes	

FR 5	Profile
Description	Will display the detailed user information about a specific user
Notes	

FR 6	Edit-Profile
Description	A user can edit information in their own profile.
Notes	Admins can edit all profiles.

FR 7	People
Description	Will list the current people in the company.
Notes	Admins can delete users from here, also where add user (FR 4) is located.

FR 8	Groups
Description	Lists the current groups with the admins and people within them.
Notes	Admins can add or remove people from within the group.

FR 8.1	Create Group
Description	Admins can create a group.
Notes	

FR 9	Offices
Description	Lists the offices on the system.
Notes	

FR 9.1	Add Office
Description	Admins can add a office.
Notes	Once office is added a new group(FR 7) is automatically created.

FR 10	Calendar
Description	Displays events and dates.
Notes	Calendar will be user specific.

FR 10.1	Calendar add event
Description	Clicking on the calendar will allow the user to add an event and specify time.
Notes	

FR 10.2	Calendar remove event
Description	Clicking on an event will give you the option to remove / change it.
Notes	

FR 11	File Management
Description	Users will be able to upload, download and view documents.
Notes	Different users will have different permissions as to what files they can view or edit. Previous versions of a file will be saved.

FR 12	Messaging
Description	Users will be able to communicate in real time with each other
Notes	

FR 12.1	Messaging create group
Description	Users can create a group and all messages will be seen by the group.
Notes	

We also discussed the non-functional requirements of the system and they are as follows:
a) Security: It's one of the most important requirements in web applications. In this application, we need to protect data when it is sent by user and when it arrives to the server when logging in and sending messages to one another.
b) Availability: The system needs to be available to users all the time, and to everyone.
c) Reliability: The system should perform correctly in different cases and it should deal with data from different users efficiently.

d) User-friendly: The system should have an attractive and clear interface, so users will not have any difficulties in using it.

Implementation

Now that the initial research has been completed the features and requirements have been defined. So too has the implementation languages that will be used (HTML 5, CSS 3, JavaScript, Node.js and SQL). The development methodology and the architecture design (Agile methodology and Event-Driven architecture). At the start of the project a project plan had been created and was then updated to facilitate the implantation process (See Figure 1).

Implementation commenced early on in the project (week 5) using the agile approach. Having never used Node.js before it required a lot of research and trial and error before I became comfortable with it. This did delay the implementation by about a week and a half and extra work had to be put in to gain that time back. During the implementation I completed the sign-in/register page, Account Page, Bio Page, Groups, Offices, People, Calendar, Messaging and Document pages. The pages and features have been implemented and unit tested to make sure they work and meet their requirements.

Agile

This project was completed using agile principles rather than using a specific type of agile like SCRUM. Each week I would meet with my supervisor who is a subject matter expert (SME) showing him the functionality had been added. In this meeting my supervisor would act as the customer to validate the functionality and provide feedback to me the developer(See Figure 5 and 6). I would demonstrate the functionality to prove that is satisfies the functional requirements. I would then take any feedback such as how I could improve the ease of use etc... I would then take this feedback and where relevant note them as requirements where they would then be added to the requirements list. Once these requirements where completed I would refactor the code to make it more efficient and easier to read removing any bugs or problems that weren't spotted before. It would then be integrated with the rest of the code where I would conduct an integration test to make sure nothing unexpected was happening. The CSS was the last thing added to the project as with the agile principles the focus was to get it working, keep it simple, refactor the code and then add the style at the end. Then full system testing was undertaken. This was followed by a demonstration of the operation of the completed system for final validation by my supervisor.

Detailed Design

HCI

The menu and page layout were something I believed to be central to the success of the system from an HCI standpoint. Early on I made the decision that the menu would always be visible to the user, this is because on a system like this it is highly likely you will want to jump around from page to page never staying on the same page for more than 10 mins therefore being able to quickly transition thought the pages was essential(See Figure 3). Putting the menu on the left-hand side was also a conscious decision as most of the pages such as the People, Messaging, Documents and Calendar have most of the interaction buttons on the left-hand side of the pages. This means less mouse travel around the screen and it allows the users eyes to do the natural movement and read from left to right. Moving your wrist slightly from side to side is a far more natural movement than moving your arm forward and backwards too.

Putting the menu to the side also frees up the very top of the screen this can be taken up with the search bar, displaying the time and date of the other offices around the world and profile information where the user can click quickly edit their profile or log out. This information is far better suited to the top corners of the screen as it is rarely going to be clicked on but regularly viewed by the users.

With the menu around the screen and always visible the actual pages would be centred within the screen with a small 15px margin separating the page and the menu from each other, so it's clear that this is where the main information is contained.

Colour & Icons

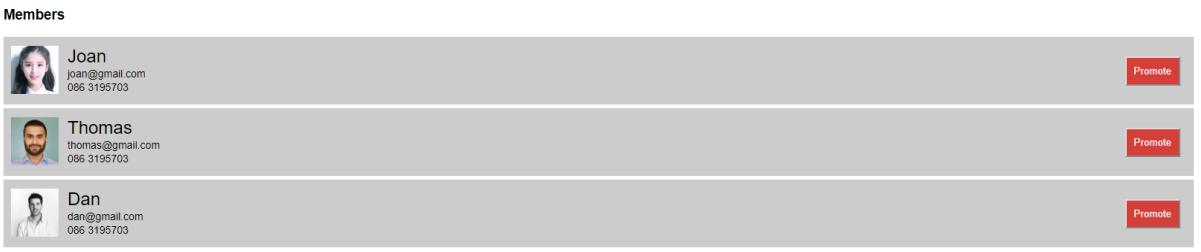
Colour is a great mechanism to draw attention to certain parts of the screen or in some cases to not drawn attention and to just fade into the background.

For the top menu the colour blue was chosen, blue is a colour that symbolizes loyalty, strength and wisdom, blue is known to have a calming effect on the psyche as well (The Color Blue). Because of this calming and welcoming effect blue was used on the top menu, chat messages and on clickable buttons that will add people or events to the system. When in the work environment people can become stressed, this is not when people do their best work, so we need them to be calm. If the primary colour used was red it would have the opposite effect because red symbolisms danger to us.

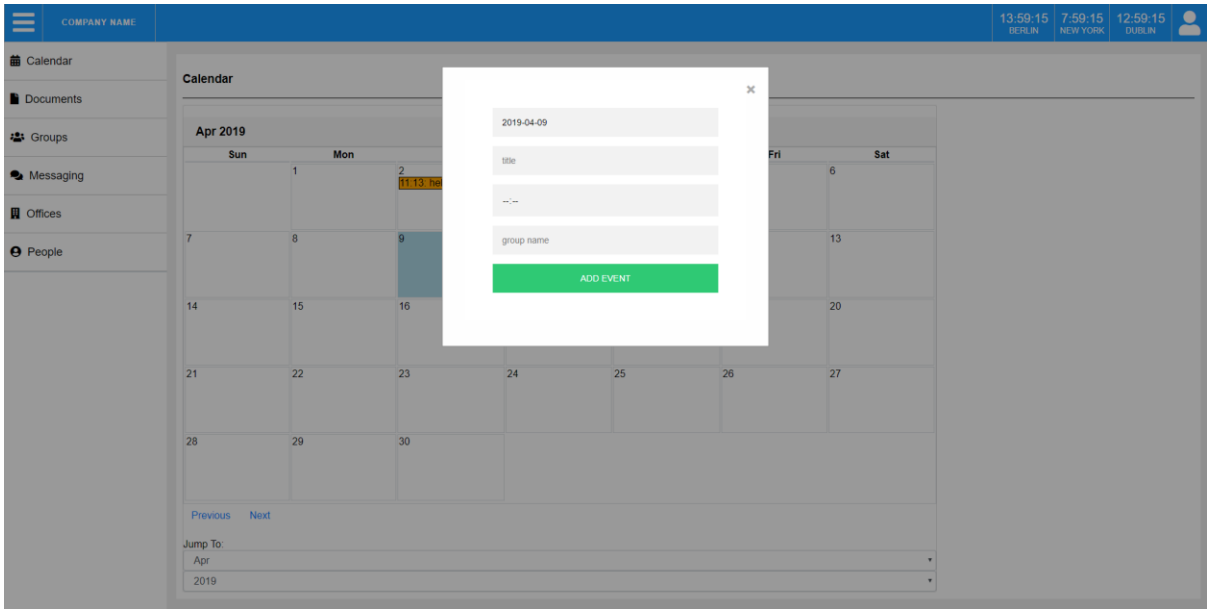
The background of the main windows and menu is white, this is because the menu should be clean and uncluttered and there is no need to add colour when not necessary, white is unimposing, pure and light (The Color White). There is no need to draw attention to these areas the text and highlighting when the mouse hovers over the menu will do that for us.

For the background of users or groups a light grey was used for much the same reasons of white it is unimposing, formal and sophisticated whilst drawing just the right amount of attention to the object in question (The Color Gray).

The colour red was used for buttons that will delete something or someone from the system or to make someone think twice about performing an action like when promoting and member in a group to an admin (The Color Red).

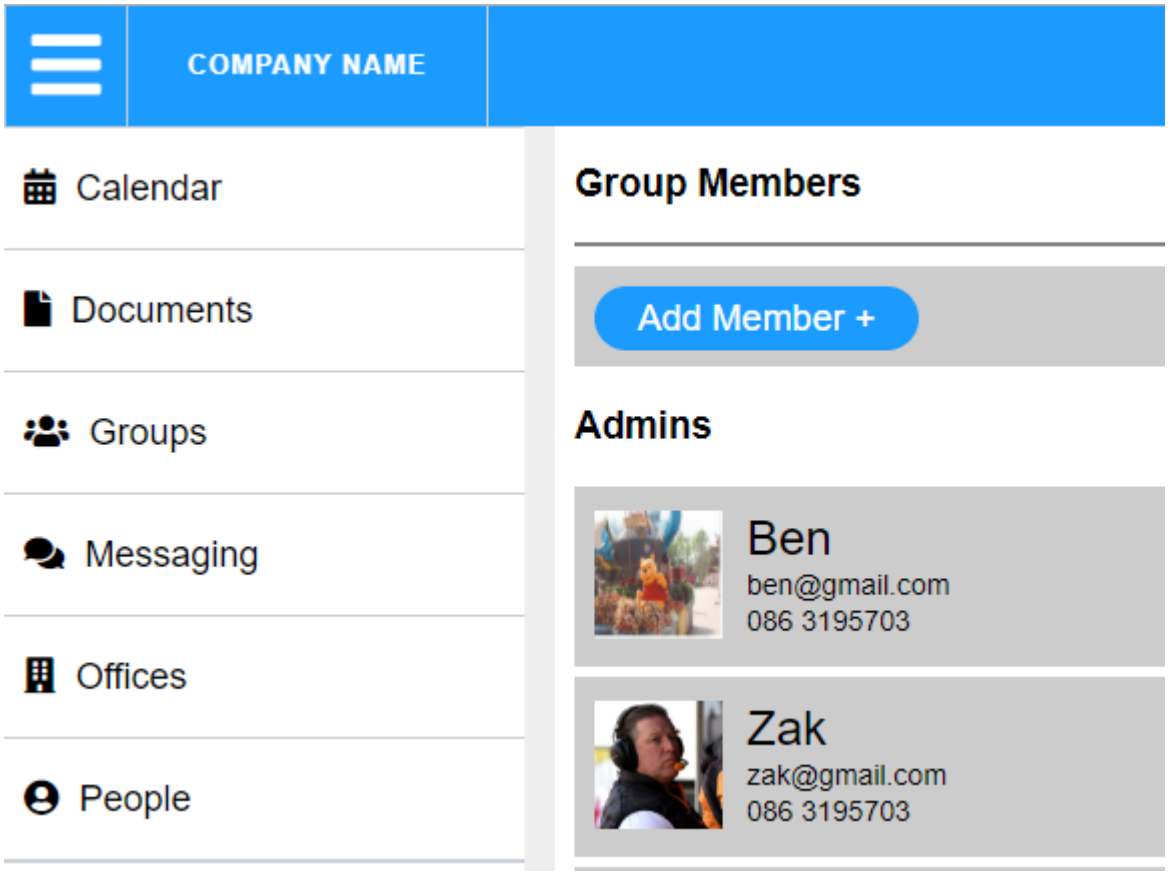


Green is used when user is going to add something or someone to the system, green to us like traffic lights means go and something is about to happen(The Color Green).



In this project colour was used to convey intent without needing to read to much and to improve the users efficiency when using the application.

Whilst colour is great at displaying intent, it means nothing to those who are colour blind that is why icons are used as well to display intent without needing to read too much and increasing the efficiency. Take the icons on the menu for example, the icons display exactly what will be contained in the page when that button is clicked on Calendar show a calendar, Documents shows a file. When in a group page on the add member button a “+” is shown this tells the user what the button will do before they click it.



Security

This software is designed to be used by companies therefore it must be secure. In particular careful attention must be paid to both the login and registering, the storing of passwords and the instant messaging potentially containing sensitive company information.

Login and sessions are handled by two modules in node.js called bcrypt and passport.js. bcrypt is used in login and registering it works by taken the entered password from the user form and encrypting that password with a unique key and can then only be decrypted with that unique key the passwords are then compared to see if they match. All the passwords stored in the database are also encrypted so even if someone got hold of these tables they would be useless. Once logged in a session is created and that session also has a unique key.

Instant messaging is also secure and is handled using a module called socket.io. It is run over SSL (Secure Sockets Layer) together with the session key so it is encrypted over the wire and secure until it meets the correct user with the correct key to decrypt the message.

To have access to any page in the system the user must be logged in, even if someone knows the URL to download a specific file it won't download it.

Database Design

Each week when a new feature would be undertaken the database would be design for that feature. I quick sketch was done first to visualise the relationships between tables (See Figure 7) and then there was an attempt at implementation where any necessary changes could be made easily. Thought was given to each table so that it could be expanded in the future. The database was designed using third normal form (3NF).

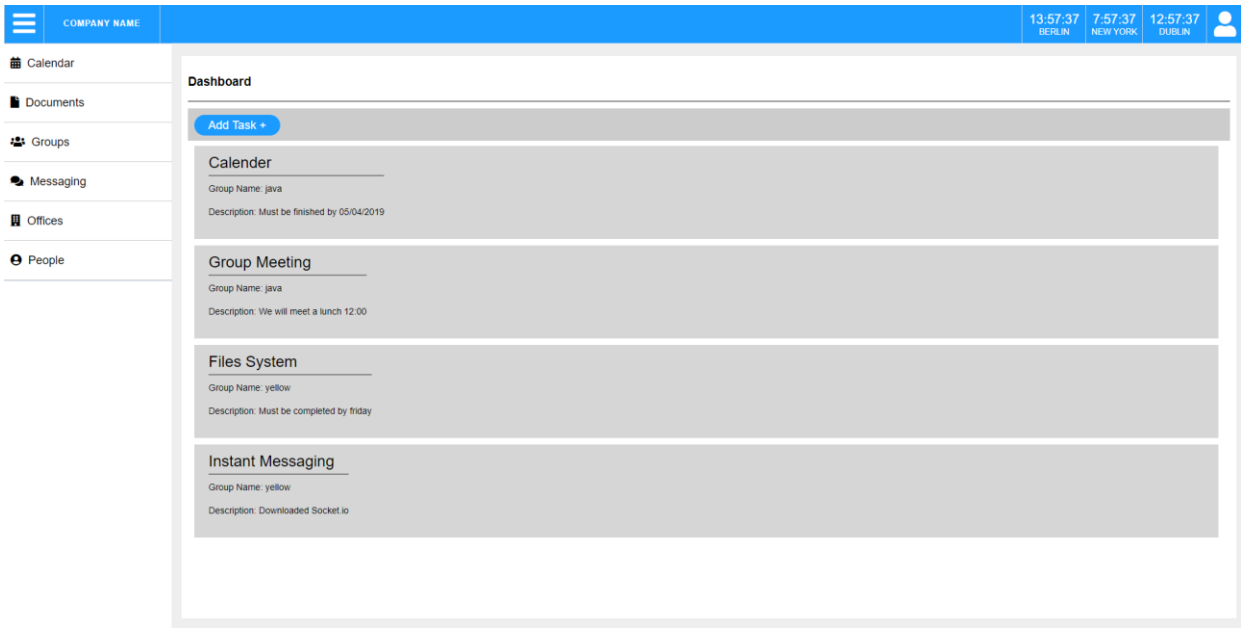
Problems with Implementation

Node.js had a big learning curve for me, because I had never used it before a lot of the concepts node.js uses was a bit foreign. PHP is quite easy to use you can just put .php on the end and put it on an apache server and it will work and display its content. Node.js however uses and require modules to have the same functionality, a large part of the first semester was spent trying to figure this out and become familiar with all the concepts. Trying to create a secure login was a challenge, the modules needed to this was bcrypt-nodejs this was to encrypt the information going to and from the server. Passport.js is an authentication middleware. Express is the web framework for Node.js. Express-session is used to store the session variables. EJS is the view engine which renders the .ejs files (similar to HTML or PHP files (See Figure 9)). Finally, the mysql module is needed to send and retrieve information from the SQL database. Having never worked like this before it was quite overwhelming and took some getting used to. Thankfully a lot of the modules needed for just the login are used for the rest of the system so its not a new learning experience when developing a new part of the system. The file management, calendar and messaging provided their own challenges because I had never developed anything like them before I truly had to start from scratch and get used to working with the modules that would make their development possible like express-fileupload, moment.js and socket.io. Once these quirks of Node.js were understood the development of the application became much easier, quicker and enjoyable.

Product Features

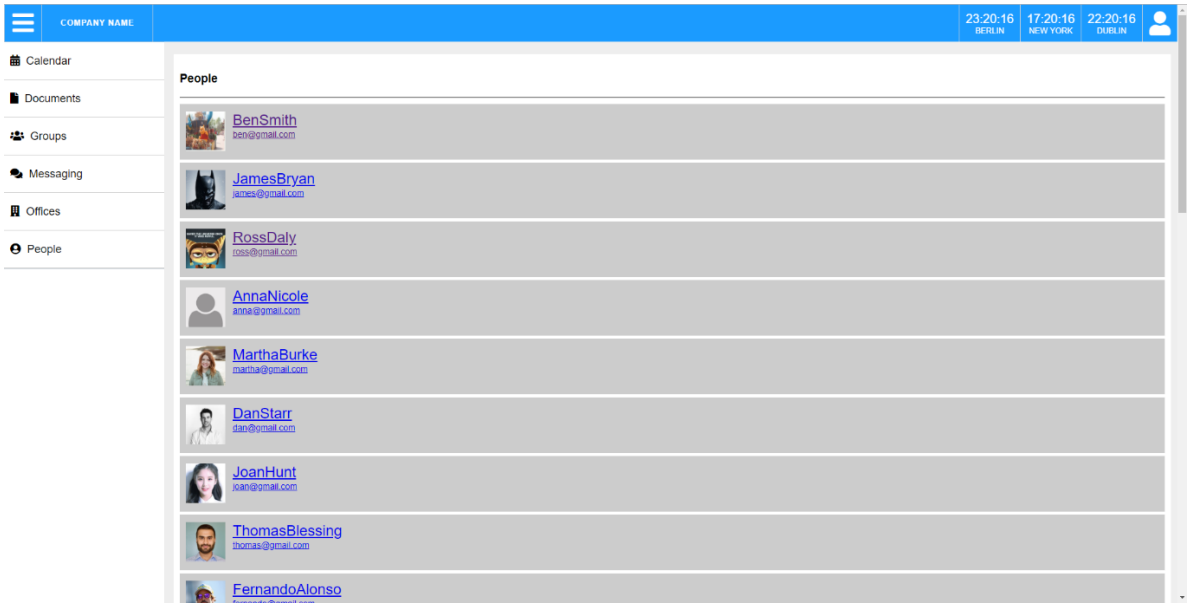
Dashboard

The dashboard is the landing page after login and it provides an over view of upcoming tasks from the calendar recent messages from people or groups.



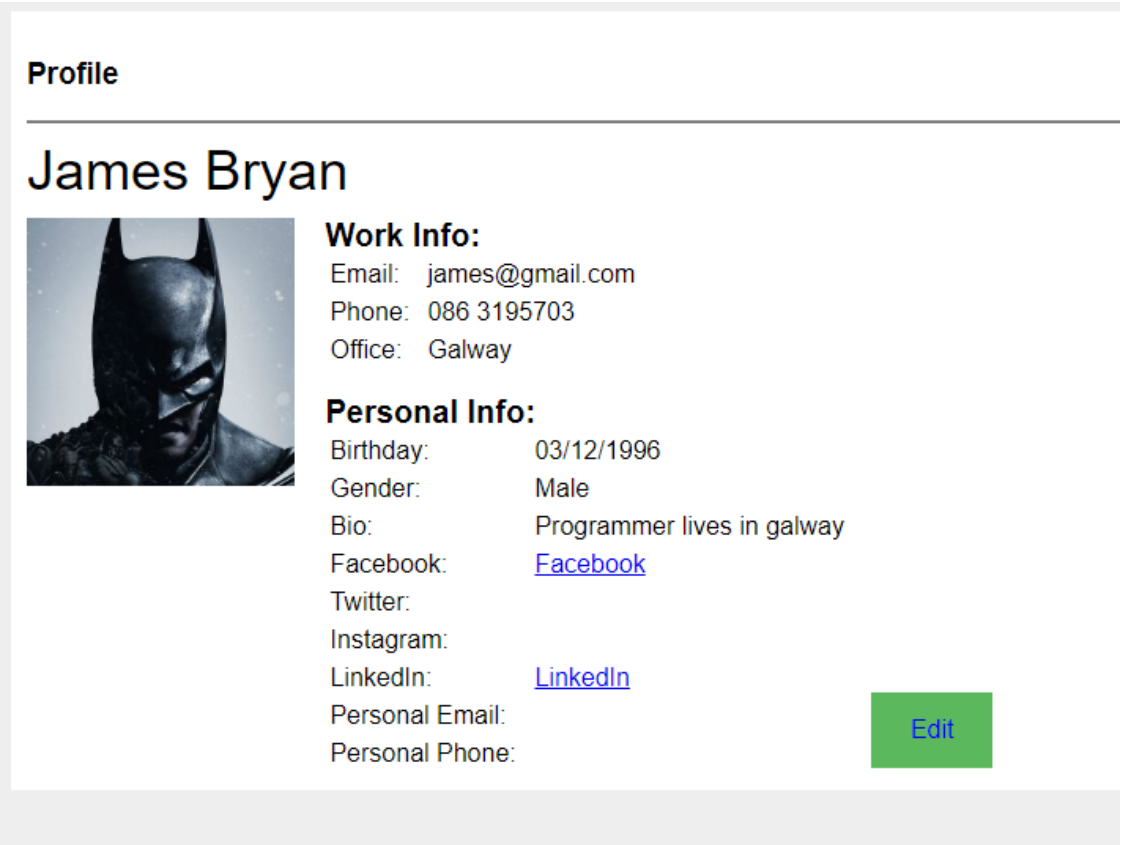
People

The people page gives an overview of everyone in the company stating their name, email, phone number the office that they work at and displaying the profile picture. Clicking on these profiles will take you to a detailed profile page of that user. Visible to admins only there is a “add user” button where they provide an email, first and last name of the user they want to add. An email is then sent to that user where they can login and set their own password.



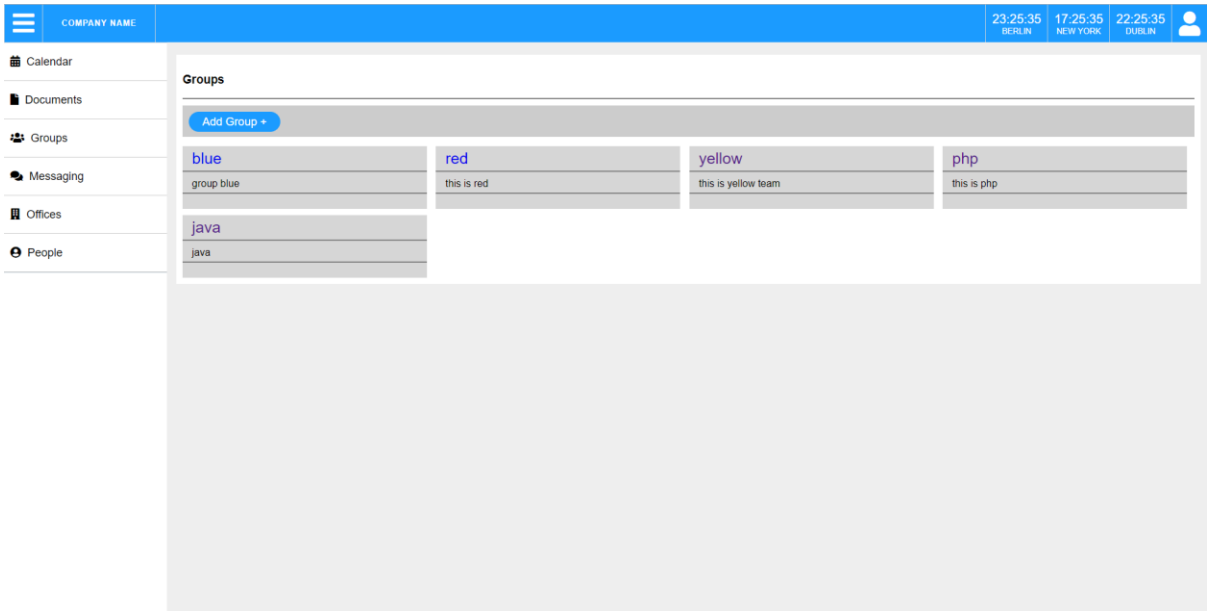
Profile

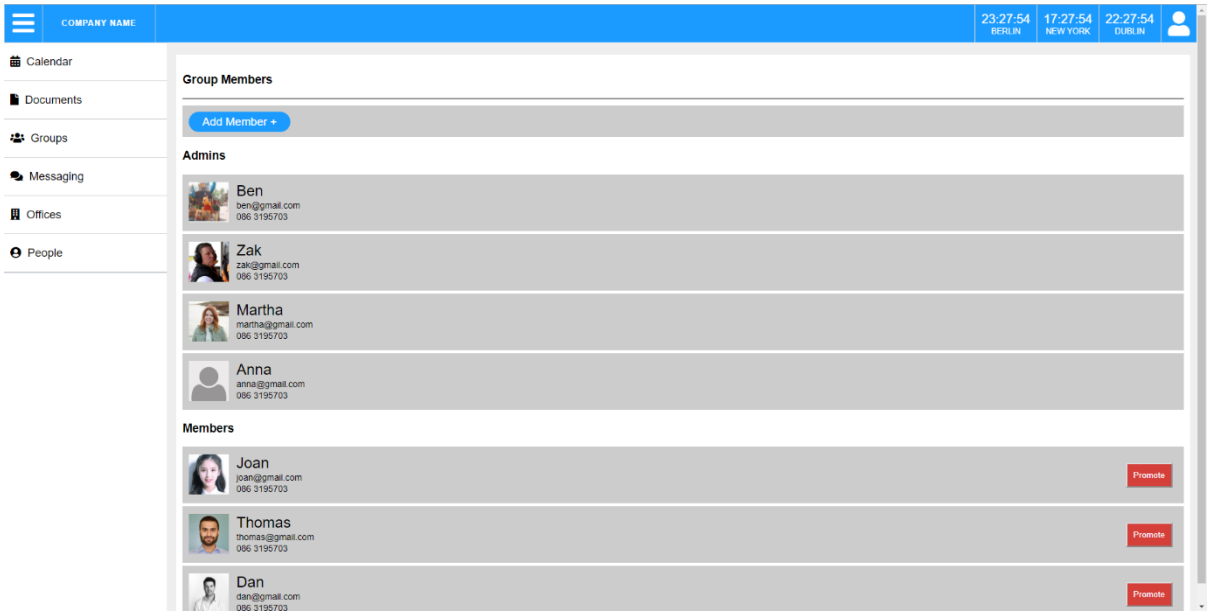
The profile page provides a detailed overview of a user such as their age, gender a short personal description of themselves and social media pages like Facebook, Twitter, Instagram and LinkedIn. This will help create a closer bond between team members especially those in different countries as things can become more informal and people become easier to relate with. Each user can only edit their own profile however admins can edit all profiles and have the ability to delete the user from the system.



Groups

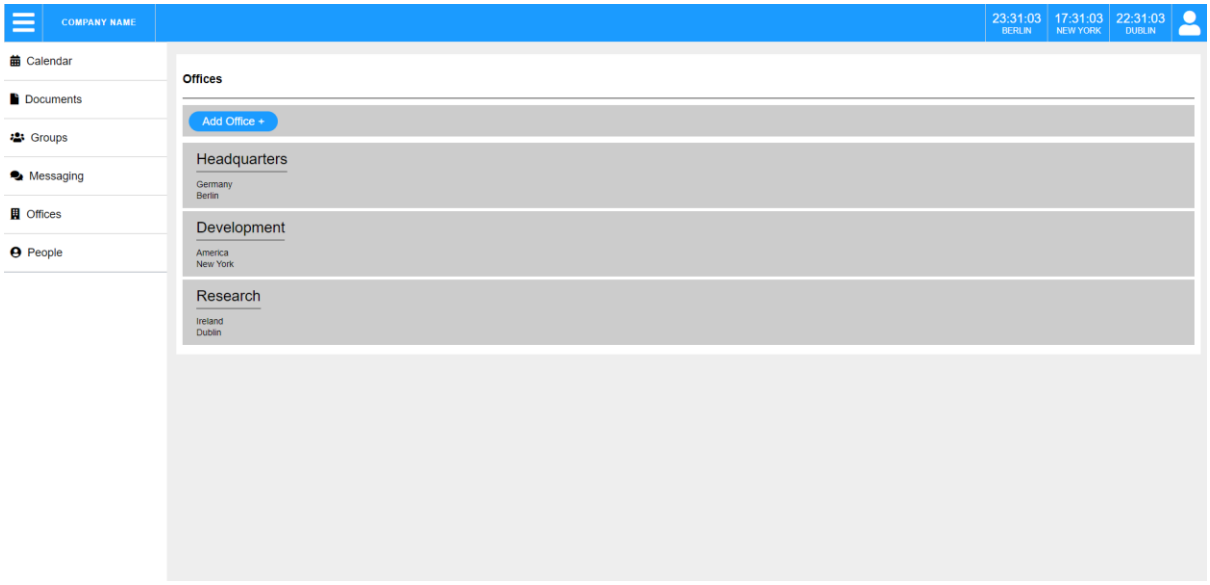
The groups page provides an overview of the groups within the company the displaying the name of the group the admins and the users again clicking on the group shows a page similar to the people page except it is split into groups admins and members. Groups admins have the ability add or remove members and can also promote members to group admins. (Note: Groups admins and Site admins are not the same thing)





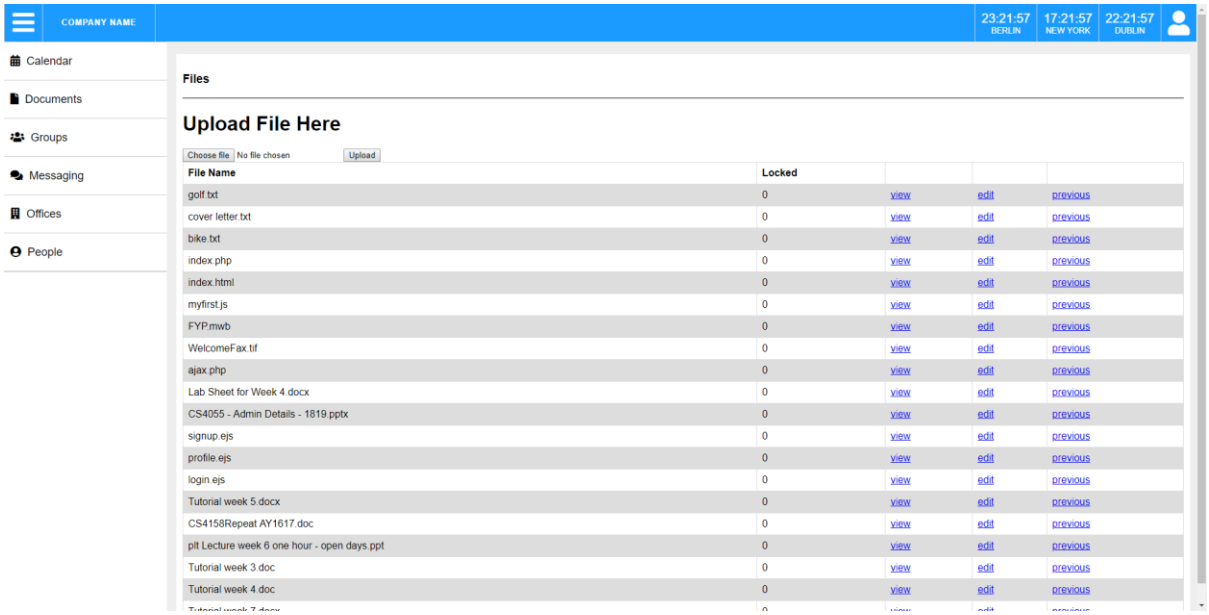
Offices

The office page lists the locations of the offices that the company currently operates out of. Once an office is adding by a Site admin a group is automatically created with the offices name and when editing their profiles users have can then add themselves to the office group. The office section adds the time of that office to the top menu and puts the public holidays of the office in the calendar.



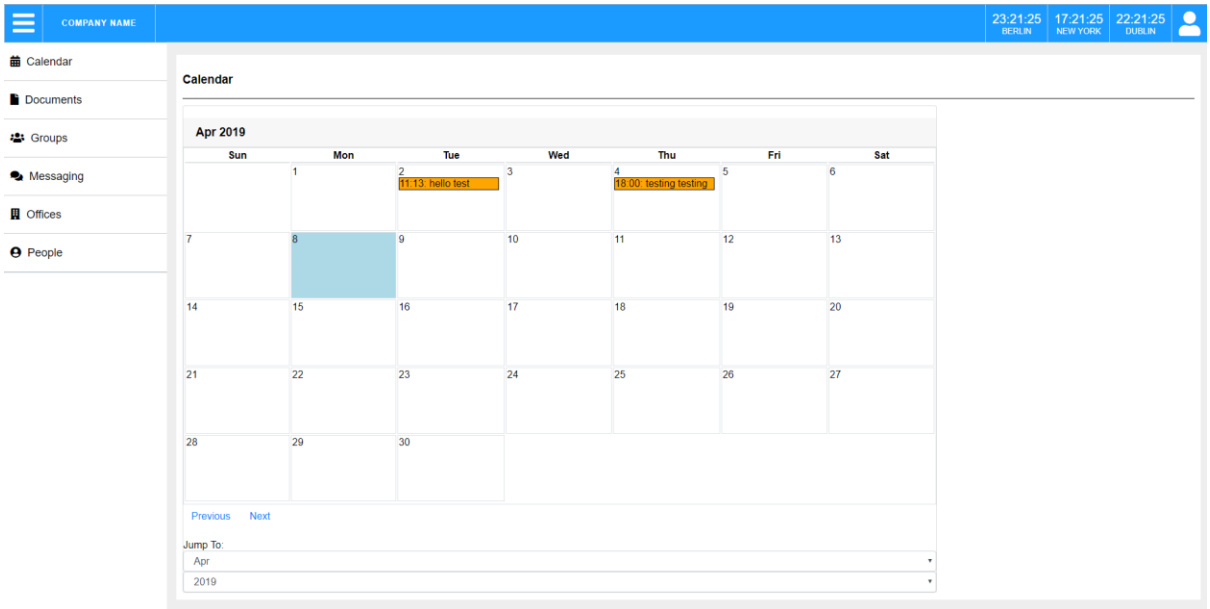
Documents

Documents is the file management part of the system here each group has a folder and only members of that group can enter that folder to either view or edit files within it. Within the folder files are listed by their name and beside them are three buttons which are “view”, “edit” and “previous versions”. View downloads the file, Edit downloads the file but places a lock on that file so no one can edit that file until the user that is editing it uploads the new version of the file then the lock on that file is remove and other people can now edit it. Previous versions list all the old versions of the file saying when they were last edited. Only the group admins can delete a file, the deleted file is then put into the recycling folder for 30 days until it is automatically and permanently deleted.



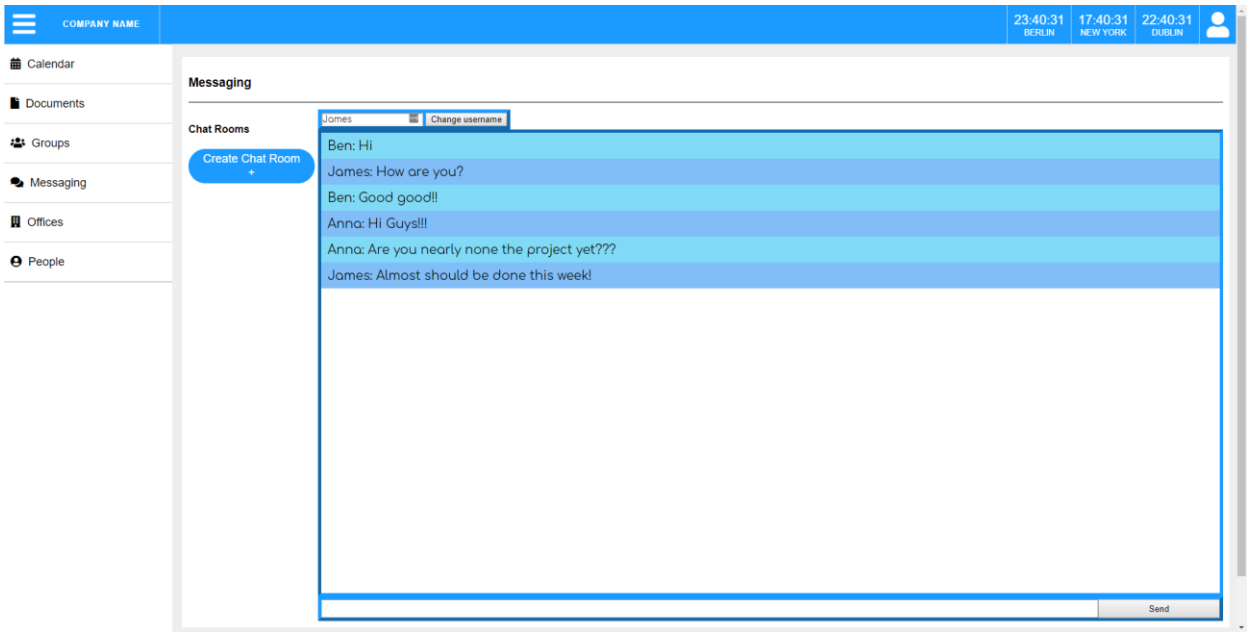
Calendar

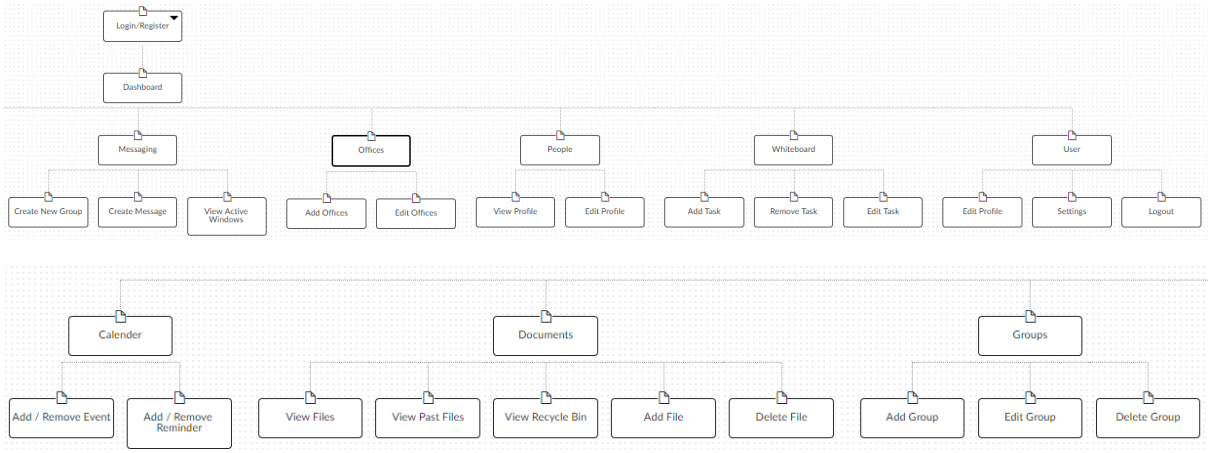
Calendar, this is personalized to each user where they can add their own dates and reminders which can be removed or edited. Reminders can also be set for groups and members of that group will have that date added to their calendar. Listed too will be the public holidays for each office.



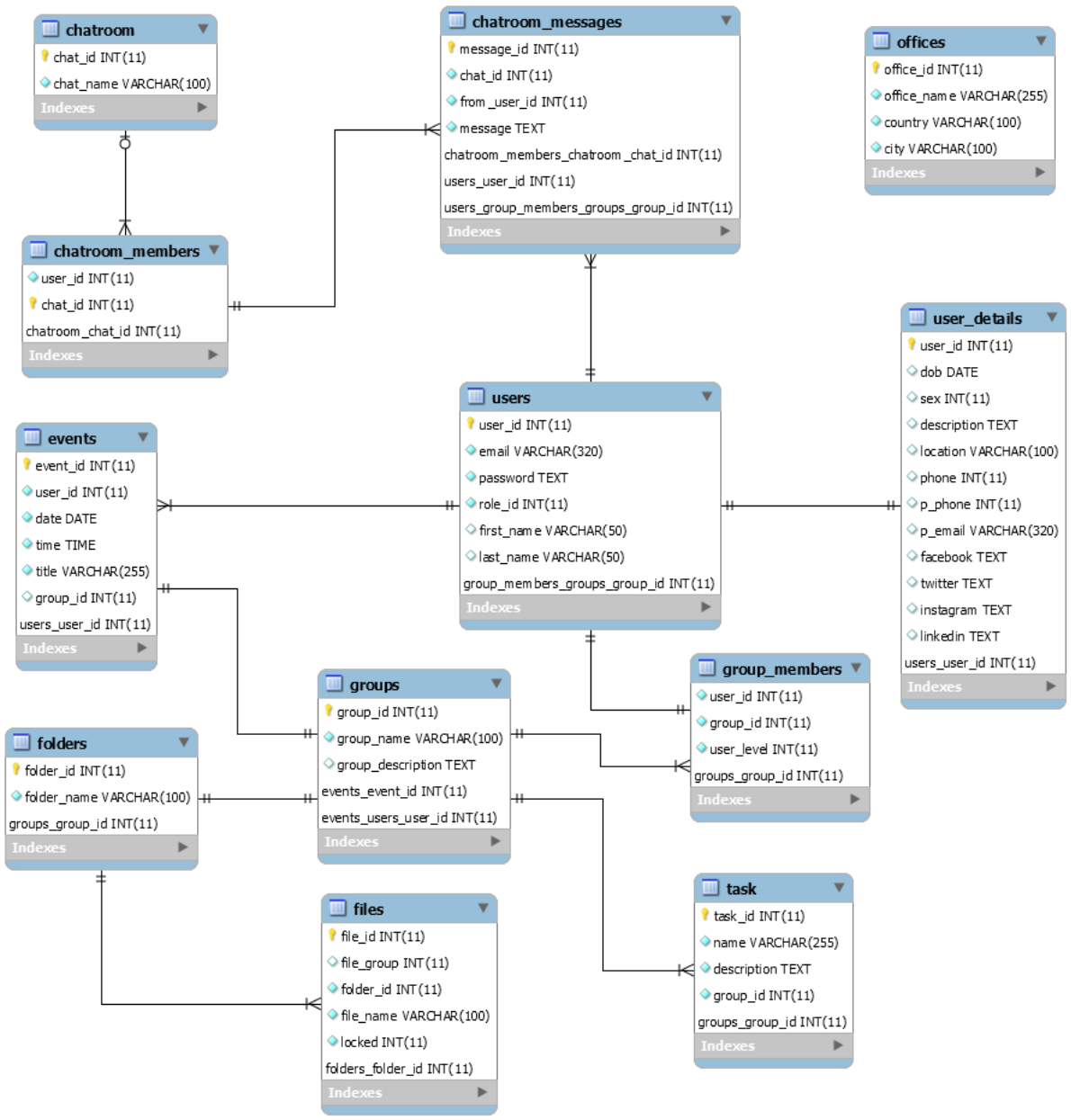
Messaging

The messaging page is where the instant messaging takes place groups and people are listed down the left-hand side of the page and on the right is the messaging window where user communicate with each other. Clicking on a user or group on the left-hand side will reveal the chat history in the messaging window. Users can also create their own messaging groups and add and remove people from the chat room. The created messaging groups are not to be confused with the groups discussed earlier. The messages sent between users is encrypted.





The Projects Site Map



ER Diagram for SQL Database

Future Work

In the future I believe there is a lot that could be done to this project to grow it and make it something that could be used by companies that have a globally distributed work force. In its current form it is quite a general system but could be customised to make it organisation specific with certain features added such as a GitHub integration to share code in one easy to find place. I would also like to add and area where people could report bugs within the code. An employee forum could too be added this could be used for both work related queries and organising activities outside of work. Within this forum I would to like to add an anonymous mode where employees could ask questions or put forward suggestions without revealing their identity. I think if used correctly it could help productivity and help break through cultural barriers by being able to confront issues with the help of anonymity. Because this software is designed to be used by a globally distributed work force I would like users to choose their preferred language, because while English is the global standard for software development some people may be more efficient in navigating the application in another language of their choice. For the groups I would like them to be assigned a colour this would help to distinguish them from each other and if the corresponding colour was displayed in the calendar this would again provide clarity and efficiency when users look at their calendar.

Results and Evaluations

Overall, I was happy with how the project went but I didn't get all the work I wanted to done. Due to time constraints I wasn't able to create a detailed task, event and knowledge management feature that would have provided a great deal functionality and overall effectiveness to the application. I would to have liked the system to be more responsive too, for example, when adding a file to a folder the file is not displayed in the table instantly, so the user doesn't get that instant feedback that it has been uploaded successfully this is something that I want to fix.

I am very happy with the rest of the system the calendar, file management system, groups, messaging, offices and people. They have met the necessary requirements for them to work well and deliver the required functionality.

I feel happy and confident that I have developed a web-based system that has met the conditions set from it to be a global software development system, that could be modified and adapted to meet specific customer needs. Thanks to it being developed with an agile manner it can be extended further without any trouble.

Conclusion

Throughout my time working on this project I have learned a great deal about GSD its benefits in downfalls and why it is so prevalent, the knowledge gained from this project will be of huge benefit to me when I go into the workforce. I have too concluded that Global Software Development can be greatly improved by developing a specific product to address the problems associated with GSD.

Learning about Node.js has been a huge gain from this project it has expanded my knowledge of JavaScript and web-based technologies a vast amount. I now feel very confident in working will JavaScript and Node.js and given the ever-increasing prevalence of the Node.js technology I am extremely proud and happy I choose to learn it from scratch.

HCI was a critical component of this project and one I have learned a lot from. Being able to create a system that is responsive, inviting and function is an incredibly useful skill and something that I put a lot of thought and effort into.

I have gained considerable knowledge in requirements elicitation and analysis. If done right requirements elicitation and analysis can define the scope of the project and avoid the scope creep that could throw the develop of track. Chasing features that may seem good at the time, but in reality, if you talked to the customers or looked at it in a critical manner, the feature provides little to no benefit and development time could be best spent elsewhere.

This project has too taught me about agile development which I had never used before and using it from project conception to completion I have learned a lot. With my supervisor acting as the customer we were able to decide which features are necessary and, in our meetings, we were able to decide how to improve the features. Developing the system in an iterative and incremental fashion, verifying and validating along the way not only creates more complete features and speeds up development but it allows for the expansion of the system.

With the knowledge I have gained from researching the literature and completing the product. I can conduct research in a far more effective manner than I could at the start of this project. I believe that I have become a better programmer by completing a successful and useful product I have proved that I am able to take on new concepts learn about them and apply them in the real world.

References

Paulish, D.J., Pichler, R. (2004) Agile Practices for Global Software Development
available: https://www.romanpichler.com/articles/pdfs/GlobalSWDevPaulishPichler.pdf
[accessed 23 Dec 2018]

Prikladnicki, R., Nicolas Audy , J.L., Evaristo , R. (2003)Global Software Development in Practice Lessons Learned
available: https://pdfs.semanticscholar.org/686f/5e78953121a049f6280f312c95c5d83da770.pdf
[accessed 31 Dec 2018]

Muthuswamy, B. (2003) GLOBAL SOFTWARE DEVELOPMENT: STRATEGIC IMPLICATIONS FOR U.S. INFORMATION SYSTEMS ACADEMIC PROGRAMS
available:
http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=D842C141963284FF5C80680FD64674D4?doi=10.1.1.109.4613&rep=rep1&type=pdf
[accessed 29 Dec 2018]

Casey , V., Richardson, I. () Uncovering the Reality Within Virtual Software Teams
available: http://seal.ece.ubc.ca/gsd2006/slides/papers/GSD04-Casey.pdf
[accessed 31 Dec 2018]

WHAT IS AGILE METHODOLOGY
[Online] Available at: https://luis-goncalves.com/what-is-agile-methodology/
[Accessed 31 Dec 2018]

Agile Alliance
[Online] Avalible at: https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto
[Accessed 3 Mar 2019]

The Color Blue
[Online] Avalible at: https://www.color-meanings.com/blue-color-meaning-the-color-blue/
[Accessed 3 Mar 2019]

MongoDB
[Online] Avalible at: https://www.tutorialspoint.com/mongodb/
[Accessed 3 Mar 2019]

Event-Driven Architecture: A Primer
[Online] Avalible at: https://medium.com/high-alpha/event-driven-architecture-a-primer-f636395d0295
[Accessed 3 Mar 2019]

The Color White
[Online] Avalible at: https://www.color-meanings.com/white-color-meaning-the-color-white/
[Accessed 3 Mar 2019]

The Color Green
[Online] Avalible at: https://www.color-meanings.com/green-color-meaning-the-color-green/
[Accessed 3 Mar 2019]

The Color Red
[Online] Avalible at: https://www.color-meanings.com/red-color-meaning-the-color-red/
[Accessed 3 Mar 2019]

The Color Gray
[Online] Avalible at: https://www.color-meanings.com/gray-color-meaning-the-color-gray/
[Accessed 3 Mar 2019]

Britrix24
[Online] Avalible at: https://www.bitrix24.com/
[Accessed 3 Mar 2019]

Comparison: Node.js vs. PHP
[Online] Avalible at: https://stackify.com/node-js-vs-php/
[Accessed 3 Mar 2019]

Node.js
[Online] Avalible at: https://www.tutorialspoint.com/nodejs/nodejs_quick_guide.htm
[Accessed 3 Mar 2019]

Trello
[Online] Avalible at: https://trello.com/en
[Accessed 3 Mar 2019]

Sharepoint
[Online] Avalible at: https://products.office.com/en-us/sharepoint/collaboration
[Accessed 3 Mar 2019]

Manifesto for Agile Software Development
[Online] Available at: http://agilemanifesto.org/iso/en/manifesto.html
[Accessed 31 Dec 2018]

Dmitry, F., 8 Characteristics Of Successful User Interfaces
[Online] Available at: http://usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces/
[Accessed 31 Dec 2018]

Human-Computer Interaction (HCI)
[Online] Available at: https://www.interaction-design.org/literature/topics/human-computer-interaction
[Accessed 31 Dec 2018]

Continuous Feedback in Agile Teams
[Online] Available at: https://www.infoq.com/articles/continuous-feedback-teams
[Accessed 31 Dec 2018]

event-driven architecture (EDA)
[Online] Available at: https://searchmicroservices.techtarget.com/definition/event-driven-architecture-EDA
[Accessed 31 Dec 2018]

Casey, V. and Richardson, I., 2009. Implementation of Global Software Development: a structured approach. Software Process: Improvement and Practice, 14(5), pp.247-262.

Casey, V., 2010. Imparting the importance of culture to global software development. ACM inroads, 1(3), pp.51-57.

Richardson, I., O'Riordan, M., Casey, V., Meehan, B. and Mistrik, I., 2009, July. Knowledge management in the global software engineering environment. In 2009 Fourth IEEE International Conference on Global Software Engineering (pp. 367-369). IEEE.

Casey, V. and Richardson, I., 2008, August. The impact of fear on the operation of virtual teams. In 2008 IEEE International Conference on Global Software Engineering (pp. 163-172). IEEE.

Casey, V. and Richardson, I., 2006, October. Project management within virtual software teams. In 2006 IEEE International Conference on Global Software Engineering (ICGSE'06) (pp. 33-42). IEEE.

Sivakumar, M.S., Casey, V., McCaffery, F. and Coleman, G., 2011, June. Improving verification & validation in the medical device domain. In European Conference on Software Process Improvement (pp. 61-71). Springer, Berlin, Heidelberg.

Bibliography

Casey, V 2008, Software Testing and Global Industry: Future Paradigms.

Building Great Work Relationships
[Online] Available at: https://www.mindtools.com/pages/article/good-relationships.htm
[Accessed 31 Dec 2018]

25 Best Team Management Software For Maximising Team Productivity
[Online] Available at: https://www.scoro.com/blog/best-team-management-software/
[Accessed 31 Dec 2018]

Agile Methodology: The Complete Guide to Understanding Agile Testing
[Online] Available at: https://www.qasymphony.com/blog/agile-methodology-guide-agile-testing/
[Accessed 31 Dec 2018]

Moore, S., Barnett, L. (2004) Offshore Outsourcing And Agile Development
available:
http://www.attitudeweb.be/doc/resources/studies/offshore_outsourcing_and_agile_development_en.pdf
[accessed 01 Jan 2019]

Why Whatsapp's design makes it the best instant messenger
[Online] Available at: https://medium.muz.li/why-whatsapps-design-makes-it-the-best-instant-messenger-aa706bf7a41d
[Accessed 31 Dec 2018]

Beyond Google Calendar and Apple Calendar: The 18 Best Apps to Manage Your Schedule
[Online] Available at: https://zapier.com/blog/best-calendar-apps/
[Accessed 31 Dec 2018]

The top 5 software architecture patterns: How to make the right choice
[Online] Available at: https://techbeacon.com/top-5-software-architecture-patterns-how-make-right-choice
[Accessed 31 Dec 2018]

Jabangwe, R., Nurdiani , I. (2010) Global Software Development Challenges and Mitigation Strategies
available: https://www.diva-portal.org/smash/get/diva2:831201/FULLTEXT01.pdf
[accessed 31 Dec 2018]

Val Casey
[Online] Available at: http://valcasey.eu/Publications.html
[Accessed 05 Mar 2019]

Appendices

Project Plan Task Name	Start Date	End Date	Completed
Decide on FYP	10/09/2018	14/09/2018	100%
Sign Supervisor	18/09/2018	18/09/2018	100%
Decide Technologies	14/09/2018	20/09/2018	100%
Initial reseach	18/09/2018	25/09/2018	100%
Select features that will be used	18/09/2018	25/09/2018	100%
Define scope of project	25/09/2018	02/10/2018	100%
Basic design concepts	25/09/2018	02/10/2018	100%
Interim report	25/09/2018	02/01/2019	100%
Prepare for presentation	25/09/2018	15/10/2018	100%
Decide on nessessary pages of website	02/10/2018	05/10/2018	100%
Design database base schemas	05/10/2018	09/10/2018	100%
Further research on HCI	02/10/2018	09/10/2018	100%
Research what makes a good IM	02/10/2018	16/10/2018	100%
Research what makes a good Calendar	02/10/2018	16/10/2018	100%
Research Document Control	02/10/2018	16/10/2018	100%
Research visual team management	02/10/2018	16/10/2018	100%
Reasearch what makes people connect	02/10/2018	16/10/2018	100%
Week 6 Presentation			100%
Build sign-in/sign-up page	16/10/2018	19/10/2018	100%
Build Homepage	16/10/2018	23/10/2018	100%
Build Accout page	23/10/2018	30/10/2018	100%
Build Bio Page	30/10/2018	06/11/2018	100%
Build Groups	06/11/2018	13/11/2018	100%
Build Offices	13/11/2018	20/11/2018	100%
Build People	20/11/2018	26/11/2018	100%
Build Instant Messgaing App	02/01/2019	09/01/2019	100%
Build Calendar	09/01/2019	16/01/2019	100%
Build File Sharing	16/01/2019	30/01/2019	100%
Build Document Control	30/01/2019	13/02/2019	100%
CSS Tweaks	27/02/2019	13/03/2019	100%
Draft Report			100%
Demo Day	02/01/2019	07/03/2019	100%
Final Product		02/04/2019	100%
Final Report	02/01/2019	09/04/2019	100%

Figure 1

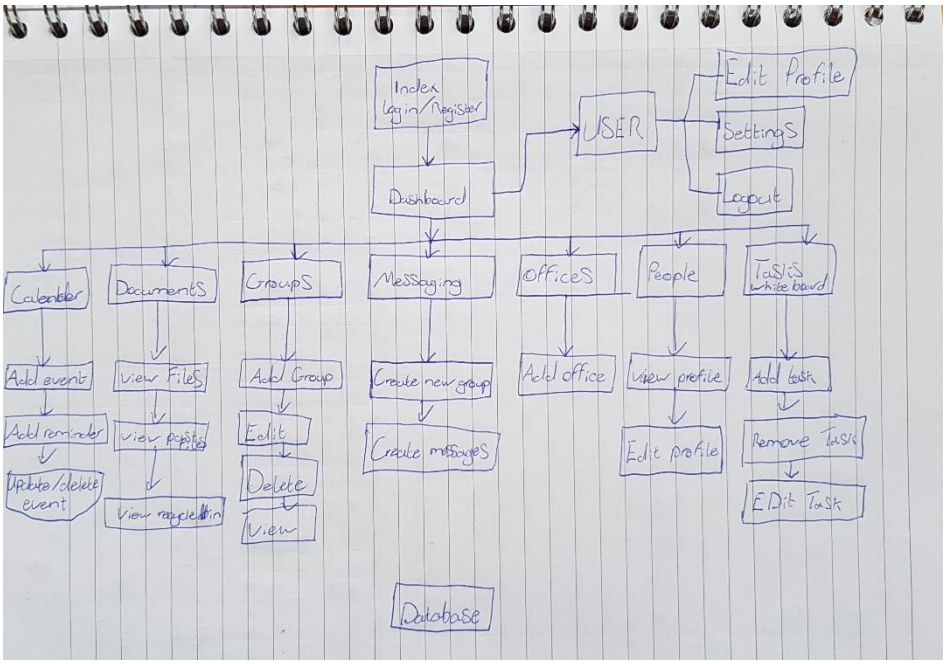


Figure 2: Early Site Map

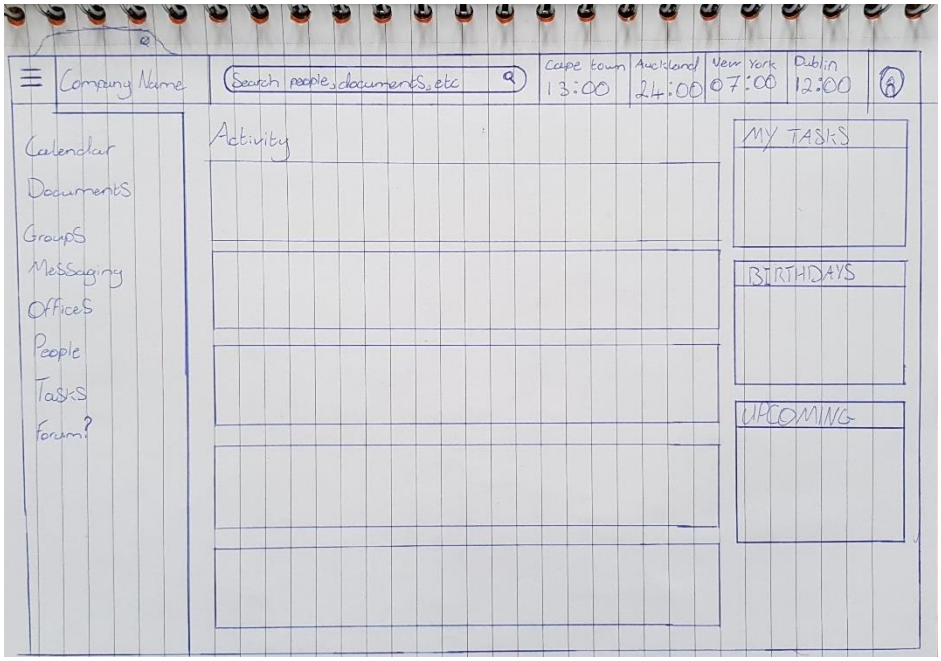


Figure 3: Early UI Design

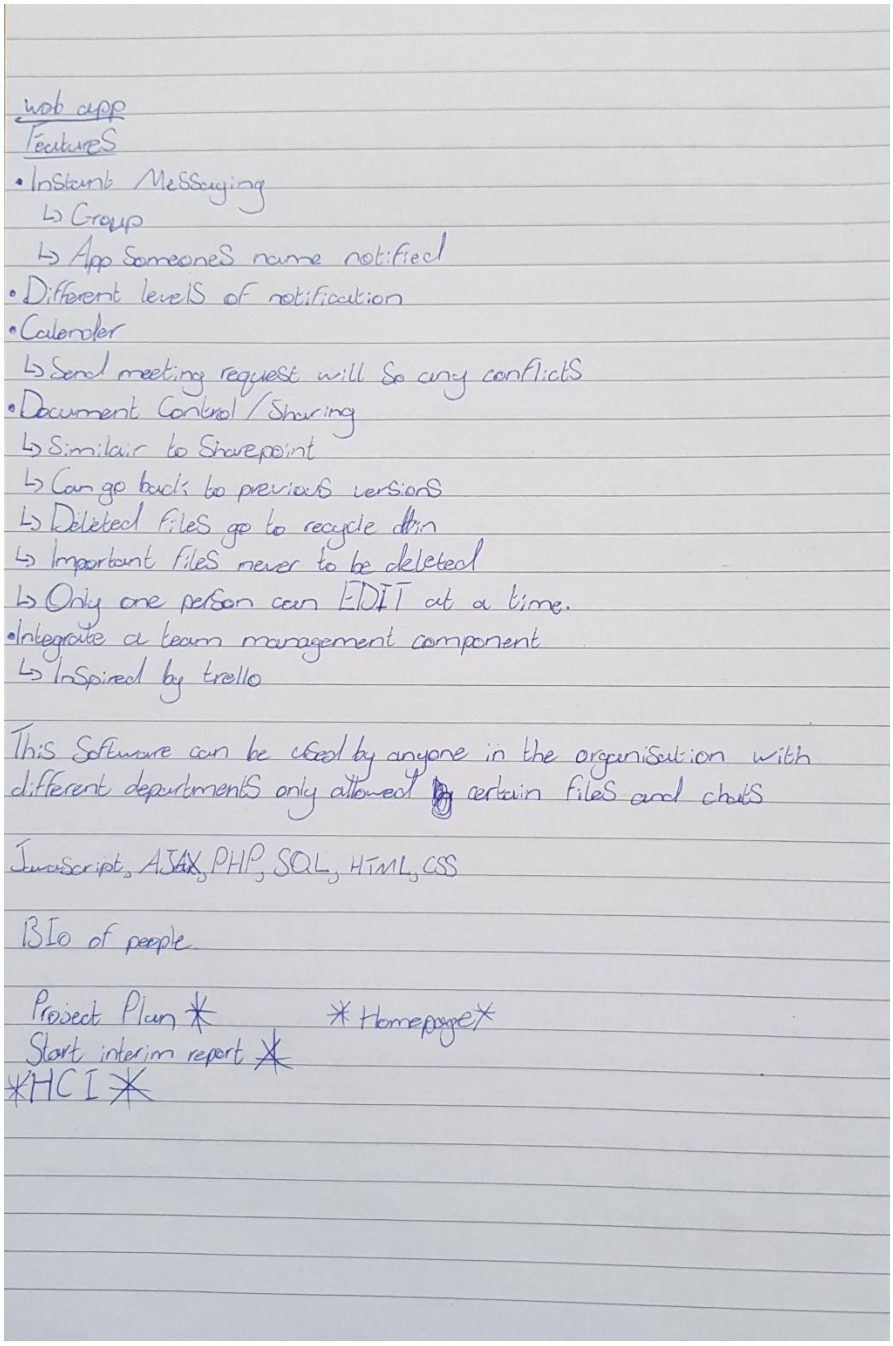


Figure 4: Early Requirements Brainstorm

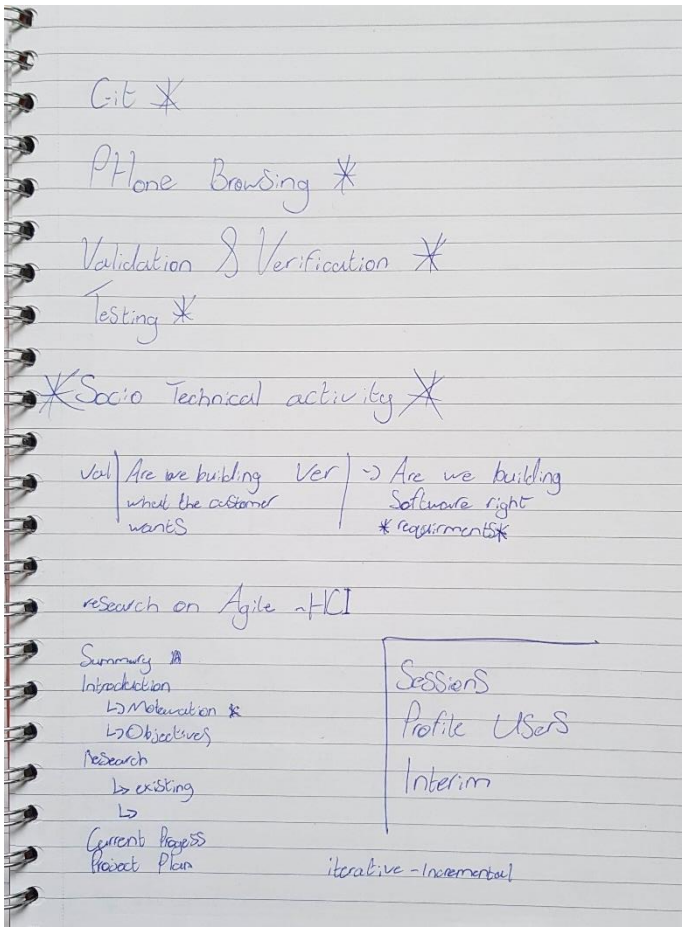


Figure 5: Notes After Supervisor Meeting

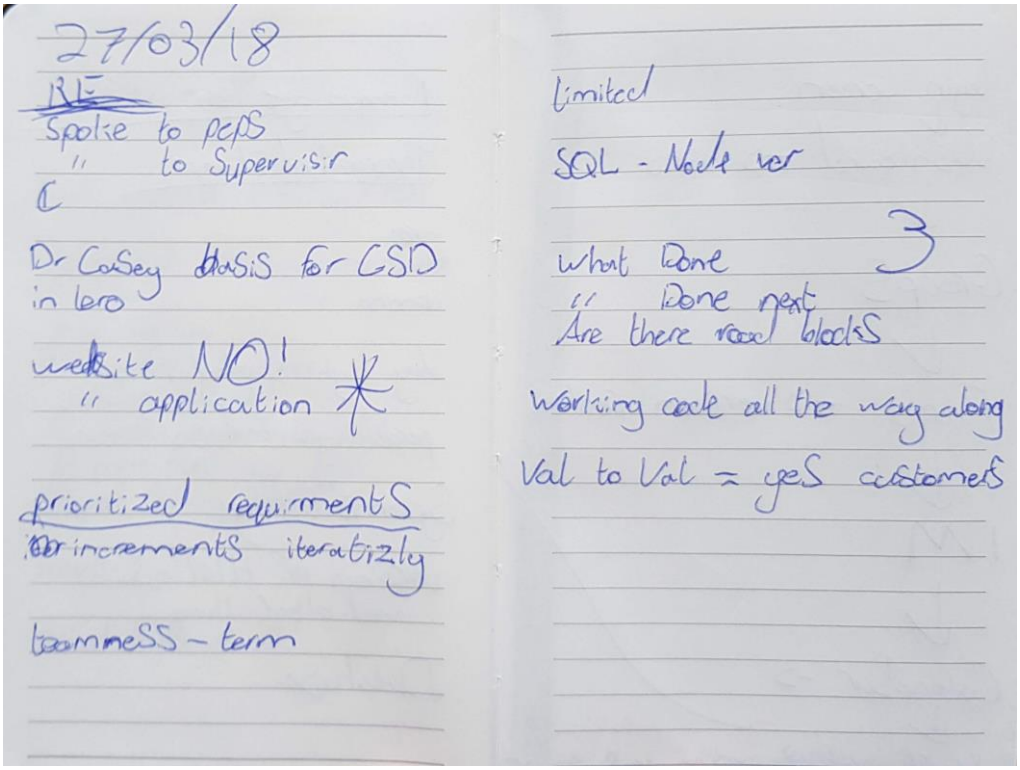


Figure 6: Note after Supervisor Meeting

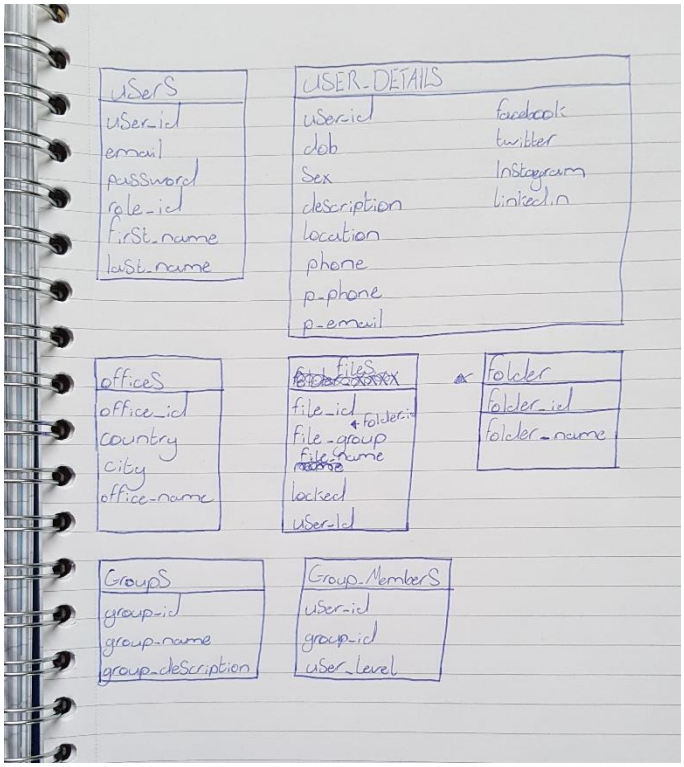


Figure 7: Early Database Schema

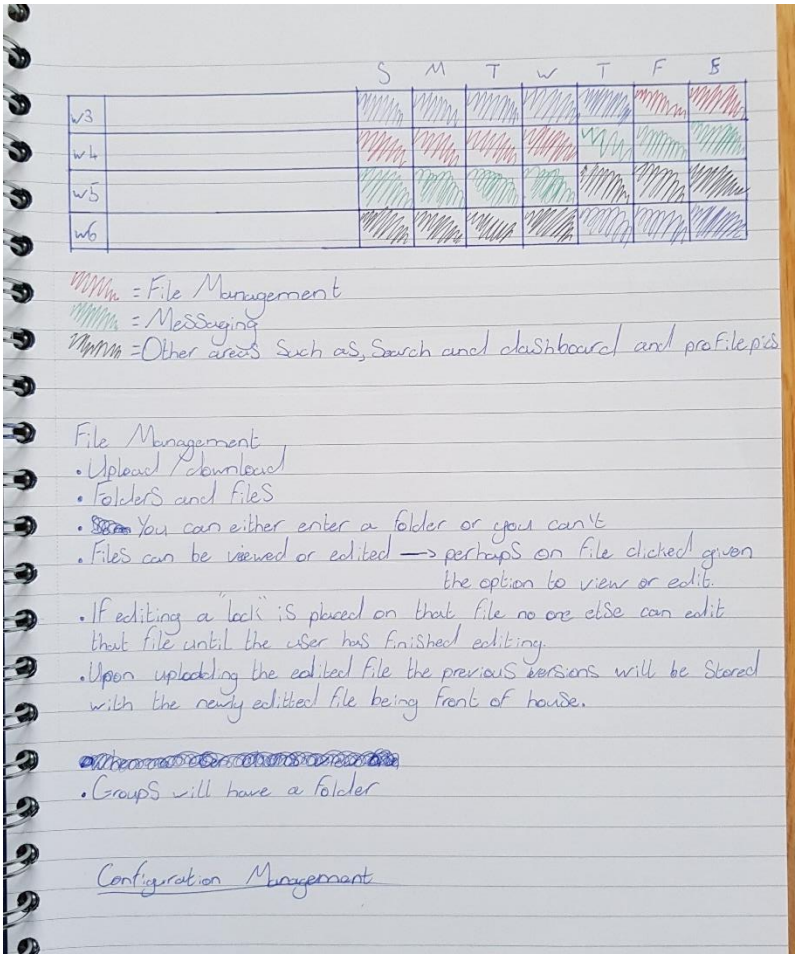


Figure 8: Project Plan and File Management Description

```
<div class="content">
    <h3>People</h3>
    <hr>

    <% if (data) { %>
    <% data.forEach(function(userss){ %>
        <div class="user-row">
            <div class="profile-img">
                <div class="image"><img src="./<%= userss.user_id %>.jpg"></div>
            </div>
            <div class="personal-info">
                <div class="profile-name"><a href="profile/<%= userss.user_id %>"><%= userss.first_name %><%= userss.last_name %></div>
                <div class="email"><%= userss.email %></div>
                <div class="phone"><%= userss.phone %></div>
            </div>
            <div></div>
        </div>

    <% }) %>
    <% } %>

</div>
```

Figure 9: EJS Example

```
    // filing data about month and in the page via DOM.
    monthAndYear.innerHTML = months[month] + " " + year;
    selectYear.value = year;
    selectMonth.value = month;

    // creating all cells
    let date = 1;
    //let dateCompare;
    for (let i = 0; i < 6; i++) {
        // creates a table row
        let row = document.createElement("tr");

        //creating individual cells, filing them up with data.
        for (let j = 0; j < 7; j++) {
            if (i === 0 && j < firstDay) {
                let cell = document.createElement("td");
                let cellText = document.createTextNode("");
                cell.appendChild(cellText);
                row.appendChild(cell);
            }
            else if (date > daysInMonth) {
                break;
            }
            else {
                let cell = document.createElement("td");
                let cellText = document.createTextNode(date);
                cell.setAttribute('class', 'day');
                let sMonth;
                let sDate;
                if(month < 9){
                    sMonth = month + 1;
                    sMonth= '0'+ sMonth; }
                else if(month >= 9){
                    sMonth = month + 1;
                }

                if(date <10){ sDate= '0'+ date; }
                else if(date >= 10){ sDate= date;}

                cell.setAttribute('id', year+'-'+sMonth+'-'+sDate);
                //let dateCompare = year+'-'+(month+1)+'-'+date;
                //dateCompare = dateCompare + 'T23:00:00.000Z';
                //alert(dateCompare);
                var br = document.createElement("br");
                cell.setAttribute('onclick','something(this.id);');
                if (date === today.getDate() && year === today.getFullYear() && month === today.getMonth()) {
                    cell.classList.add("bg-info");
                } // color today's date
                cell.appendChild(cellText);
                row.appendChild(cell);
                date++;
            }
        }
        tbl.appendChild(row); // appending each row into calendar body.
    }
}
```

Figure 10: Generating the Calendar

```
app.get('/profile/:id', isLoggedIn, function(req, res){
    var id = req.params.id;
    connection.query('SELECT * FROM users JOIN user_details ON users.user_id = user_details.user_id WHERE user_details.user_id = ?', [id], function(err, rows, fields) {
        //if(err) throw err
        if (err) {
            req.flash('error', err)
            res.render('profile.ejs', {
                title: id,
                user:req.user,
                data: ''
            })
        } else {
            // render to views/user/list.ejs template file
            console.log(rows)
            res.render('profile.ejs', {
                title: id,
                user:req.user,
                data: rows
            })
        }
    })
});
```

Figure 11: Get the Profile Information and Send it to the EJS File

```
app.post('/addgroup', function(req, res){
    console.log(req.body.g.name);
    console.log(req.body.g.des);
    var fs = require('fs');
    var insertQuery = "INSERT INTO groups (group_name, group_description) values (?, ?)";

    var newGroupMysql = {
      name: req.body.g.name,
      des: req.body.g.des
     };
     connection.query(insertQuery, [newGroupMysql.name, newGroupMysql.des],
      function(err, rows){
       newGroupMysql.id = rows.insertId;

            var user_level = 0;
            insertQuery = "INSERT INTO group_members (user_id, group_id, user_level) values (?, ?, ?)";
            connection.query(insertQuery, [req.user.user_id, newGroupMysql.id, user_level],
            function(err, rows){

                insertQuery = "INSERT INTO folders (folder_id, folder_name) values (?, ?)";
                connection.query(insertQuery, [newGroupMysql.id, newGroupMysql.name],
                function(err, rows){
                    var dir = './uploads/';
                    var result = dir.concat(newGroupMysql.name);
                    console.log(result);
                    //if (!fs.existsSync(result)){
                        //fs.mkdirSync(result);
                        fs.mkdir(result, {recursive: true}, err => {})
                    //}
                if (err) throw err;
                console.log("folder created");
                });

            if (err) throw err;
            console.log("1 record inserted");
            });

       if (err) throw err;
        console.log("lower");
     });
});
```

Figure 12: Add Group and Folder Code

Short description of the various features will do:

Instant Messaging
• Must be able to communicate with anyone in the company
• Must support group messaging
• People must be invited to groups they cannot simply join
• Can be used whilst navigate through other pages of the software
• If the users name is mentioned in the chat they should then be notified of that.
• All communication must be secure

Calendar
• Must be individual to each individual user
• Send a meeting request to a user will so any conflicts e.g. out of office...
• User will be notified of reminders that they set in calendar

Document Sharing and Control
• Only allowed users can access certain files and folders
• Only allowed users can edit/view files
• Only one person can edit a document at a time
• Important files can never be deleted (only with admin permission)
• Deleted files will go the recycle bin for a period before being erased permanently
• The file system will hold previous versions of each file

Employees Personal Pages
• A quick summary of their person
• List birthday, age, profile picture, work email, work phone etc
• Will list social media and their general interests
• Behind a button will also list their emergency contact details

Sample Unit test:

- Unit must meet requirments
- Unit must be bug free
- Unit must be sercure (encrpyt if nessessary)
- Try all possible compinations of clciks an inputs

Sample Integration test:

- Units must meet all existing requirments
- Units must be bug free
- Try all possible compinations of clciks an inputs
- The new Unit must not break the existing Units

Virtual Team Software Development System

Global Software Development

1. GSD is now incredibly common even amongst smaller companies.
2. It provides huge benefits.
    a) Reduction of development expenses.
    b) Speed development.
    c) Modularization of tasks.
3. Software development is a highly sociotechnical activity.
4. Global software development problems.
    a) Poor communication.
    b) Cultural issues.
    c) Knowledge management.
    d) Poor cohesion.
5. These problems can be addressed by better communication and understanding.

Web Based System

- Instant Messaging
- Document control and sharing
- Staff profiles
- Calendar
- Groups
- Offices
- Task assignments
- Real time location information
- Users check-in and out
- Developed with an Agile approach
- Verification and Validation
- Unit, integration and system tested
- Large focus on Human Computer Interaction
- Mechanism's to address to temporal distance
- Humanizes colleagues and develops teamness
- Maximize synchronous communication
- Increases efficiency
- HTML 5
- CSS 3
- JavaScript
- jQuery
- Node.js
- SQL
- Event driven
- Synchronous