Bachelor (Hons) of Science in Computer Science

Creating a messaging client using a decentralised network in order to reduce or replace requirement for servers.

May 2021

Word Count: 25,609

Student: Craig Cargill (1604113)

Supervisor: Dr John Isaacs

Declaration

I confirm that the work contained in this Honours project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed:  Date: 03/05/2021

Abstract

The world has become more connected than ever with the global rise of the internet; more than ever, people across the world have access to information on a scale never imagined only a few decades ago. The internet also unlocked a communication protocol that allows for information and messages to be sent across the world in less than a second. Corporations have monopolised the hosting and transactions of this communication and can analyse and own the data of millions of users. The need for a new protocol has never been greater to help return the ownership of this data to the hands of the users that create it. Current implementations of new protocols often also come with a significant learning curve that is often too complicated for little top-level gain for everyday users. Users often gravitate to messaging services with large user bases and well designed robust feature sets that allow for simple account creation and use. This project aims to create a service that uses a decentralised network to store messages rather than traditional centralised services. The project also aims to marry this technology with an easy to use a simple design that is no more complicated to set up and use than services currently on the market.

Acknowledgements

I would like to thank Dr John Isaacs for providing support and guidance throughout this year. For striving to provide the same level of support and availability especially through the pandemic and working from home. Furthermore, the University and all staff for continuing to provide support and knowledge across all aspects of the course. I would also like to thank my partner Tricia Wagg for supporting me and providing motivation and care throughout the year. My friends, Calum McGuire, Charlie Marsh, and Ryan McConnell, for ensuring my time at university was not only educational but filled with support and friendship. Finally, I would like to thank my family for helping me not only throughout this year but my entire academic and non-academic life.

Contents

1 Introduction

1.1 Introduction Overview

With the world becoming more dependant on technology for everyday life and communication, data ownership has become more important than ever (Aiimi, 2021). Data has become the most valuable resource globally, surpassing oil in 2017 (The Economist, 2017). The most influential companies in the world are the companies that regulate data transfer, communications, and the internet as a whole. With some of the biggest companies in the world such as Netflix, Twitch, and Facebook relying on AWS for some or all of the data hosting (Saunders, 2020). AWS and other data hosting companies have ownership of the internet as we know it. For this reason, new options have to be researched to ensure that users data is owned by that user and not a large corporation (Waters-Lynch, 2018). One critical component of the current internet is communications, with more than 60 billion messages sent every day on WhatsApp alone shown in a study in 2017 (Statista, 2017).

Messaging applications on the internet take many forms; the core principle allows one user to communicate with another. Most traditional messaging services accomplish this; however, the messages are all stored on that services server. A true messaging system with no ownership of the data being held by a corporation is achievable by implementing a decentralised network to store data. With some decentralised messaging applications on the market, the focus is on the technology and what that technology provides, rather than a user experience that all users could take advantage of (O'Reilly, 2021). Some of these applications are also complicated to set up and would not be reasonable to assume many of the general public would be able to complete. Most decentralised applications also require the download of the application or some of the device's resources to access the network to help run the network. To be appealing to end-users, decentralised applications must be as easy to use as traditional messaging options and also ensure that the user does not have to set up any extra technology or require any advanced technical knowledge. Otherwise, most users would not spend the time or effort to use the application.

1.2 Project Aims and Objectives

This project aims to create a messaging client that stores the messages using a decentralised network. To achieve this aim, a list of objectives were created that would outline the core targets for the project; these are as follows:

1. Design a user interface for users to interact with that is formed from feedback from real-world users
2. Research current solutions and develop an understanding of why users gravitate towards these options
3. Ensure the project is available through a traditional web browser
4. Only store user messages on a decentralised network rather than a traditional server
5. Develop a messaging app that can send and receive messages to different users on the system.

2 Literature Review

The main techniques used in this research will be based around the development and deployment of a decentralised network and then subsequently, the same for an application on the network. In order to understand the requirements for this, relevant research papers and websites will be used to develop the network and application.

2.1 Background

A decentralised network aims to solve three main problems with the current implementation of the internet. When the internet first became a large-scale tool, it was built on the technology at the time. Everyday users did not have powerful computers, and the average storage size of a hard drive in a personal computer in 1991 was around 1 gigabyte. (ThinkComputers, 2013) This meant the internet needed to be spread across servers designed to hold a large amount of data. This began the architecture of our centralised network and paved the way for companies to build large storage systems and own the process we use to store and share data on the internet.

The two main problems that decentralised networks aim to solve are anti-competitiveness and privacy. Currently, the largest companies hosting files on the internet are Amazon, Google, JD, and Facebook. These companies control what data is stored on their infrastructure and how and when users can access that data. This gives these companies a large amount of power and control over a network that was designed to be for the people. This has led to multiple allegations of anti-competitiveness lawsuits. (Cellan-Jones, 2020)

The next issue is privacy, as of 2019 data has overtaken oil as the most valuable asset on the planet. (Bhageshpur, 2019) Companies that host data often use that data to build profiles of users in order to sell advertisements tailored to each user. This is a very lucrative business for these data mining companies; however, it comes at the cost of privacy to users. Websites use cookies (Kristol, 2001) which help track users across different websites in order to build a picture of likes, interests, and hobbies. A decentralised network hopes to remove the power from these companies making it much harder for them to track without explicit consent. The user would own data stored on a decentralised network, and the user would decide what data they wanted to share with advertisers, removing the need for a middle man and putting the power of a users' data back into the hands of the user. (Corbyn, 2018)

Current messaging applications use these centralised systems and often work in conjunction with a social media platform aggregating data. A large amount of messaging apps are available to be analysed if the company hosting the service were inclined. Security and safety of these messages are entirely reliant on the service itself and how secure the information is stored. This provides a potential risk for users using the service. (Rittinghouse & Ransome, 2005)

2.2 Decentralised Networks

2.2.1 Fully Centralised vs Decentralised Servers

A decentralised network differs from a traditional network setup in the way it is organised, and data on the network is stored. A traditional network is built out from a central server; the differences are shown in figure 1. This server is responsible for routing requests and providing access to data. The server is assigned an IP address, and this is how any device on the network can communicate with it. The server then communicates with each other user, relaying information. (Stanford University, 2002) A decentralised network has no central server. The network is setup was as many client/server nodes as possible; each of these nodes operates in conjunction with each other, storing fragments of the overall network data. (Raj, 2019) The user requests data from a manifest of the networks available information and collates the encrypted fragments (shards) and returns this to the user. A decentralised network can also be set up using a blockchain; (Dongdong Yue, 2018)this is a data structure that holds information in a block and links each of these blocks to one another, using a consensus of each user to ensure reliability.

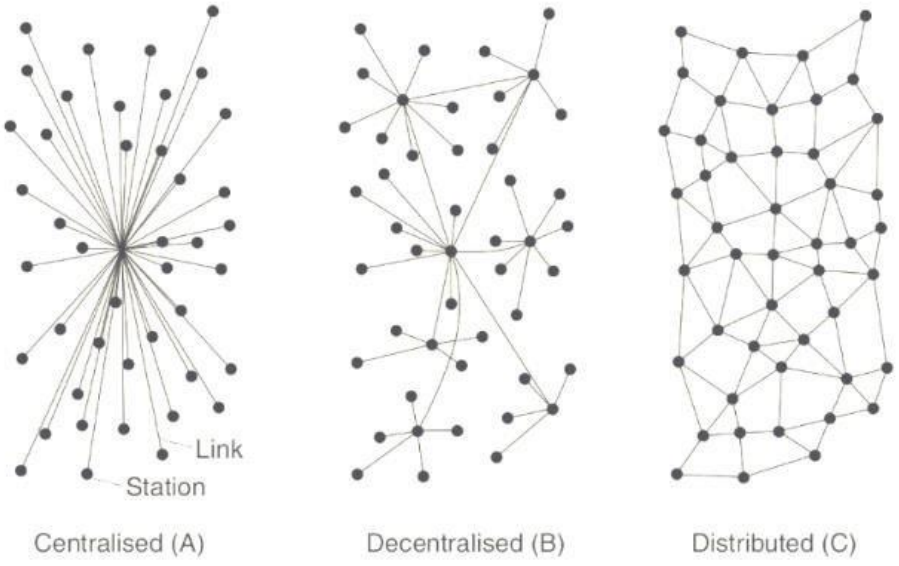


Figure 1: Centralised vs Decentralised vs Distributes Networks (Jayasinghe, 2016)

2.2.2 P2P

A p2p network is a type of network configuration. P2P stands for peer to peer and is a style of a network where, instead of a traditional client-server configuration, each computer on the network is connected. Each computer serves as both a client and a server node. (Kaashoek, 2003)

Each device on the network is equally responsible for storing and providing the data on the network. Reducing the risk of losing connection, as a traditional server-client set up is reliant on the single centralised server being available to connect. If an attack took place on that central server, the entire network would not be able to access that domain. A peer to peer network does not have a single point of failure like a centralised system which reduces the likelihood of a fatal attack. (Cornelli, et al., 2002)

2.2.2.1 Gnutella

Gnutella was designed in the early 2000s by Justin Frankel and Tom Pepper. The goal was to create a scalable p2p network that would allow the transfer of data reliably. Gnutella network worked on a Ping, Pong system. A node would send a ping to the network looking for a host; this took the form of a 'message' if the message found the file it was looking for, that node would reply in the form of a Pong. Each time a file was shared on the network, in order to prevent flooding of bandwidth, each file was given a Time To Live (TTL). This meant that the file would only be transferred as many hops as the TTL. As the network grew, file transfers started occurring over HTTP (Hypertext Transfer Protocol) which allowed for a less fragmented sharing method. (Ripeanu, 2001)

This network, however, started to be used for the illegal sharing of files such as music or movies, as the network did not have a central location, it became the network service for LimeWire, an illegal media sharing service. (Rasti, et al., 2006)

2.2.2.2 Freenet

Freenet is similar to Gnutella in its topology. Freenet has a goal of providing a p2p data distribution service. Alongside the file transfer service, Freenet also has a software suite for publishing on its network. (Freenet, 2020) Freenet allows users to either join the main network by connecting the user node to 'stranger' nodes, or a user can opt to only connect to peers that are on a whitelist. This topology is shown in Figure 2.

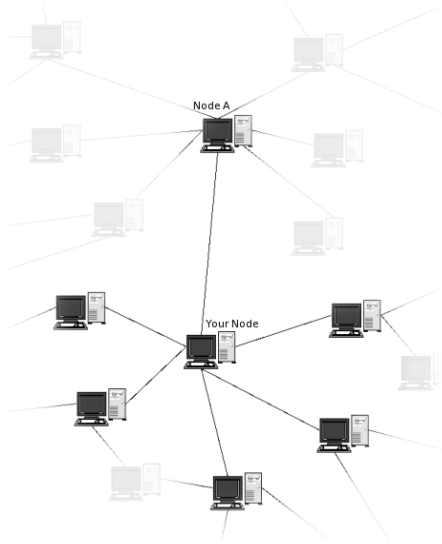


Figure 2: Freenet network topology (Freenet, 2020)

Each time a user stores data on the Freenet network, that data is assigned a key, this allows the user to request that data back from the network at any time as long as an active peer has the file to share back. This differs from the Gnutella framework as the network requires the user to have the key to access the file; there is not a system to 'search' the network for available files.

2.2.3 BlockChain

Blockchain is a core technology that enables a secure decentralised network. Blockchain was invented in 2008 by an unknown developer under the name Satoshi Nakamoto. (Nakamoto,

2009) Blockchain was first used on a large scale in the cryptocurrency bitcoin, shown in Figure 3; starting in 2009, bitcoin used the blockchain concept to ensure a secure, online, and safe means of transferring money globally. (Marr, 2017)

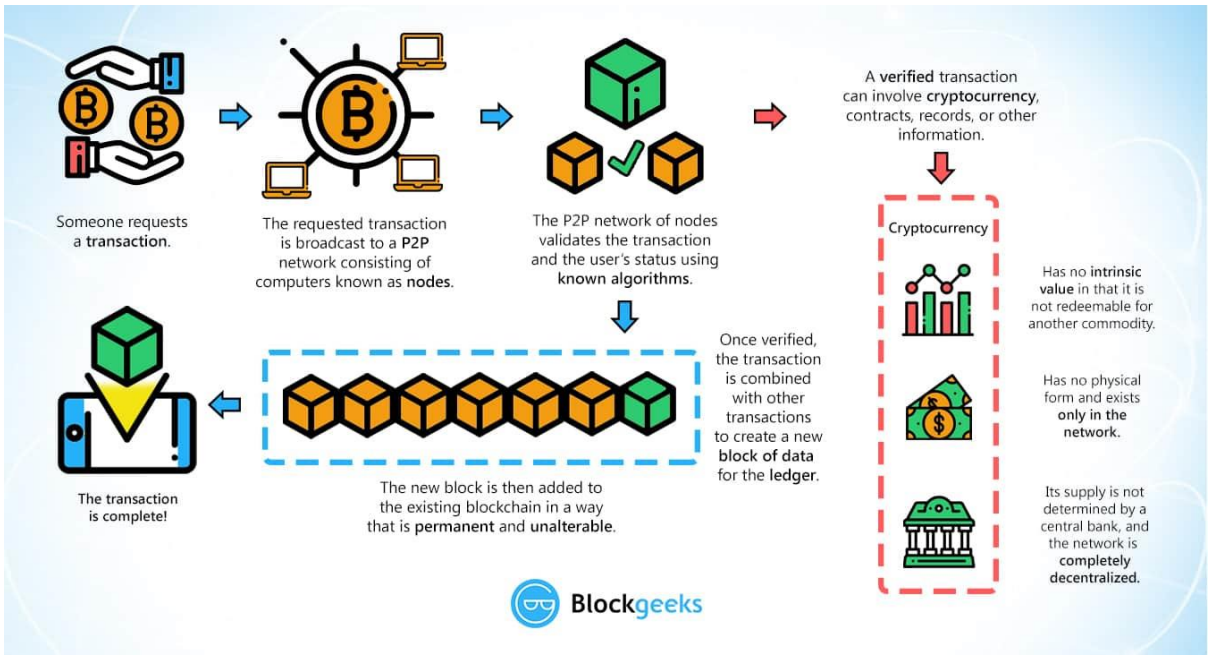


Figure 3: What is Blockchain technology? (Rosic, 2020)

The blockchain is a concept that allows a secure means of transferring data between two people. Built similarly to a Linked List data structure, each block in the blockchain holds at minimum three pieces of data. The data stored inside of the block, this is usually transactional information. Next, the block holds its hash, referred to as a fingerprint of the block. Commonly using the SHA256 method for hashing (Gueron, et al., 2011), this makes the hash very secure, and as of current technology, it is unable to be solved faster than brute force. The next piece of information stored in the block is the hash of the previous block in the chain. If the block is the first block in the chain, this is known as the genesis block. (Sihi, 2020)

Blockchain works on the principle that if a block is modified, the fingerprint would be changed. Thus the chain would be broken and invalidate any transactions on this modification. A vital characteristic of the blockchain is that it works on a consensus principle. Requiring the network to agree on a new block. (Marr, 2017)

2.3 Messaging Clients

Messaging is a core requirement for consumer devices such as mobile phones, tablets, and PCs. Currently, the most popular messaging client in the world is WhatsApp. A study performed in 2020 by Statista reported that there are only 25 counties in the world that WhatsApp is not the market leader. (Statista, 2020) Users were reporting WhatsApp ease of use and ubiquity as the core reason for use. Followed by Facebook messenger as the second-most popular messaging client, closely tied in with Facebook, many users who already had profiles for facebook naturally progressed into using Messenger. (Statista, 2020)

2.3.1 Current Messaging Clients - Centralised

Current messaging clients work on a client>server>client system. Each time the user wishes to send a message, depending on the application used, the message is encrypted. Most well known messaging apps use one of two systems for encryption. (BBC, 2018) Either end to end encryption or user server encryption.

Every message sent, whether encrypted or not, is sent to the central server for the application. Then each of these messages is sent to the recipient from the server. If the application does not have E2E encryption, the application can read messages in transit, whether used for advertising, data mining, or security. The user has no control after the message has been sent. In most cases, if a user stores messages or content on a server, that service is now free to store, copy and share that data. In some cases, the service provider owns the data stored. (Berry, 2019)

2.3.2 Current Messaging Clients – Decentralised

There are currently many messaging clients making use of a decentralised architecture. One of which is Element (Loynes, 2020), element runs on the matrix network, which has over twenty million users worldwide. Utilising end-to-end encryption on all messages, the network allows users to send secure communications. The client is used by governments of France, Germany, and the United States of America. (Matrix, 2020)

Decentralised messaging clients have varying usability, some which rely on solely distributed client/server nodes, are not available at all times or are slow in messaging transactions.

A large benefit of a decentralised app is security. Most decentralised messaging solutions state security as a key feature of the service. If the service uses encrypted sharding or a blockchain, it is much harder for an attack on the service. The lack of a single point of failure removes the possibility of a traditional attack that would leak a large amount of data.

2.4 Incentives for Daily Active User Growth

Decentralised networks have an inherent issue of requiring daily active users to function. While traditional networks only require daily users for growth, a decentralised network requires users to use the network in order to grow storage capacity and data redundancy.

2.4.1 Gamification

A common way to ensure daily active users is to gamify the application. Gamification is generally defined as a means to implement game elements in a non-game context. (Bagnoli, et al., 2016) Applications such as Snapchat, a picture sharing chat app, use a ‘streak’ system to incentivise users to log in each day and use the application. Alongside this, mini-games to play with friends inside of the game encourage messaging more frequently. Applications such as DuoLingo, a language learning application, use goals and mini-games alongside the streak system. (Deterding, et al., 2011)

Gamification can help increase not only adoption of the application, but continual use. In cases of a decentralised network, daily active users is a large metric as of which the success and viability are measured. (Geared App, 2020)

2.4.2 Cryptocurrency

Incentivisation for the continual use of a decentralised service is necessary as if the use of the service declines, the service becomes more vulnerable to attack and will decrease in usability.

Bitcoin in 2009 implemented an incentive to each 'miner' building the blockchain of a return of bitcoin (Nakamoto, 2009). This solution helped both grow the platform and ensure that miners would continue to provide compute resources. This method also theoretically reduced incentive to attack as the resources required to attack could be used to gain coin legitimately.

Cryptocurrency mining can be implemented seamlessly with a blockchain system. Nakamoto used the technology first in Bitcoin and as such has already established itself as a secure system for monetary transactions. With the estimated total bitcoin mined at 18.5 million and each bitcoin at its highest value worth $19,783. The estimated total value of all bitcoin would be $365,985,500,000. (Edwards, 2020)

2.5 Security

Messaging clients arise a privacy concern. Some current centralised networks such as WhatsApp use end to end encryption, meaning each message is encrypted at the client end and not decrypted until it has reached the recipient. (WhatsApp, 2020) End to End Encryption uses a public and private key system in order to ensure there are no points of weakness in the transaction. Other messaging services such as Facebook Messenger do not use this standard; instead, messages are encrypted between the client and server. However, the message is decrypted and then re-encrypted to be sent to the recipient. Facebook would then have access to all messages sent on the platform. A decentralised network would take advantage of end to end encryption on all messages. (BBC, 2018)

2.5.1 E2E encryption

Encryption is essential in the 21st century. As the system stopping malicious hackers from accessing messaging data, bank data, and personal information. Encryption is required in almost every application to some extent, as data transfer should be at least encrypted between the client and server. (Dent, 2008)

2.5.1.1 Signal Protocol

The Signal Protocol is utilised in many end to end encrypted applications currently such as WhatsApp (WhatsApp, 2020) and within the Secret Conversation feature of Facebook Messenger. Signal works on a public key and private key system, making many single-use public keys and making them available across the network. When a message is sent to the recipient, the sender requests a pre-key bundle. This includes the recipient's identity key, signed pre-key, and one of the single-use public keys. This system also deletes single-use session keys to prevent attackers from replaying messages using a repeat session key. (Ermoshina & Francescaand , 2016)

The signal protocol ensures that both the sender and receiver are the only users able to read the contents of a message, the server itself only ever sees an encrypted message. This prevents any third party from being able to intercept messages, including the service provider such as WhatsApp and governments.

2.5.2 51% Attack

Within a blockchain system, the validation for authentic new blocks has to be approved by >50% of the network. In order to prevent malicious blocks being added, compute power is

required to add a new block, generally solving a mathematical equation. Cryptocurrencies such as Bitcoin use a hash system that requires a specific return in order to be approved. For bitcoin, this equation takes approximately 10 minutes.

If a single entity had over 50% of the computer power on the network, that entity could approve the addition of new blocks and as such compromise the integrity of the network as a whole. (Binance Academy, 2020)

2.5.3 Single Point of Failure

A decentralised network removed a massive security vulnerability in current centralised systems. A centralised system has a single point of failure, the central server. This means that if an attacker were able to bring down this server, the website or service would no longer be accessible. In order to mitigate this, a decentralised network does not have this single point of failure; instead, all data stored on the network has redundancy across every/ most node. A traditional server is susceptible to multiple attacks such as Denial of Service attacks (DOS) or Distributes DOS attacks (DDOS). (Lohachab & Karambir, 2019)Attacks like these can bring down large scale networks and cause massive disruption to users.

2.5.4 Decentralised Messaging Drawbacks

One drawback of a decentralised network is as there is no central server monitoring communication and file transfer, it is harder to detect malicious files such as viruses and malware. (Rasti, et al., 2003) This could be an issue; however, the scanning could take place on the end-users device. This would prevent unwary users from opening a malicious file, however, would take more resources from the end-users' devices.

2.6 Development

Decentralised networks and messaging apps can be created in a variety of ways. Some networks have created their own language within their network for the creation of apps or websites. Others utilise more traditional languages such as C++ and Java. The creation of the network architecture is commonly on a JavaScript.

2.6.1 Languages

2.6.1.1 Express js and Node js

Express js is a framework for Node.js. This framework is commonly used for network creation and more specifically, decentralised networks. (Tilkov & Vinoski, 2010) Node is a server-side javascript environment that brings the benefits from client-side js to server. As nodes in a decentralised network can work as both client and server, the seamless integration over javascript allows the network to optimise efficiency.

2.6.1.2 Solidity

Solidity is a recent language to have emerged in the decentralised networking space. Created in 2014 for implementing smart contracts for Etherium, Solidity takes inspiration from object-oriented languages such as C++, while adding functionality similar to Python and JavaScript. (Solidity, 2020)

2.6.2 Frameworks and Networks

2.6.2.1 Pastry

Anthony Rowstron and Peter Drushel designed pastry in 2001. Pastry is a distributed object location and routing substrate for wide-area peer to peer applications (Rowstron & Druschel, 2001). Pastry can be used for multiple distributed applications such as file sharing and

communication. In this system, each node categorised as a leaf node knows its successors and predecessors. Pastry works on the principle that each node is self-organised, as this is a distributed network; there is no central governance to organise the nodes.

Pastry has been applied to multiple applications, including a large-scale event notification system called Scribe. (Rowstron, et al., 2001) Scribe was created alongside pastry to utilise the distributed architecture to notify users of events that matched their interests in specific areas of interest—working independently from the event publishers.

Pastry has an Application Programming Interface (API) that allows for seamless implementation of its structure into third-party applications. This includes operations such as routing, delivering, and forwarding messaged related to a key. Also creating new leafNode set on the fly. Scribe uses this API alongside other first-party applications such as PAST. Third-party utilisations of this include a distributed network messaging application created in conjunction with the University of Sweden. (Lundgren, et al., 2003)

2.6.2.2 Matrix

Matrix is a non-profit UK community Interest Company. The main goal of Matrix is to provide a decentralised real-time communication framework over IP. The Matrix open standard is available to power a wide variety of applications. With a large user base of over 10 million global accounts, Matrix also is responsible for communicating 2.5 million messages per day. (Matrix, 2020)

Matrix provides an API that allows for synchronising extensible JSON objects referred to as “events”. Matrix provides a federated real-time chat service. (Wills, 2015)By transmitting messages to a ‘Virtual Room’ Matrix allows for secure communication between users, with end-to-end encryption being utilised in first-party messaging clients such as Element. Users can create a HomeServer to help host the Matrix network, this aids in the goal to be completely decentralised with as many individual server nodes as possible, reducing likelihood of a loss of service across the network.

Matrix has available clients for a range of equipment and use cases, with messaging applications for iOS, Android, Windows, macOS, Terminal, Web Bases, and Nintendo DS. With each of these clients connecting to the matrix network, the issue of onboarding new users is mitigated by the ease and options available to connect to the network. (Matrix, 2020)

However, although Matrix is structured as a decentralised network, it still relies on traditional servers. This is because the demand for data storage space outweighs the available resources on home servers created by users. Each user has the option to build their server to help aid the network, or just use the network as a client only. Leading to traditional servers being required to help host data. (Matrix, 2020)

2.6.2.3 MaidSafe

MaidSafe is a company founded in 2006 by David Irvine. MaidSafe created the ‘Safe Network’ that is an autonomous, decentralised, and encrypted network. The safe network does not use a blockchain framework as previous network examples. (MaidSafe, 2020) Instead, the data is sharded across the server nodes and is encrypted from the point of upload to the network. While the data is stored on the server, the encrypted shards are continually moved around, making it harder for attackers to intercept data. (Roy, 2008)

With a focus on safety, the Safe Network is not as developed as other networks in terms of compatibility and availability. A large focus of the network is file storage; each user can store on the network after paying with the networks proprietary cryptocurrency SafeCoin. SafeCoin also does not use a blockchain, instead using a close group consensus, only the current and previous owner of each coin are known to each other. This allows for private transaction and anonymous transactions.

MaidSafe does provide an API for this network, using an authenticator, seen in Figure 4, to authorise any modifications to the network, the network allows for developers to use the authenticator as a dependency in their application directly. (Viganotti, 2020)

The SAFE Network is primarily based around a safe and secure network. However, due to the anonymity of the network architecture, many believe that the SAFE network may be used nefariously. (Kantaria, 2019)

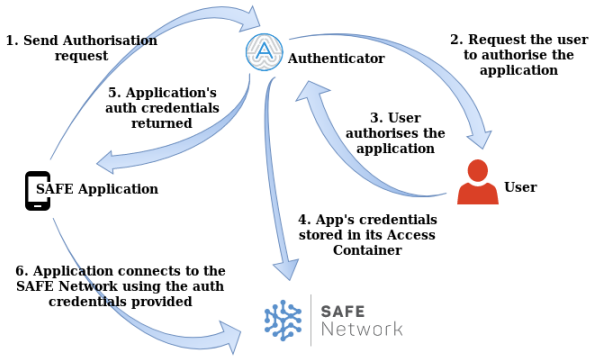


Figure 4: MaidSafe Authenticator Process (MaidSafe, 2020)

2.7 Interface Design and Usability

Adoption if any software is heavily reliant on a users reception of the design. While software developers can focus on functionality and efficiency, it is a combination of all of these principles that will ensure a user accepts the application. Usability is a critical factor in this as if a user cannot immediately figure out how to do their desired task. They are likely to look for another solution. (Nielson, 2012)

Some of the key usability requirements are consistent design, memorability, learnability, efficiency, and satisfaction. In conjunction, if these elements are well implemented, a user is likely to not only use the software but return when requiring again. This also aids in the word of mouth marketing. (Meiners, et al., 2010) Word of Mouth Marketing (WOMM) is one of the most effective marketing strategies in the 21st century; consumers are more likely to trust friends, family, and colleagues over marketing on traditional media.

2.7.1 Interface Design

The visual design of any user application is essential. Layout and colour are two key factors of the impact of the software package used by consumers. (Locke, 2020)

Successful platforms have a consistent and recognisable palette across any elements of the program. Twitter, for instance, has a particular shade of blue that aids in its recognisability, as seen in Figure 5. This aids in user awareness of the platform and ensures recognisability across other platforms. (Yu, et al., 2020)

Figure 5: Twitter Colour Palette (Locke, 2020)

2.7.2 Complexity vs Usability

When the internet first became public, users would have to manually connect to the IP address of the server they wanted to use. This limited the use of the internet to hobbyists and academics. When the internet became more widespread, a means to simply connect to websites using text was required. This became what is known as the Domain Name System (DNS) (Cloudflare, 2020). The DNS server held an address book of each domain name such as www.google.com and its IP address. This meant that users of the internet who were not technically experienced did not have to understand that the DNS server was there, but could use it seamlessly. This is an excellent example of taking complex technology and distilling it for the everyday user.

For a technical product such as a decentralised network to become widespread, it not only has to appeal to technical users. The network also has to appeal to the everyday consumer; this involves stripping back technical jargon and options and allowing for a simple application everyone can use. As seen in Figure 6, a study in 2016 showed that in the UK, around 70% of the public between the ages of 16 and 65 have low technical literacy. (Nielson, 2016)

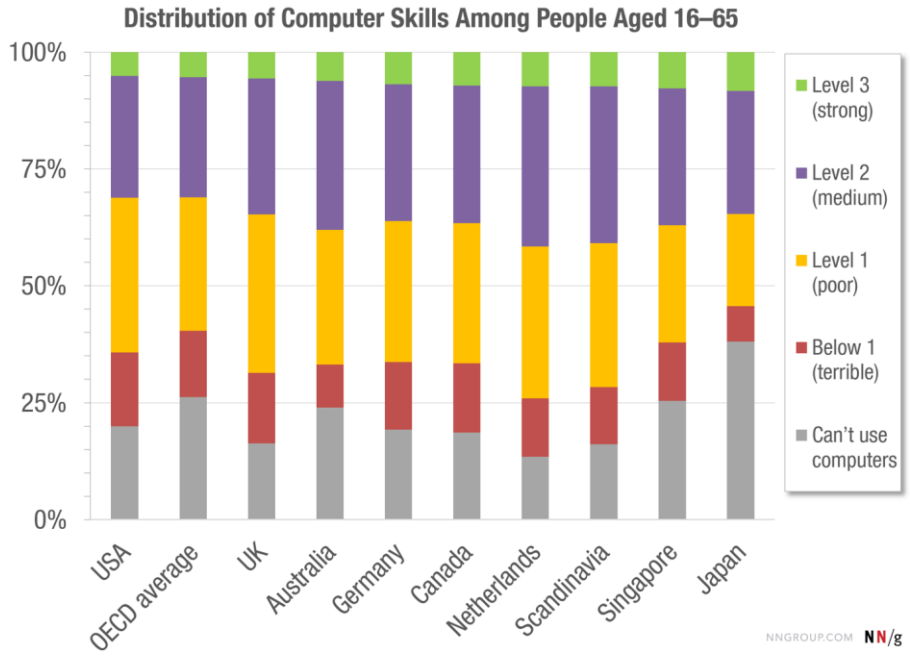


Figure 6: Distribution of Computer Skills Among People Aged 16-65 (Nielson, 2016)

The purpose of this graph is to display the percentage of internet users that understand or care about 'how' it works. This has to directly affect the creation of any new networks as mass adoption cannot take place until the system is easy to use and reliable until then it will be for hobbyists and academics.

2.7.3 User Testing

User testing is a core aspect of any development process; it ensures that users not familiar with the inner workings of a system are also able to use the application. A key element of user testing is Cognitive Walkthrough. Cognitive Walkthrough is a theory that looks to evaluate user experience based on their ability to navigate the interface with no prior knowledge. This tests the learnability of the application by requesting users to complete specific tasks using the application. (Blackmon, et al., 2002)

2.8 Conclusion

From the research, decentralised networks can be created in a variety of ways. Utilising blockchain for a secure and decentralised network provides a variety of benefits against a traditional centralised system. Networks such as Matrix have a large user base and developing for that framework would only help to grow an already successful solution. However, although Matrix is growing, future networks need to be adopted by mass markets and provide a simple, worthwhile, safe option for users. Reviewing the literature relevant to security shows that decentralised systems inherently have benefits against a traditional system, removing a middle man that can analyse and sell data sent across its network. Using frameworks such as pastry alongside signal for encryption would provide a new, secure system that would allow users to own the data they add to the network. Something that is not currently possible unless a user hosts a personal server. The interface design and usability literature shows that as important as a working service is, a well designed and intuitive service will be required for mass adoption.

Design / Methodology

3.1 Requirements Analysis

The aim of this project is to create a messaging client that stores the messaging data on a decentralised network, not relying on traditional centralised servers for this. The project will use either a web-based client or application in order to access the network. The application will quickly send messages across the decentralised network and ensure that the conversation feels natural to users. The application should, to the end-user functionally be indistinguishable from traditional messaging clients, ensuring that using the system feels as natural as not to deter users from the new back-end.

The requirements in this section will follow the MoSCoW method. This will be split into sections of requirements that the project "Must" have, "Should" have and "Could" have. The section will also be split into functional and non-functional requirements.

Functional Requirements

- Must make use of a decentralised network.
    - Should be using centralised servers to a minimum, ensuring that all media, messages, and content are stored on decentralised nodes rather than centralised server databases.
- Must compress messages before sending.
    - Should use a standard such as Huffman Encoding to compress messages, reducing stress on the network.
- Should allow users to send multiple media types.
    - The application should allow for emojis to be sent and received.
    - The application could allow users to send some image formats such as jpg.
- Should be as simple to set up for a new user as a traditional messaging client.
    - Should have a login or signup page with clear ability to use. The application must be usable by users with both a technical background and a non-technical background.
- Should have a responsive and intuitive interface
    - The interface should not be slow or unresponsive to user input. Responding to input in a reasonable and expected time.
    - The interface could notify the user if a specific action is holding the application; this could be achieved with a notification or loading animation.
- Could include video and audio call functionality over the same network.
    - Could implement a real-time video call for 1:1 or group calls. This could be implemented as a seamless part of the chatroom and not a separate section.
- Could include end-to-end encryption on all messages, resulting in a safer and more secure messaging client.
    - Could implement an End to End encryption standard building on the security principles shown in the Literature review.
- Could include group-messaging options with chat rooms for friend or family groups.
    - Could include group calls alongside this functionality.

Non-Functional Requirements

- Must have a client that the user can interact with.
    - Must be an application or web interface that allows users to connect via multiple devices
    - Should work on devices of different sizes, scaling display and options to the devices.
- Must process messages within a reasonable time
    - Messages must be sent and received within up to 4 seconds in order to ensure conversations are coherent for all users
    - Should have timestamps for messages on both sender and receiver side.
- Must be accessible at any time for users to either send or receive messages.
    - Should have an uptime of >95%. This will likely be achieved if a service such as AWS or DigitalOcean are used to host.
- Must allow multiple users to access simultaneously and send messages to any other participant of the service.
    - Must be able to scale to many users at once.
    - As hosting is likely to occur on a free or low-cost subscription, user limitations should be bound by these subscriptions rather than the software limitations.
- Must adhere to data protection laws and ensure that no user data is stored other than a user-chosen password and a randomly generated username.
    - Must ensure no other information is stored about the user in their profile.
    - Could implement a captcha style verification system to ensure the application is not overloaded with requests.
- Must be completed by hand in date of 9th April to show demonstration and poster.
- Should have appropriate documentation guiding users to utilise the application.
    - Should have documentation available in pdf format
    - Could have a tutorial inside of the app showing users how to utilise the application.
- Should have a consistent user interface across the entire application
    - Should be consistent in theme, layout, images. Ensuring a seamless experience for the user.
    - Should follow standard design principles for colour schemes and layout.
- Should be accessible to users with some disabilities that could affect the use of the application.
    - Should have alt text on buttons and graphics for screen readers.
    - Should follow guidelines for accessibility features of applications.
- Should use Node.js for development and build of back end web interface.
    - Could run back-end on a Linux Virtual Machine such as Debian or Ubuntu
- Should be accessible from connecting to a regular internet browser
    - Should have a bridge from decentralised network to the internet to allow access from traditional internet to the new network.
- Could use a pre-defined network such as Matrix or Etherium to host application.
    - Could build using Solidity, built for the Ethereum network, this language provides necessary tools to create the application.
    - Could use Matrix API to build the application for their network

- Could be built on a custom network, hosted by either home server devices or sandbox Virtual Machines on AWS or similar.
    - Could be a custom network working independently from an existing system.

If the requirements outlined in this section are met, the project should be usable as a messaging client. This could then be passed on to a development team to build out feature sets and security. The application would work as a foundation for further development.

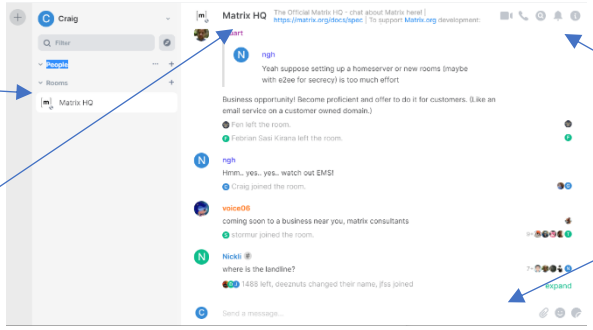
3.2 Project Design

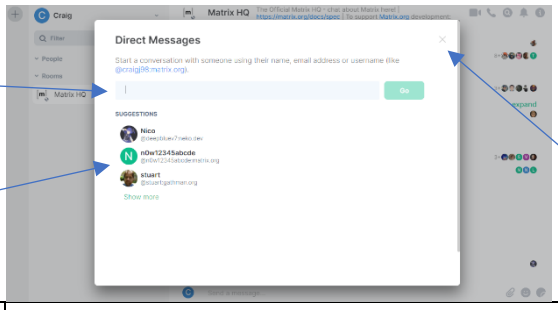
As one of the project goals is for the application to be usable by both technical and non-technical users, the design was a significant factor in the development process. For this, logo designs, colour palette choices and layout styles are to be researched. Messaging applications currently available for desktop, traditional centralised applications, and decentralised applications were to be investigated to understand how these applications approached user design. The user experience, from design consistency to features, would be analysed in this research. Then, wireframes and mockups were to be created to portray the information gained in the research. After these were created, users would be polled on their option of the designs. From this data, the final design would be created.

3.2.1 Market Research on Current Messaging Apps for Desktop

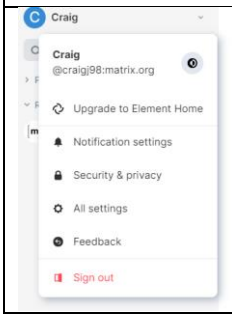
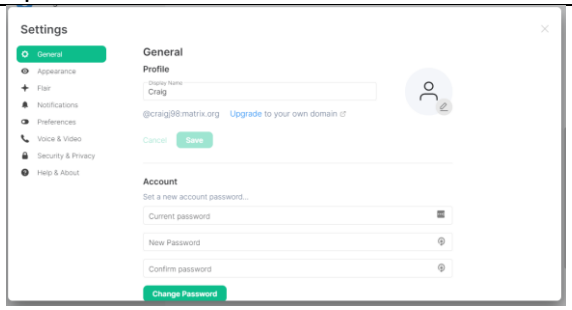
Decentralised messaging apps are currently growing in popularity among tech enthusiasts. The first application to be researched is this space's leading application based on daily active users and monthly growth, “Matrix.org”, with their chat app “Element”. Beyond decentralised applications, “WhatsApp” is one of the most significant messaging applications globally; their Desktop application will be researched alongside Facebook Messenger. In the research, the leading messaging page and the applications page provides the ability to create new messages, either 1:1 or group chats.

3.2.1.1 Element

Page	Main Messaging page (1)
Description	This page allows users to send messages to the selected person or group. The user can also add users and navigate to the profile page
Screenshot (annotated)	
List of friends and joined rooms, alongside selected Information about group or friend selected  	
Background	Light greys/whites, bright colours used to represent users.
Interface Components	Text Area to send messages Text area filter option to search for contacts
Accessible Sub Pages	Profile Menu Dropdown (Sub Page 1), Settings Page (Sub Page 2)

Page	Add friend page (2)
Description	This page allows users to add friends to send messages to within the website
Screenshot (annotated)	
Search bar to search for unique username List of suggested friends to add  	
Background	Light greys/whites, bright colours used to represent users.
Interface Components	Text Area to search for friends Button to perform search Exit button to return to previous page

The following sub-pages are accessible from the application

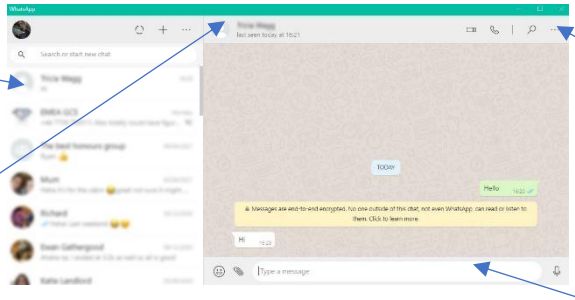
Sub Page 1	Sub Page 2
Profile Dropdown, various options for account	Settings page, various user preference options
	

3.2.1.2 WhatsApp

Page	Main Messaging page (1)
Description	This page allows users to send messages to the selected person or group. The user can also change which chat is selected or call the current chat selected.
Screenshot (annotated)	

List of friends and joined rooms, alongside selected room

Information about group or friend selected



Options to call and set alert to selected group, also gain more information on group

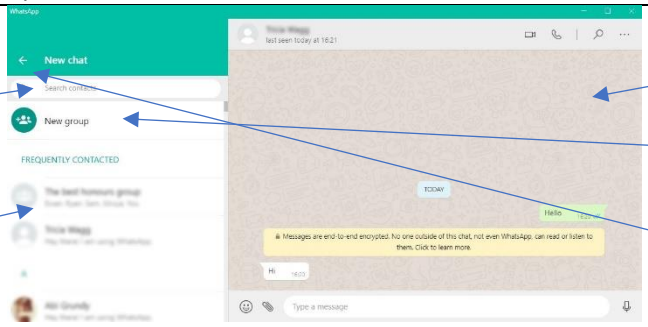
Send Message text are and button to send to current group

Background	Light colours, WhatsApp green accents and patterned background.
Interface Components	Text Area to send messages, Text area filter option to search for contacts Buttons to view user dropdown
Accessible Sub Pages	Profile Menu Dropdown (Sub Page 1), Settings Side Menu (Sub Page 2)

Page	Add friend page (2)
Description	This page allows users to add friends to send messages to within the website
Screenshot (annotated)	

Search bar to search for unique username

List of suggested friends to add



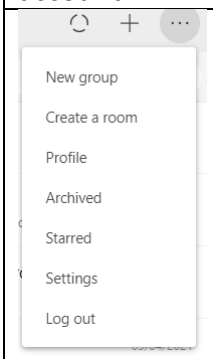
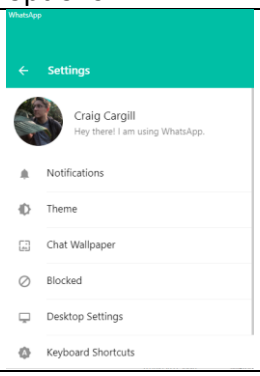
Regulat chat window still visible

Create new group button

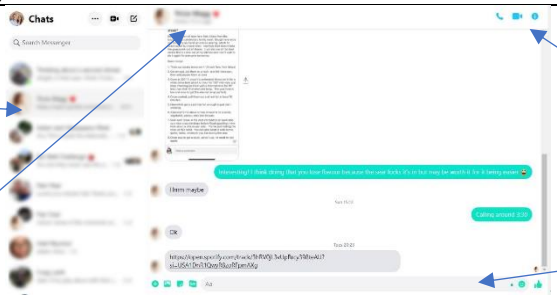
Close popup and return to main app

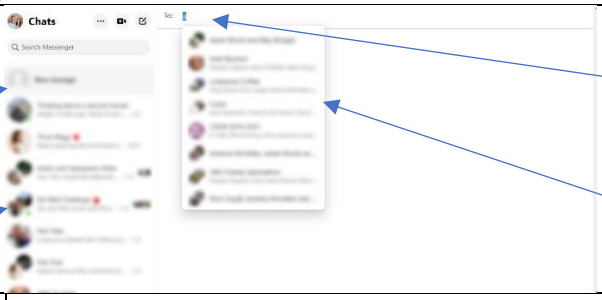
Background	Light colours, whatsapp green accents and patterned background.
Interface Components	Text Area to search for friends, Button to perform search Back button to return to previous page

The following sub-pages are accessible from the application

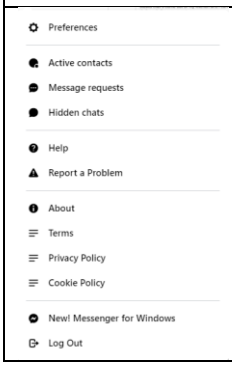
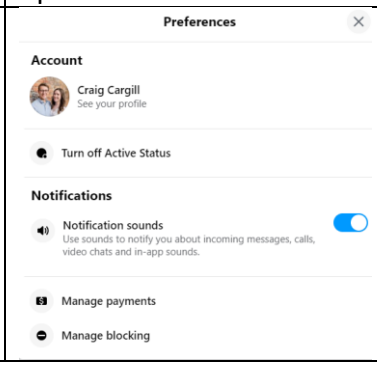
Sub Page 1	Sub Page 2
Profile Dropdown, various options for account	Settings page, various user preference options
	

3.2.1.3 Facebook Messenger

Page	Main Messaging page (1)
Description	This page allows users to send messages to the selected person or group. The user can also change which chat is selected or call the current chat selected.
Screenshot (annotated)	
	
Background	Light Colours, chat specific accent added. Default is Facebook Blue
Interface Components	Text Area to send messages, Text area filter option to search for contacts Buttons to view user dropdown
Accessible Sub Pages	Profile Menu Dropdown (Sub Page 1), Settings Page (Sub Page 2)

Page	Add friend page (2)
Description	This page allows users to add friends to send messages to within the website
Screenshot (annotated)	
	
Background	White background, chat specific accent added. Default is Facebook Blue
Interface Components	Text Area to search for friends, recent chat previews to navigate to those chats instead of creating new.

The following sub-pages are accessible from the application

Sub Page 1	Sub Page 2
Profile Dropdown, various options for account	Settings page, various user preference options
	

3.2.2 Analysis of researched competitors

3.2.2.1 Element

Element's design follows on from traditional chat applications. A large window to show the current messages and a left bar that shows the available chats. With complimenting accents of grey, the application takes a modern approach with minimal clutter. The navigation is somewhat complicated with multiple dropdown settings options for either chat settings, listed room settings, or profile settings. However, the coloured names for each user provide an easy way to identify which users are messaging and what they are saying.

Overall, Element provides a modern approach to a messaging app, focusing on functionality and features. However, settings are often hard to find, which causes frustration if a user is attempting to complete a simple task such as view profile information.

3.2.2.2 WhatsApp

WhatsApp design follows suit in a similarly large area for the focused chat. The left-hand pane is also navigation between recent chats. WhatsApp uses the WhatsApp green colour as an accent which helps identify the application as WhatsApp quickly. With a clear focus on chat, WhatsApp provides a simplistic display without clutter. A patterned background provides a clear separation between the chat window and the navigation pane. The patterned background helps as chat settings are easily separated from account settings.

Overall, WhatsApp provides the core functionality of messaging without extra features. This provides a simple design that users without technical knowledge can understand and use. WhatsApp reduces user click rates between interactions by ensuring the chat is always accessible, and that feature panes do not hijack the screen. However, in creating a clean and simple design. Extra functionality is sacrificed.

3.2.2.3 Facebook Messenger

Facebook Messenger uses a large chat pane on the right with message navigation on the left once again. With white panes separated by a thin grey line, the page does not have a precise segmentation of its areas. User messages have the Facebook Blue background colour by default, and received messages are grey. Facebook Messenger does, however, provide densely packed information in the left pane. With each chat showing the users profile, the last message and when that message was sent. If the chat is a group, recipients that have read the message are displayed.

Overall, Facebook Messenger has a hard to separate display. However, it does provide a simple way to navigate and shows users dense information in previews. Alongside this, chat customisation such as custom colours help appeal to its target demographic of non-technical users.

3.2.3 Website Name and Logo Designs

3.2.3.1 Website Name

To start design on the application, a suitable name must be identified, research into available chat applications names and their etymology must be completed to decide which name is most suited for the application. Furthermore, identifying key components in the application to make a name out of these could create a name. There are two components to the application name; the first is the type of application it is. In this case, the application is a chat app. Therefore, the word "chat" and synonyms of this word may be incorporated into the

name. The other part of the application is the decentralised network. As this is a critical marketing point of the application, this may be a factor in an appropriate name. Alternatively, names such as “google”, “apple”, “amazon” are successful names that are unrelated to their core value proposition. For this reason, unrelated words were also researched as potential names. To find an appropriate name, a Mind Map was produced; this is shown in figure 7.

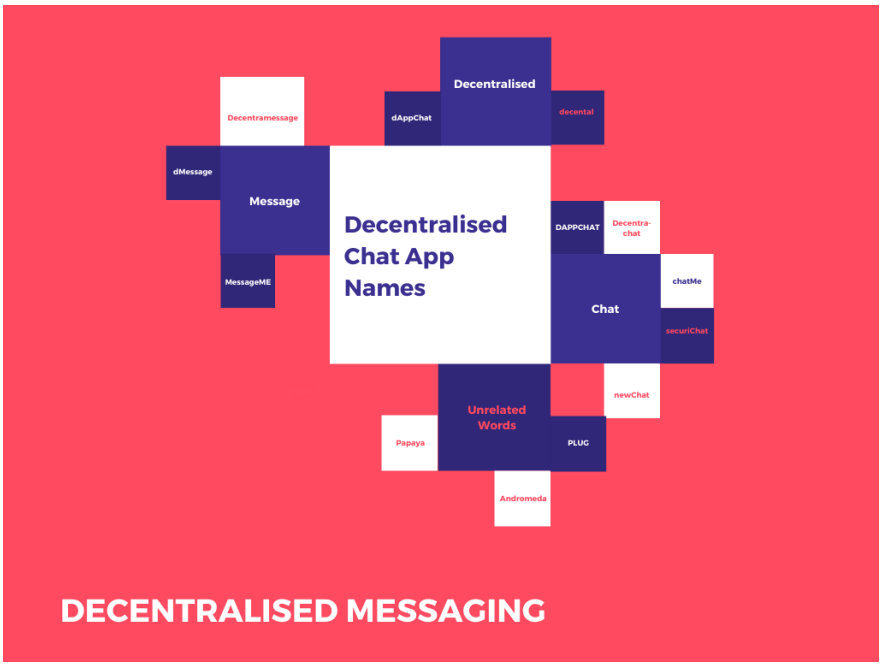


Figure 7: MindMap for Website Names

Once the mindmap was produced, users were polled with these name options in a survey. The users were given the option of all of the names created in the mindmap stage. The names were in no particular order as to not bias the results. The results of this survey are shown in figure 8. Once the responses had been gathered, the most popular name was selected to be “dAppChat”. The second most popular choice was “Decentramessage”. This survey results indicate users prefer relevant wording and clear messaging of what the application is and what the functionality will be.

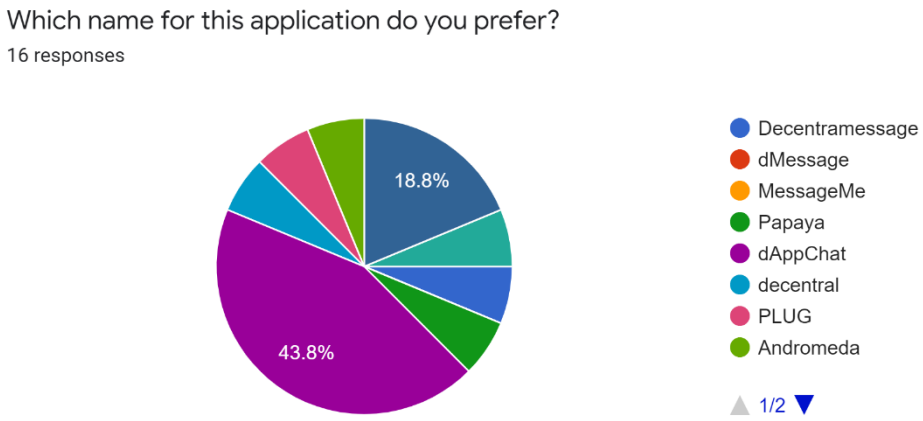


Figure 8: Application name survey results

3.2.3.2 Website Logo

The next stage of the design is to create a Logo for the website. In order to create this logo, similar design stages were taken to creating the name. The first step would be to create mockups of the logo design. From there, the logos would similarly be surveyed by users. Market research shows, most popular messaging apps, both centralised and decentralised, use a chat bubble as a core piece of the logo design. “WhatsApp”, “Facebook Messenger”, and “iMessage” all make use of a chat bubble in their logo; this shows to the user quickly that the website or application is for chat and may help drive users to use the application if they need to send a message. Another core technology in the website is the decentralised factor. Decentralised networks have a different topology to centralised networks; this could also be implemented into the logo's design to differentiate the website from other services. Another consideration is whether or not the website’s name will accompany the website logo. Figures 9-17 show the mockups of the potential logo designs.

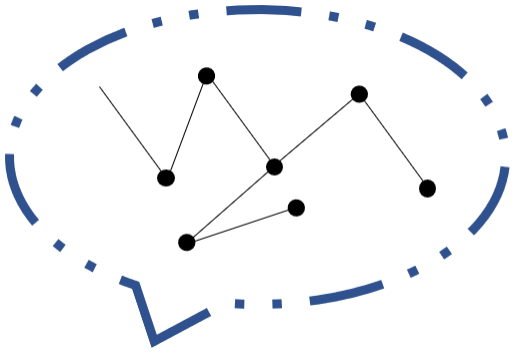


Figure 9: Logo Design 1

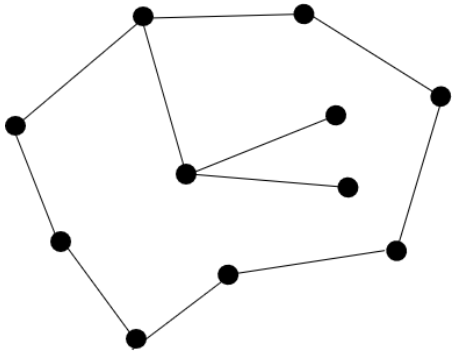


Figure 10: Logo Design 2

Figure 11: Logo Design 3

Figure 12: Logo Design 4

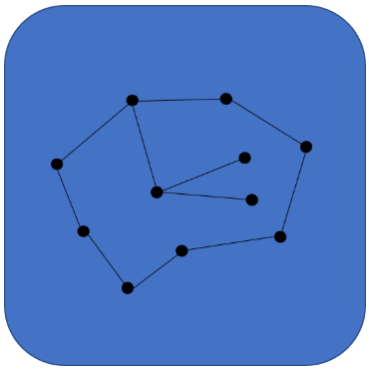


Figure 13: Logo Design 5

Figure 14: Logo Design 6

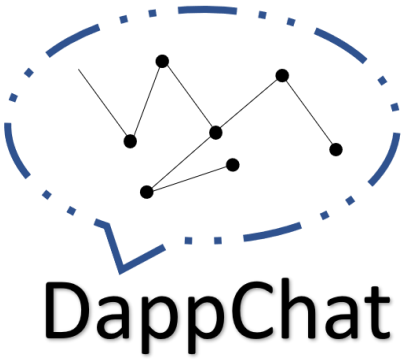


Figure 15: Logo Design 7

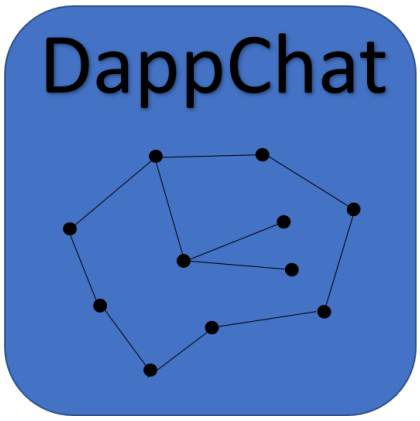


Figure 16: Logo Design 8

Figure 17: Logo Design 9

These mockups were then posed to users in the survey. The users were asked which logo they prefer; the logos were again positioned to the users in no particular order to not bias the results. From the survey results shown in figure 18, "Logo Design 1" gained over 50% of the vote. "Logo Design 2" was second with 12.5% of the vote. These results show a user interest in a transparent background and no text in the logo. Other than "Logo Design 1" and "Logo Design 2", the other designs were either split at 6.3% or had no votes cast for them. With the largest share of 56.3%, "Logo Design 1" will be taken forward and used in the following stages of the design methodology.

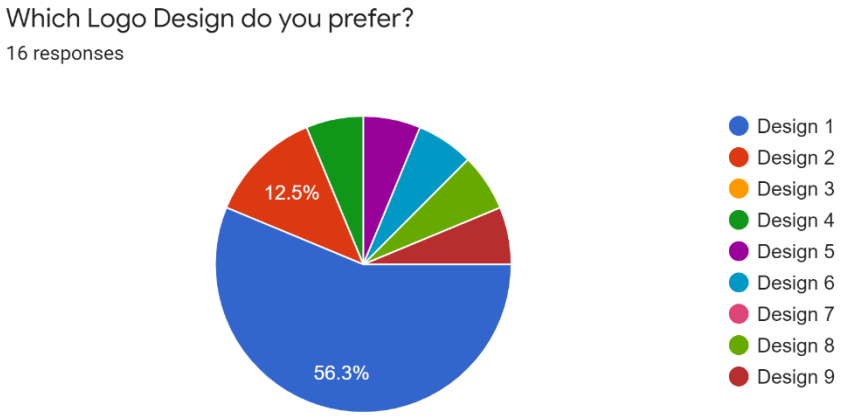


Figure 18: Logo Design Survey Results

3.2.4 Website Page Designs

After researching current messaging websites such as WhatsApp and Facebook Messenger, the core design choices will be the base for dAppChat. Before creating the mockups, wireframes must be designed to create a foundation for the mockup design. The wireframe will function as a template for the mockup design and ensure that design queues from the researched applications are implemented in the desired designs. There are to be three main pages on the website with which users will interact. The first of which is the “Login Page”, this page is which users will be greeted by upon opening the website. This page will then push users to the “Chat Page”, which is the second page that will be wireframed. The final page that a wireframe will be created for is the “Create Room” page. The “Create Group” page will be a sub-page of the main chat page and allow users to create rooms that other users can join.

Login Page Wireframes

The login page is the first page users are greeted with when they attempt to use the website. For this reason, the design language chosen for this should be consistent with the rest of the website. There should not be any obstacles for users to log in to the website, or if they do not have an account, register for one. Figures 19 and 20 show two different wireframe designs for the login page. The wireframes take inspiration from the researched websites; for instance, Facebook Messenger takes an approach of offsetting the login boxes to the side. By splitting the page up, the page feels larger and more professional. Alternatively, many login pages take a traditional approach of a panel in the middle of the display that houses the information. Ensuring users know what they need to do to log in and have easy access to any alternate pages such as the register page. As the register page will house similar information, the design chosen for the login page will be replicated on this page.

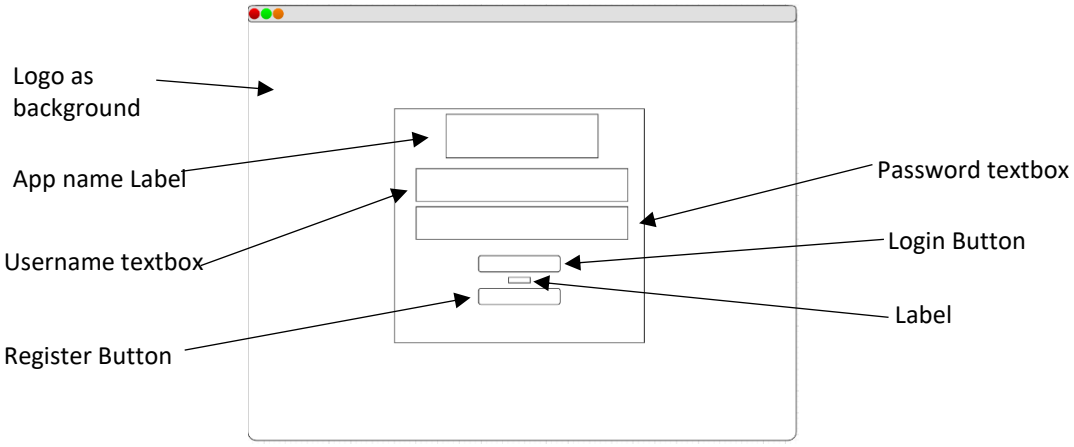


Figure 19: Login Wireframe 1

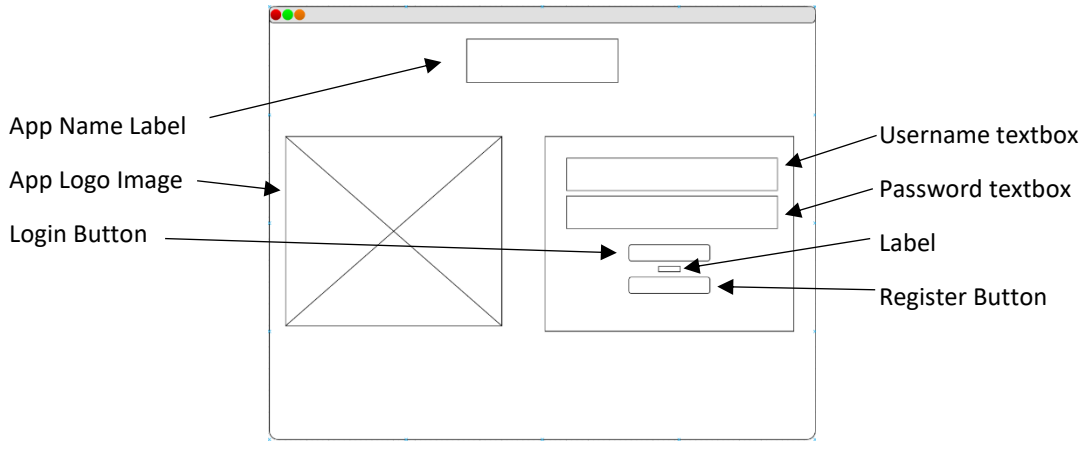


Figure 20: Login Wireframe 2

Chat Page Wireframes

The chat page is the website's main page; this is where users will send and receive messages while using the website. Users will also view the current rooms they are in and navigate to the create or join room page. Most researched websites take a different approach to the messaging architecture than this website will. As researched applications tend to use direct user messaging rather than a more "forum" style approach, "Google Meets", which uses a similar room-based system, was researched. Figures 21 and 22 show two different approaches to the chat page; the researched websites inspire each of these designs.

The first wireframe shown in figure 21 uses a similar approach to figure 19, in the login page. This approach is to have the contents of the page suspended in a pane on the page. Suspending the contents into a pane draws the users focus to the centre of the page. This technique gives a style similar to an application rather than a website. Alternatively, many websites take an approach similar to figure 21. Using this approach involves taking advantage of the entire window of the browser. By doing this, items are given more space and instead of splitting the user's experience into an application inside the browser, provide an experience as if the browser is the application. Both designs contain the same contents such as buttons, text areas, and labels; however, they differ in their approach to convey the options to the user.

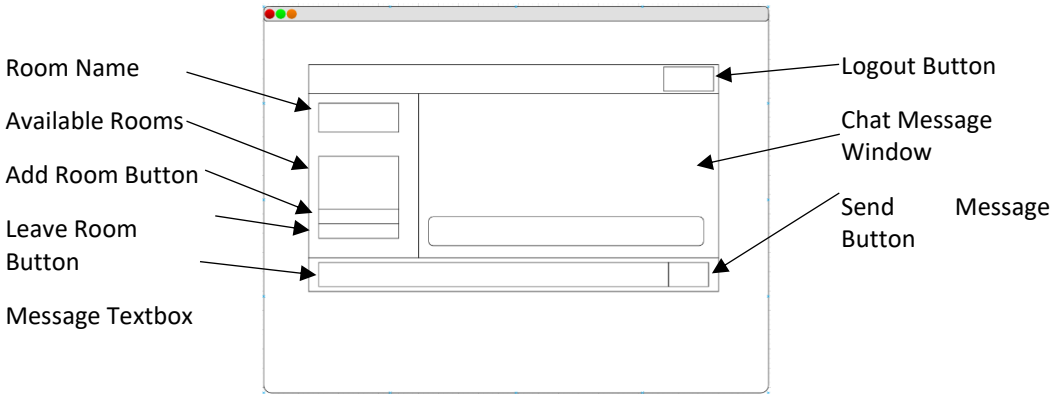


Figure 21: Chat Wireframe 1

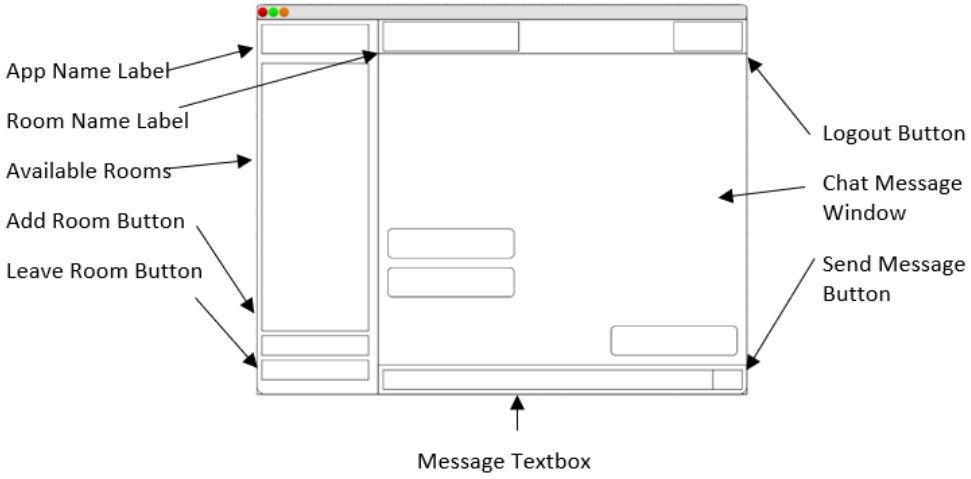


Figure 22: Chat Wireframe 2

Add Room Sub-Menu

While on the "Chat" page, the user has the option to add a room to their list of available rooms. To do this, user can either join an existing room or create their own. Room names are auto-generated; to create a room, a user needs to click 'Create Room'. A unique room code would be generated, which other users would need to join that room. "Google Meets" uses a similar room generation technique.

The wireframe, shown in figure 23, limits items on the page and uses text inside the button and text box to convey meaning to the user. This approach results in a cleaner look, with only essential items to be displayed on the page. This approach, however, does not use labels to convey meaning, which could confuse users. Alternatively, figure 24 shows a different approach using labels to convey meaning for each of the button options. This results in more items on the page however ensured users are informed of the options.

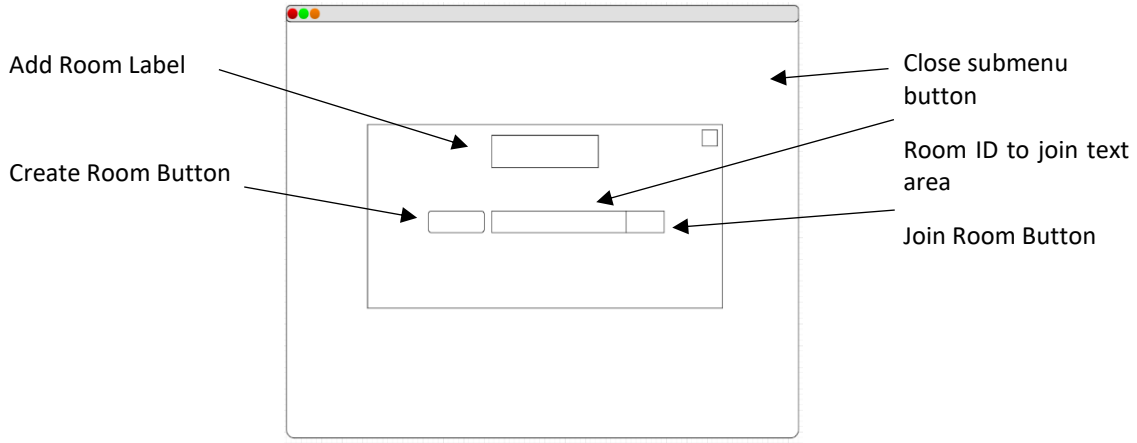


Figure 24: Add Room Page Wireframe 1

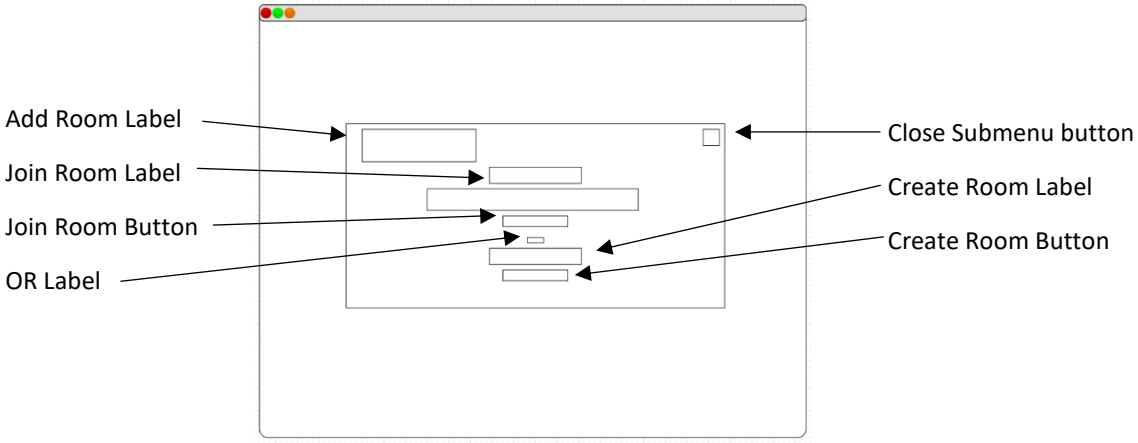
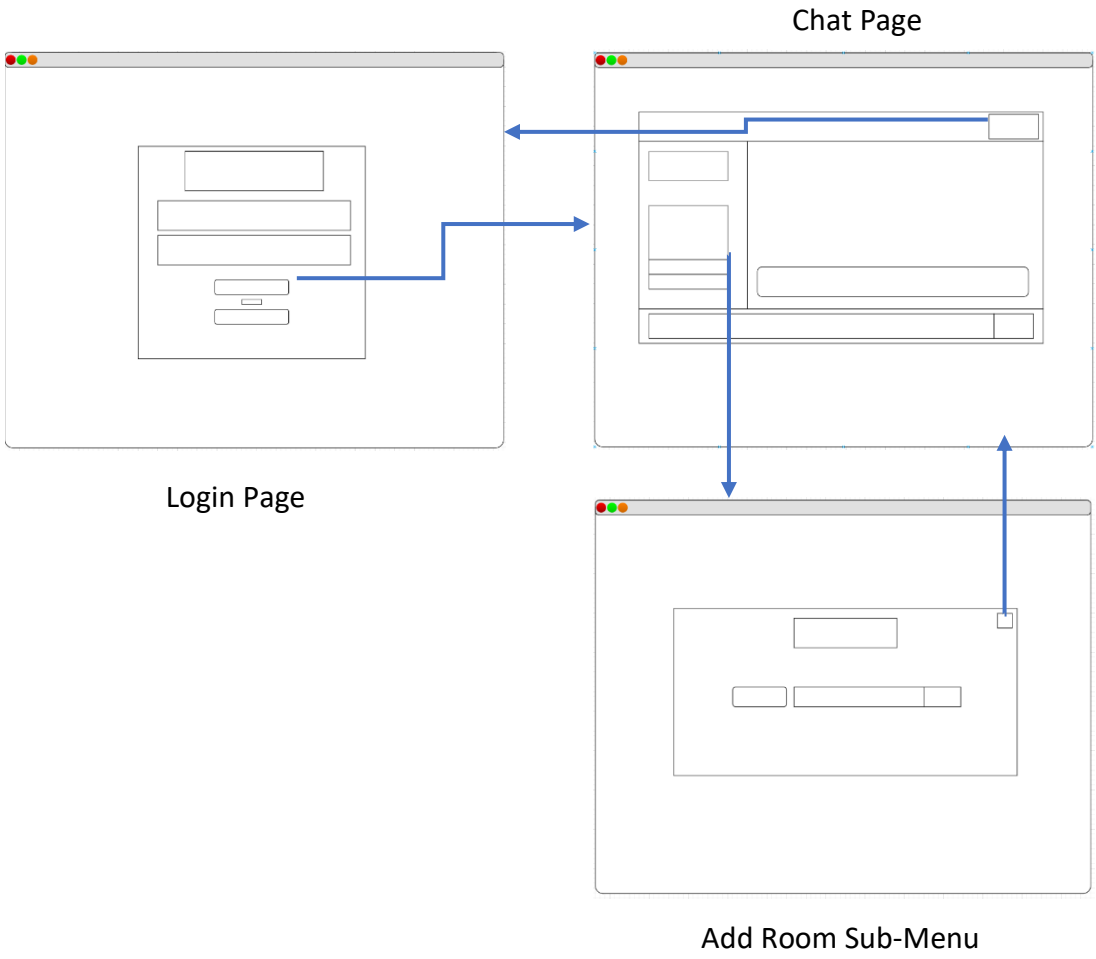


Figure 23: Add Room Page Wireframe 2

Navigation

Buttons interconnect each of these pages, shown in the wireframes on their page. The first wireframe for each of these designs will be used to show this structure to represent this.



Login Page Mock-up

The next step was to create mock-ups of each of the login page design to understand the visual representation of these designs from the wireframes. The mock-ups replicate the wireframes using “MarvelApp”, a software that creates a prototype of an application or website design. Creating the designs by utilising elements from existing researched chat applications and websites such as “Facebook Messenger” and “Element by Matrix” allowed the prototype mock-ups to become accurate representations of what a user may interact. Ensuring accuracy will aid in the results for the user survey on these mock-ups.

The prototype for the Login page in design one, shown in figure 25, can be accessed at: https://marvelapp.com/prototype/c6e3526/screen/78598157

The prototype for the Login page in design two shown in figure 26 can be accessed at: https://marvelapp.com/prototype/c6e3526/screen/78598216

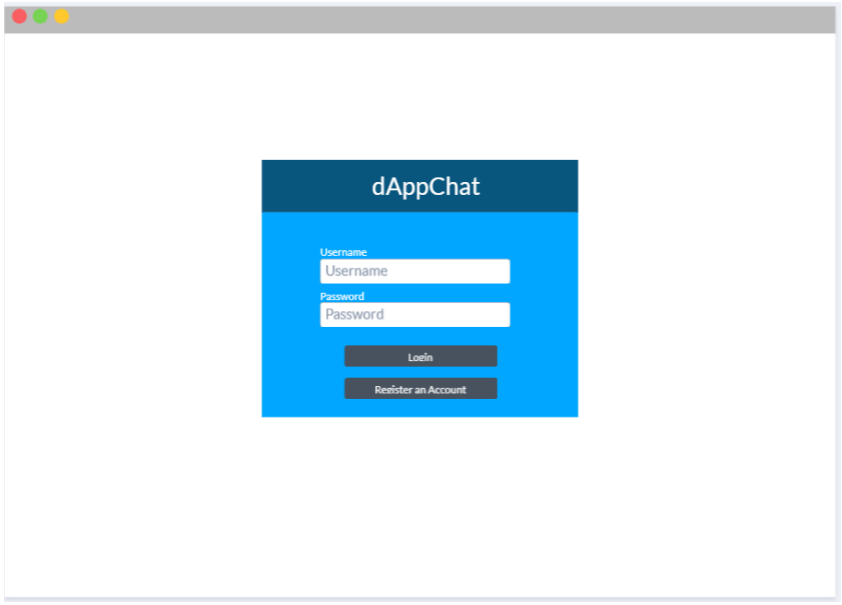


Figure 25: Login Page Mockup 1

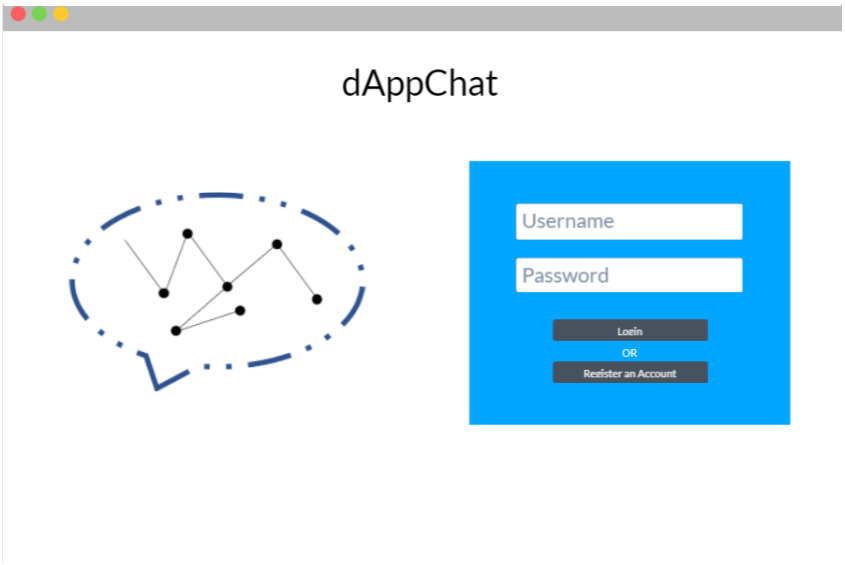


Figure 26: Login Page Mockup 2

Chat Page Mock-up

Similarly to the Login page, mock-ups were created from the chat page wireframes. These mock-ups were again created using "MarvelApp." From the designs laid out in the wireframes, these mock-ups were created alongside inspiration drawn from the researched websites and implemented into the designs.

Figure 27 shows the mock-up implementation of using the floating pane approach. The floating plane approach aims to encompass the application in a segment on the page, allowing the user to feel the application is self-fulfilling rather than part of the browser. The prototype of this can be accessed at https://marvelapp.com/prototype/c6e3526/screen/78598230

Figure 28 shows the mock-up using the alternative approach; this design utilises the entire browser window to display to the user. Websites such as "Facebook Messenger" and "Element by Matrix" use this approach to conveying the design. The prototype of this can be accessed at https://marvelapp.com/prototype/c6e3526/screen/78598323

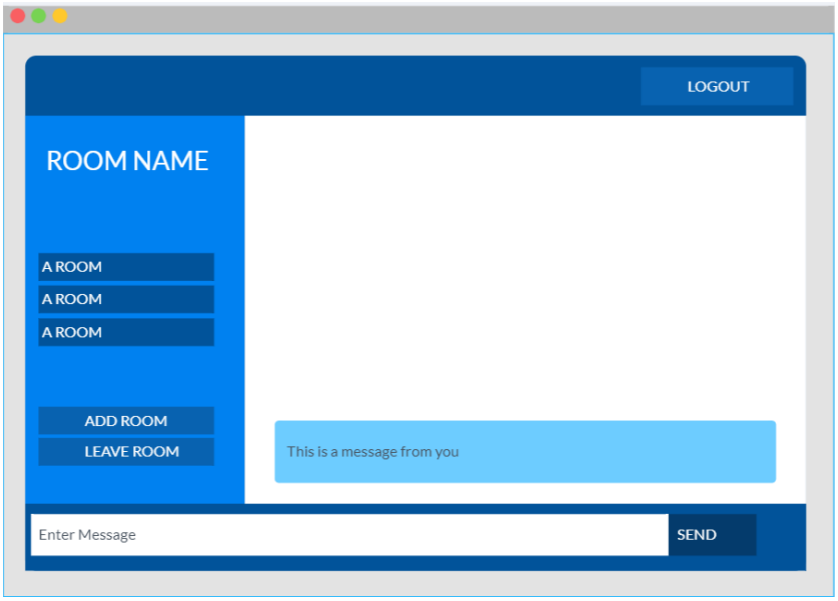


Figure 28: Chat Page Mockup 1

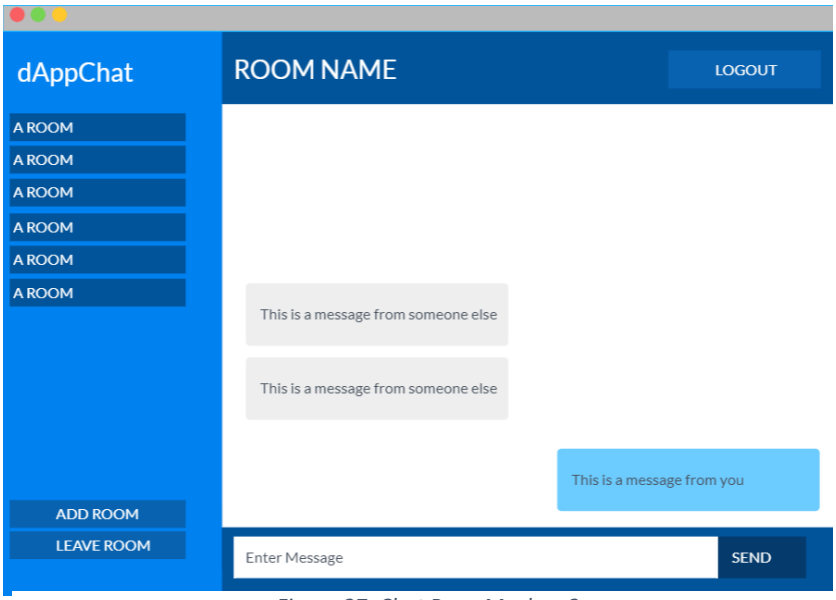


Figure 27: Chat Page Mockup 2

Add Room Sub-Menu Mockup

Finally, the wireframes were again used to create the mockups for the Add room sub-menu. This menu will appear when the user selects “Add Room” from the chat page and will allow users to either join a room or create a new room.

Figure 29 shows the first approach to the design by creating labels to understand which section is for each goal. By splitting the Join and Create room segments, the user could clearly understand the options. This design takes inspiration from “WhatsApp” by ensuring every option is labelled and clear. The prototype for this design can be found at https://marvelapp.com/prototype/c6e3526/screen/78598763

Figure 30 shows an alternative mockup to the first wireframe design. This design takes inspiration from “Google Meets” and aims to reduce clutter displayed to the user by limiting what is shown to the user to the bare minimum. The prototype of this design can be found at https://marvelapp.com/prototype/c6e3526/screen/78598734

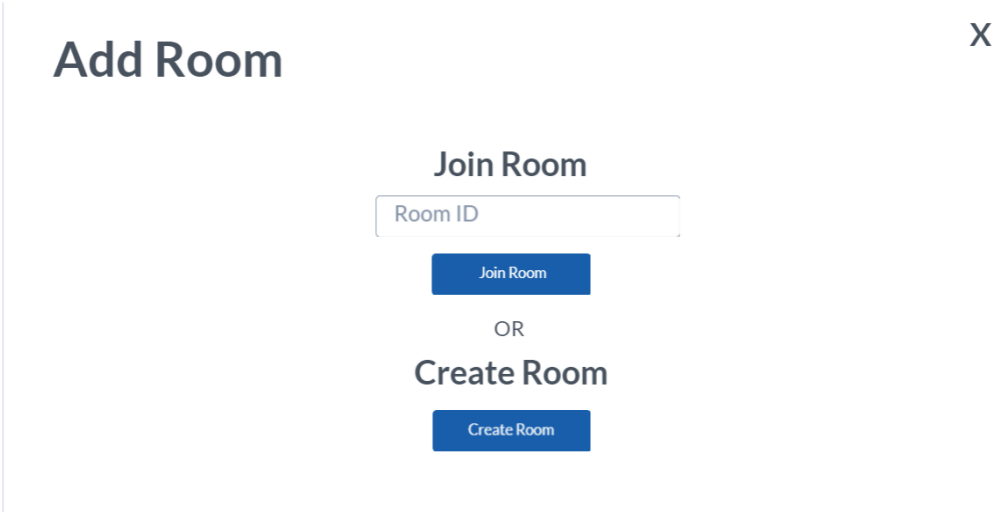


Figure 29: Add Room Page Mockup 1

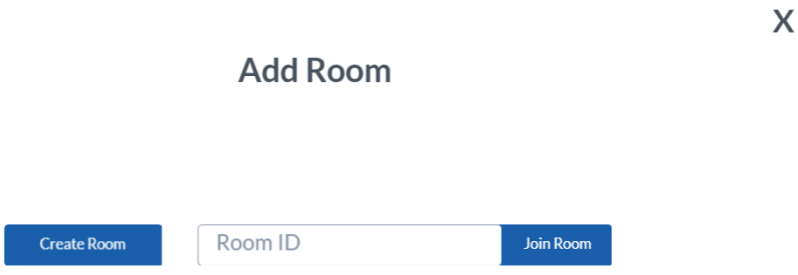


Figure 30: Add Room Page Mockup 2

3.2.5 Colour Palettes

The mockups used a generic blue style palette for all designs to limit bias in selecting the designs. The next step in the design methodology was to select a colour palette that would be used for the final design. This colour palette would ensure that the entire design was unified with a single colour palette. Ensuring that a single colour palette was used ensured that the design methodology of maintaining a singular design across the entire website was followed. Ensuring this design methodology is followed is essential, as this was outlined in the requirements analysis for the project.

To gauge user interest in different colour palettes, "Coolors.co" was used to identify ten different colour palettes used throughout the website. These colour palettes were then shown to users in a survey where they would select their favourite, and this would become the predominant theme throughout the website.

Figures 31-40 show the different colour palettes; these were selected from the trending page on coolers.co.

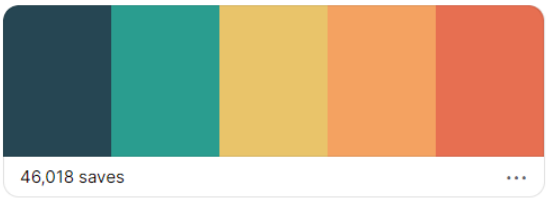


Figure 31: Colour Palette 1

Figure 32: Colour Palette 2

Figure 33: Colour Palette 3

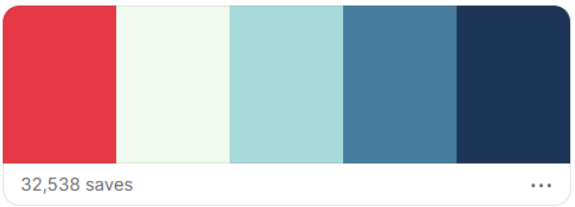


Figure 34: Colour Palette 4

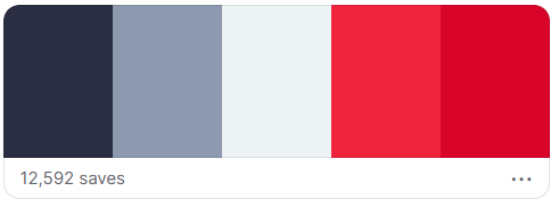


Figure 35: Colour Palette 5

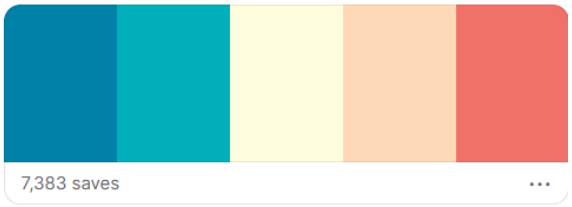


Figure 36: Colour Palette 6

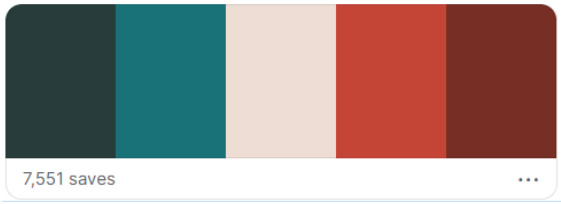


Figure 37: Colour Palette 7

Figure 38: Colour Palette 8

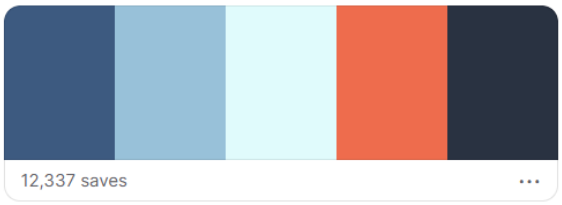


Figure 39: Colour Palette 9

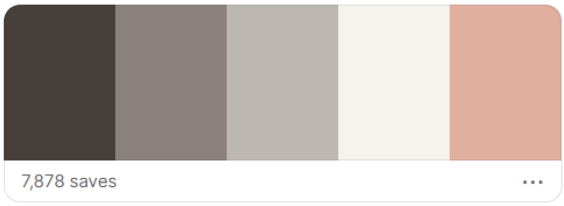


Figure 40: Colour Palette 10

The first design for the “Chat Page” was used to convey each of these themes from these palettes. The user testing would be focused on these images and narrow them down to the most popular colour palette. These designs are shown in figures 41-50

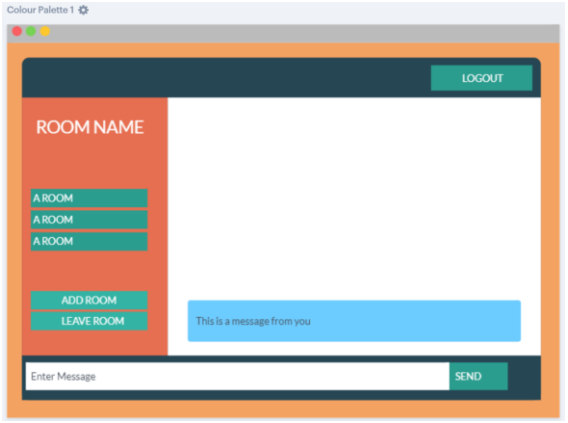


Figure 41: Colour Palette Mockup 1

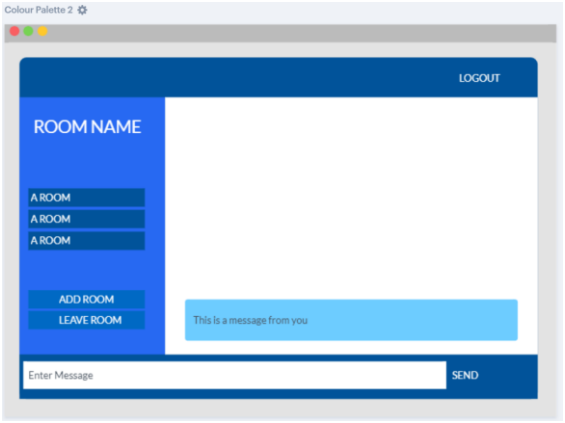


Figure 42: Colour Palette Mockup 2

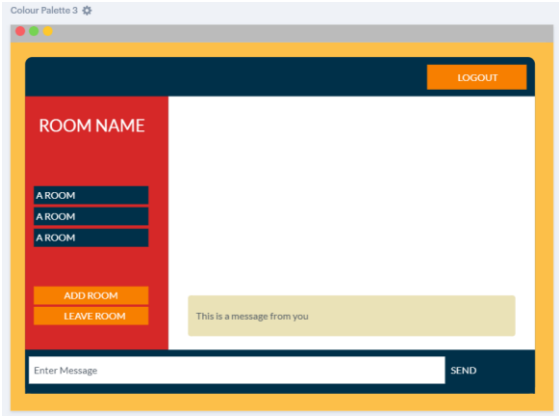


Figure 43: Colour Palette Mockup 3

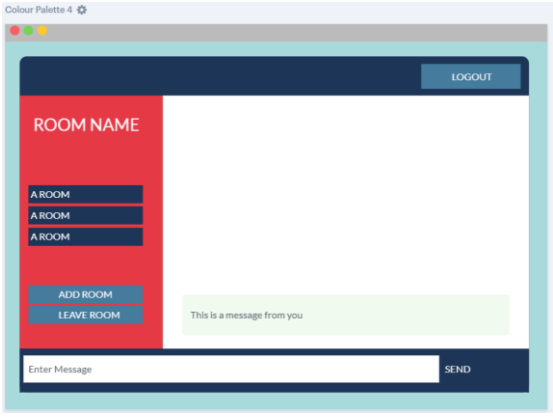


Figure 44: Colour Palette Mockup 4

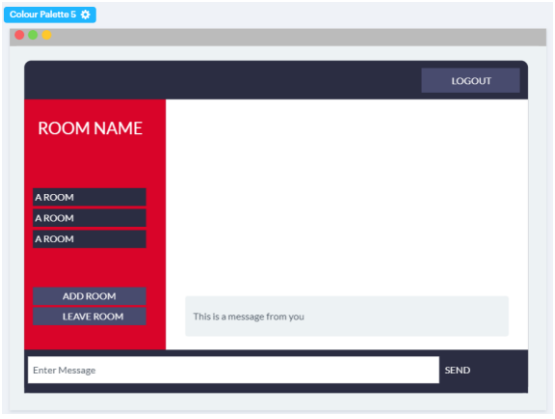


Figure 45: Colour Palette Mockup 5

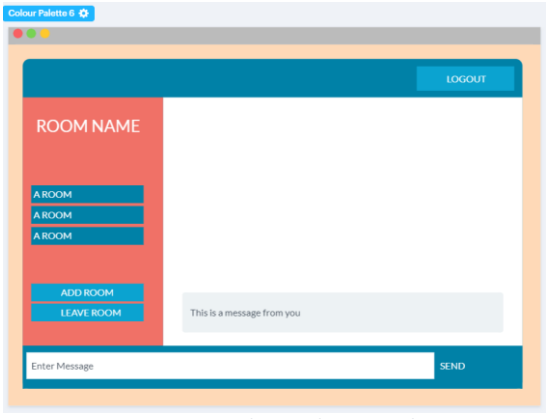


Figure 46: Colour Palette Mockup 6

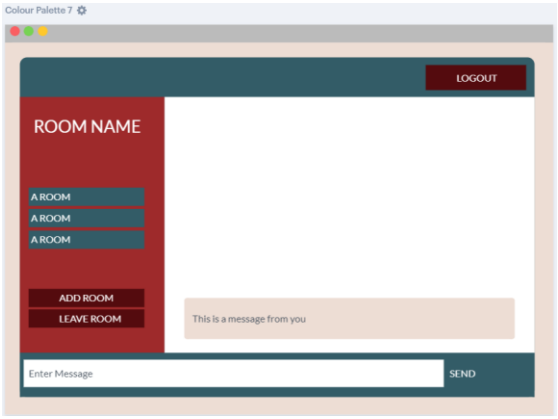


Figure 47: Colour Palette Mockup 7

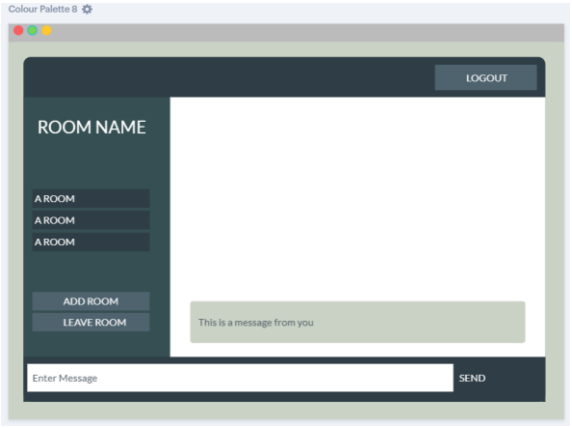


Figure 48: Colour Palette Mockup 8

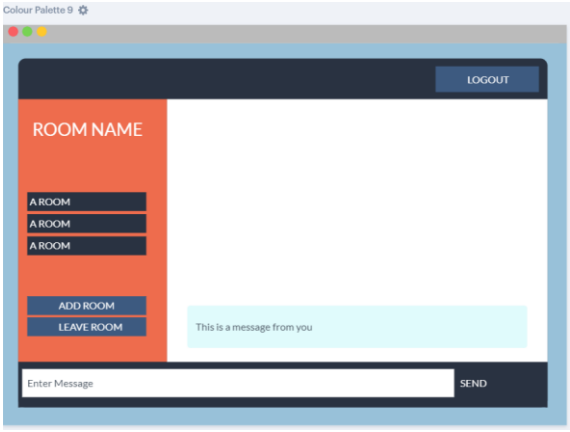


Figure 49: Colour Palette Mockup 9

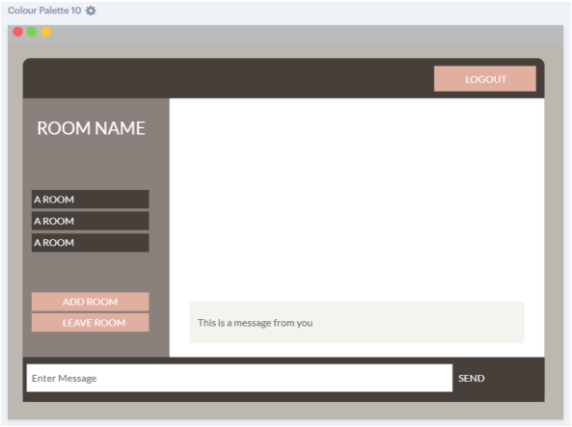


Figure 50: Colour Palette Mockup 10

3.2.6 User Feedback

The methodology used in order to gain user feedback was to create a survey. The survey was created using “Google Forms” to ensure all participants could access from their homes and visualise the data received. As the user feedback took place during a pandemic, surveying had to be completed remotely.

This survey asked the participants their opinion on each of the mockup options. By A/B Testing these mockups, user interest in each one would be gauged, and from there, a final design could be chosen. In the survey, the two designs were shown side by side using the same generic colour scheme as to not bias the results. Next, the survey asked the participant their preference on the colour scheme of the website. For this, the chat pages shown in figures 41-50 were shown. To ensure standardised feedback, the same design from figure 27 was used. Users were then asked to select their favourite

The first question in the survey regarded the login page; the users were shown two design options. These options were the mockups shown in figures 25 and 26. Users were asked for preference between the two designs. The results of this survey are shown in figure 51.

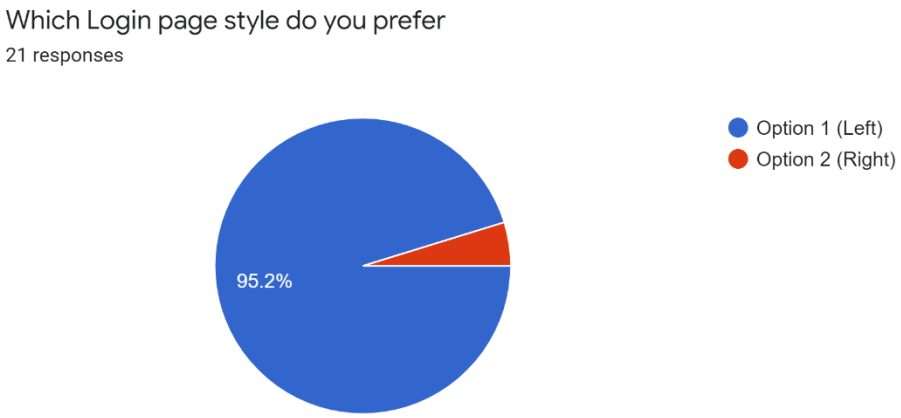


Figure 51: Login Page Preference Survey Results

From these results, all 19 out of 20 participants preferred design one. This shows a clear preference for the centre page plane with necessary and straightforward inputs. Therefore, this design would be applied to the login and register pages. Users were then asked for any feedback on the design. One response said, “I like the design is simple and reminds me of professional login pages”. The response indicated the simplicity was a crucial element in the choice; this is a theme that can be carried out through the rest of the website. Another response said, “I don't like the way the box looks in the other one”. Indicating the user did not like the plain blue box with text that is shown in the mockup in FIGURE. Again, this reinforces the preference for a unified login box with a clean and straightforward layout. The third and final response for this question stated, “I think it looks like a professional website login that I would trust”. A key element in this response is ‘trust’ as users' trust is essential in a successful website.

The next question asked again which design users preferred between for the “Chat Page” shown in figures 27 and 28. The participants again were shown the two options side by side and would decide a preference, then were asked subsequently for comments.

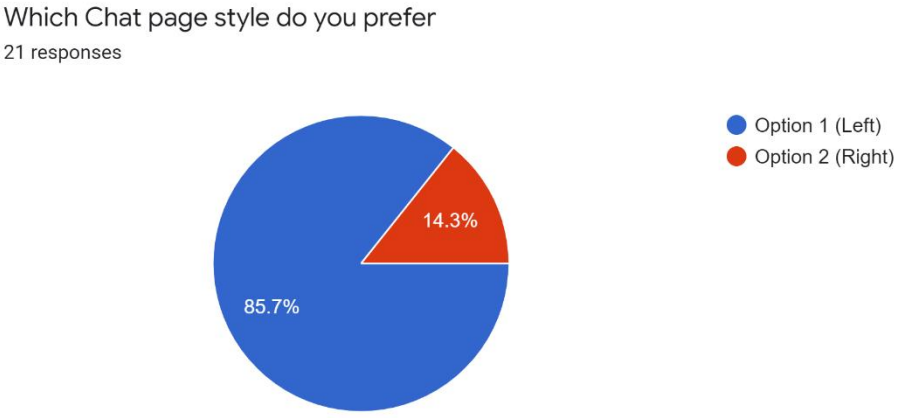


Figure 52: Chat Page Preference Survey Results

Figure 52 shows that 85.7% of participants preferred design one. Design one utilised a 'floating plane' approach to the chat page; participants in the survey showed a preference for this as stated, "I prefer the space around the content". Another comment echoed the sentiment of this feedback, "I prefer that it feels like an app". Both of these comments prefer the first design for its floating plane style and the overall design. Furthermore, one comment stated "I like the message goes across the whole area". Showing a preference to the message block taking up the entire width of the area. Design two uses shortened bubbled, text areas pinned to the left if the message is from another user and the right if the message is from the user themselves. Another feedback comment "I think it looks more professional" is an overall view of the design, in this case the participant believes the design indicates a higher level of professionalism, a trait required in any successful website.

The next question participants were asked was regarding the Add Room Sub Menu. The add room sub menu would appear to users when "Add Room" was selected on the "Chat Page". The results of this question are shown in figure 53.

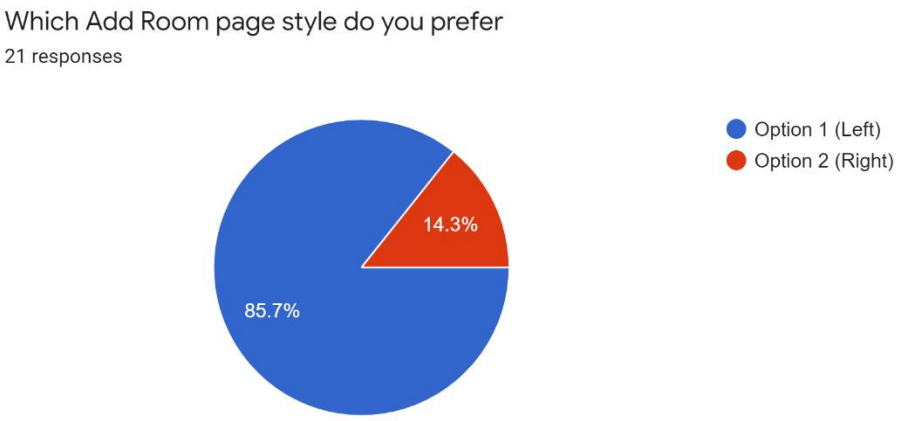


Figure 53: Add Room Preference Survey Results

From the results shown in figure 53, 85.7% of participants preferred design one. This design split the Join Room and Create Room areas with a label that said "OR" and had labels above each section indicating what they were for. This opposes design two, which took a more straightforward approach to the design and had only three elements, a Create Room button, a Join Room ID textbox and a Join Room Button, all with no labels. Participants were asked a follow-up question regarding any comments about the design choices. The first feedback comment was "I like the labels show you what each box is for". This comment directly identifies the critical difference between the two designs and shows a clear preference for design one. The next comment states, "I like that its clear what part is for what" this comment again directly indicates the segmentation in design one is preferable. However, a comment for design two stated, "I like it's simplicity", which is in direct disagreement with other comments; this may be that this participant has had some experience with similar layouts such as google meets that has led them to this preference.

The final topic in the survey asked the participants their preferred colour scheme. For this, the first design of the Chat Page was modified with colours from the selected palettes in figures 31 - 40. However, designs were consistent through all options; this was to remove bias

from design and ensure decisions were solely based on the colour scheme. The results of this survey question are shown in figure 54.

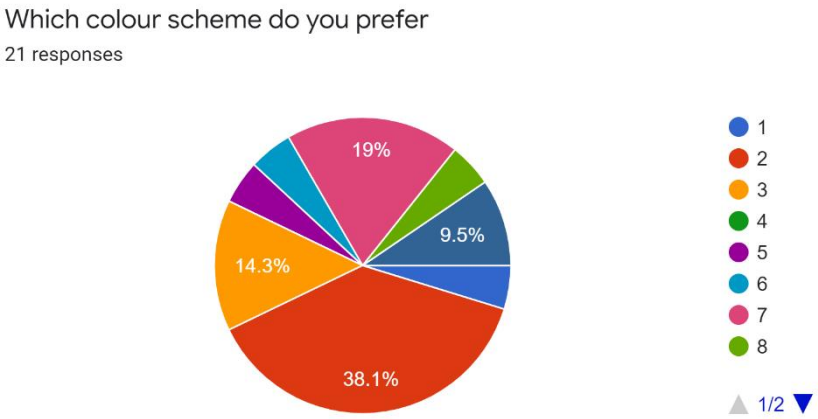


Figure 54: Colour Scheme Preference Survey Results

From the results shown in figure 54, 38.1% of participants preferred the blue colour scheme from the options given. Option seven was second with 19% of the votes, and option three with 14.3%. With the highest percentage votes, the blue theme will be taken forward to the final design phase. Two participants who voted for the blue colour scheme also left comments in the following question regarding additional comments. The first comment states, “I think the blue make me feel like its a regular social media chat app” which indicates the user has a preference for this colour due to familiarity. The familiarity of this colour could ensure users feel comfortable and safe using the website. The following comment stated, “The blue makes me feel like its a professional website” This indicates that the user feels a professionalist due to this colour scheme. This comment ties into the previous comment regarding familiarity; professionalism is a key component to successful applications and websites. This means that the user also feels secure and safe using the website; this can help with user growth and daily active users.

3.2.7 Final Designs

The final designs chosen were created using CSS and html to display to the user. Each of the three pages were created and shown below:

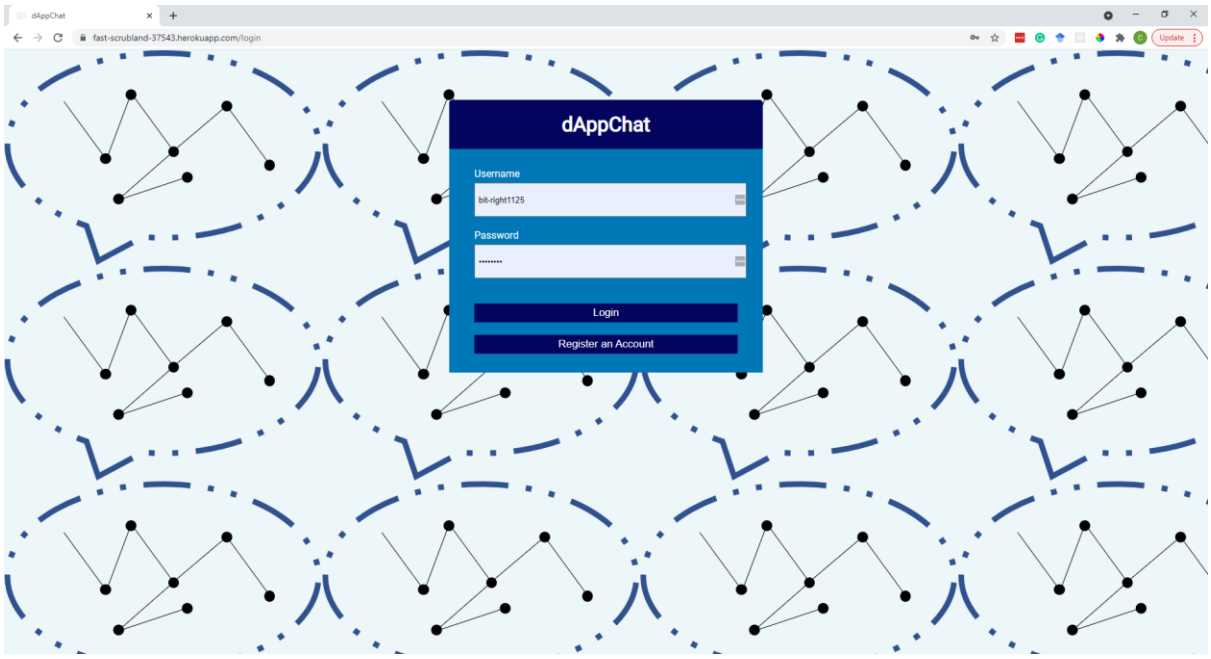


Figure 55: Final Login Page Design

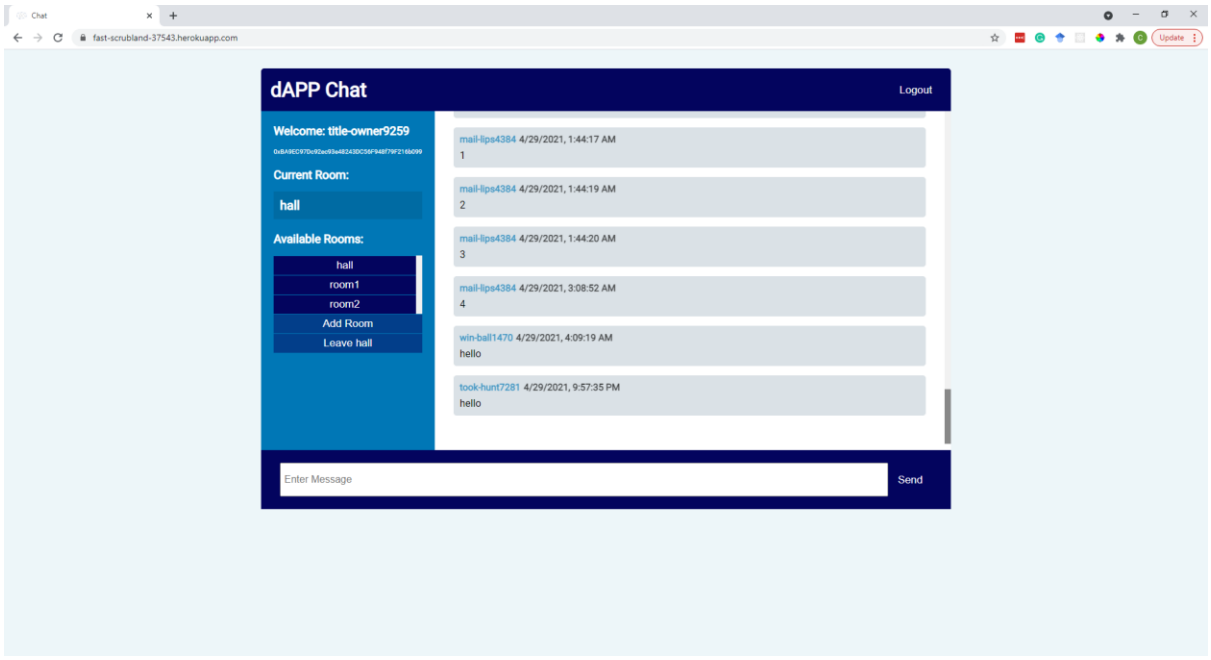


Figure 56: Final Chat Page Design

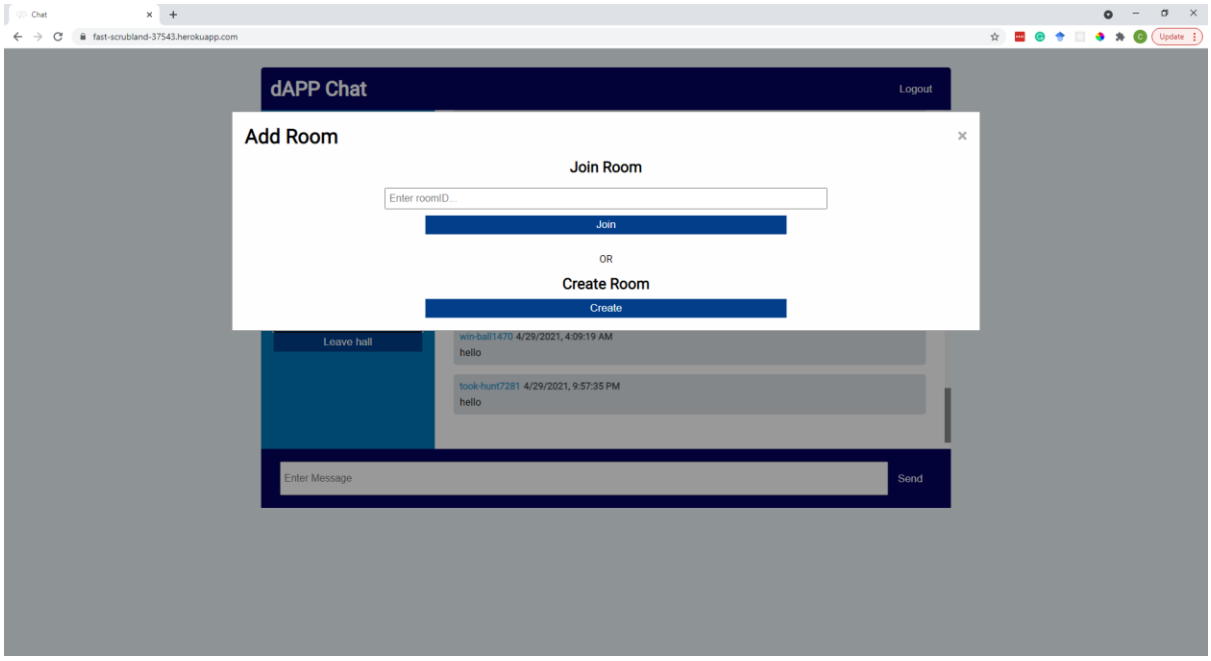


Figure 57: Final Add Room Page Design

3.2.8 Design Methodology Conclusion

In conclusion, laying out the requirements for the project at the start of the design methodology, the fundamental requirements for this process were set. By ensuring the requirements, both functional and non-functional were clearly defined, starting the design process centred around these requirements was critical. The next step of researching existing messaging applications was critical in discovering design methodologies used by successful applications to take inspiration and understand user preference currently in the market. Creating wireframes was important in understanding a foundation for the pages, leading into mock-ups and then prototypes being created. The final design creation was to research colour palettes that would ensure a cohesive and not cumbersome display. From there, the mock-ups were redesigned using these colour palettes individually.

Once completing the research and creation techniques, a survey was performed on users to gather feedback and preference. This survey was conducted to a wide range of participants. Including participants from a technical and non-technical background, older users and younger users. This helped gauge a preference for the general market rather than a specific group.

Overall, these methodologies came together to create the final designs shown in FIGURE and will proceed into the implementation section as the front-end design.

4 Process of Implementation

4.1 Introduction

This section will discuss the process in which the artefact was implemented. This section will break up the artefact into its core components and discuss the decisions made that ensured the functionality set out in the requirements analysis. As the website is designed to run in traditional web browsers, the design process is similar to that of a traditional website. However, rather than storing user messages on a server, they will be stored on a blockchain.

The implementation process will also delve into areas that were researched but ultimately abandoned, the reason for which will be discussed in their respective sections. The functionality was built using VisualStudio Code and maintained using consistent commits to a GitHub repository. The front-end design was created using HTML and CSS and was built around the designs created in the design and methodology section.

4.2 Researched Methods

4.2.1 Matrix.org

The initial research was into a technology created by Matrix.org, an open-source standard for decentralised communication. Matrix provides an API that allows developers to interact with the network they created and utilise their infrastructure to create chat rooms. The initial research into this system was promising; with substantial documentation and support, Matrix would provide an API that could take a large amount of the load of the data required for decentralisation.

The first step in development in Matrix required the research into their developer docs to understand how to create a website using their technology. To develop with the Matrix API in any meaningful way, a 'home server' must be created. For the proof of concept in this project, a home server was hosted on AWS. This links into the Matrix network and acts as a node in their network. Once this was set up, a simple app utilising the API was built using VSCode. This app had the functionality of creating and joining a public room.

After initial testing with the Matrix server, it became apparent that as the network currently has a higher requirement for data than the ability to store on 'homeservers', Matrix had to lean on utilising traditional server style storage. Although each server was federated, the network was not fully decentralised as of this project, and therefore, as one of the requirements was for the network to be decentralised, Matrix was not a good fit.

4.2.2 Custom BlockChain

Following from Matrix, research was placed into a fully decentralised option. Blockchain became the clear choice for this. Blockchain relies on an incentive system, ensuring users will "mine" or verify block transactions to continue to ensure trust across the system. Research into a custom blockchain to transact message data was conducted.

To create a custom blockchain, VSCode was once again used utilising TypeScript to implement a proof of work system on a local pc. Proof of work requires 'miners' to continually attempt to solve the 'nonce' value combined with the hash of the block transaction to verify the block. Once each block is verified, the blockchain ledger is updated and globally accepted.

Although implementing a custom blockchain was completed, the concept of a proof of work system does not inherently work without consistent miners on the network. To effectively

create a decentralised system, one entity cannot have 51% or more of the network. In this case, to host an effective system that would allow asynchronous mining of blocks, continuous transactions would need to occur on multiple systems under different domains. For this reason, it is not feasible to create a fully decentralised system without a large number of computers continuously running and to be a 'fair' system; these resources would need to be maintained by different entities. As such, creating a custom blockchain is not a good fit for this project.

4.3 Project Design

Next, research into the technologies that would be used in the development would be completed. The project design methodology had to be created, designing the structure of the website and its interaction with a decentralised network had to be laid out. Figure 58 shows the structure that was used as a basis for the development. This is a high-level design that would be used to ensure relationships between the development files would be maintained and a clear idea of the architecture would be understood.

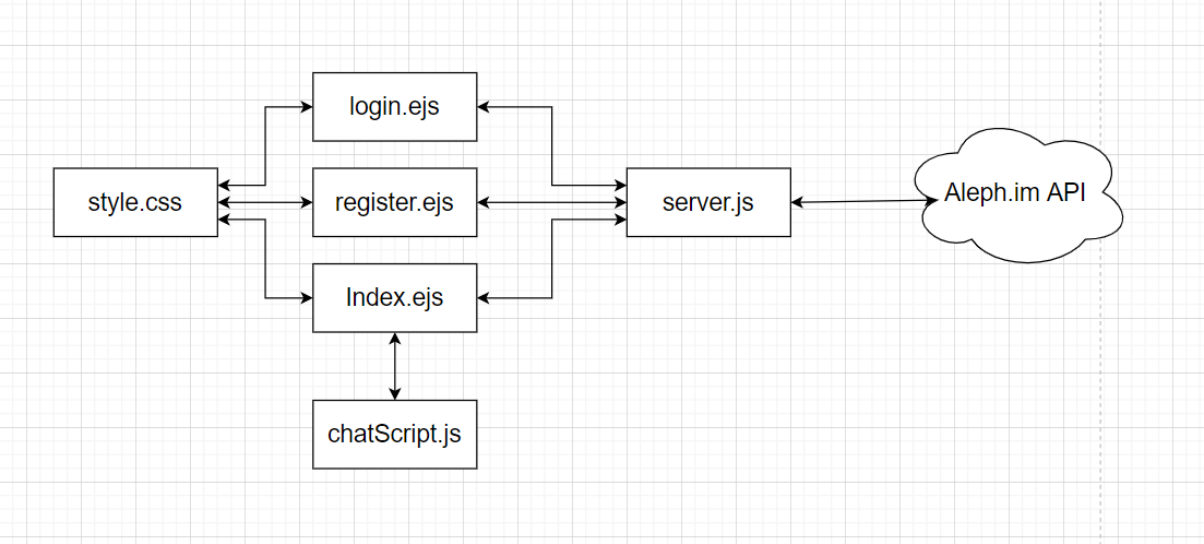


Figure 58: Pre-Development Architecture Design

4.3.1 Decentralisation

Aleph.im is a cross-blockchain network that allows for a fast transaction between a client and a blockchain. Aleph is able to connect with multiple blockchains and allows for interaction between these networks. Aleph allows for a fast transaction that send a user transaction request to the blockchain for validation. Once validated by mining, the block is legitimised and will be locked into the chain.

Aleph provides an API that allows developers to connect to multiple blockchains via the aleph Network. Aleph also handles the on the fly opening of Crypto Wallets that are required for blockchain posting.

To implement this into the project, the API documentation was referenced to ensure correct and RESTful calls.

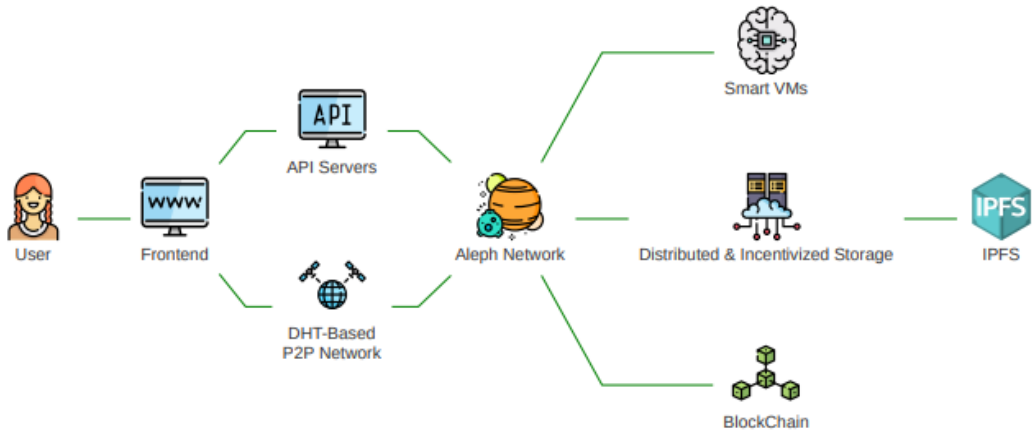


Figure 59: Aleph.im Network Architecture

4.4 Implementation

4.4.1 Front end

The creation of the website front end utilised HTML, CSS, and JavaScript in conjunction. The combination of these languages displays the content, design, and functionality, respectively. The front end of the website consisted of five HTML files, two of which were modified to ejs files to interact with the serverside. This file structure is shown in figure 60 and is the basis for the front end of the website.

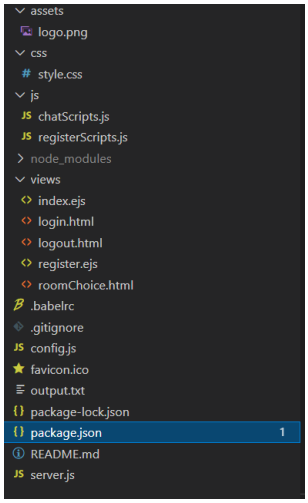


Figure 60: Front End File Structure

To help with the front end development, JQuery was implemented as an external library JQuery helps with a variety of front-end development such as DOM traversal and manipulation. For this project, JQuery also aided in AJAX calls and event listeners (jQuery, 2021).

For the login page, the user was presented with a floating plane that houses the text boxes for both the username and password. Below this, the user has the login option and the register an account option. A form is used for both the username and password to log in, ensuring that both of these inputs are required before a user can call the login request. If the user tries to login with an incorrect username or password, a message is displayed explaining that one of the entries is incorrect. This uses EJS (Embedded JavaScript) to check if the username or password had been inputted incorrectly.

The register an account option moves the user to a visually similar page to the login page. This page, however, has the username and text boxes for a new account with a button to register with the inputted data or login instead, navigating back to the login page. If the user attempts to create an account with an existing username, they are prompted that the account already exists and to try and log in instead.

When the user logs in, they are redirected to the chat homepage. By default this enters the room “hall”. The chat page is comprised of three main sections. The first, on the left houses information. The left bar is held in a div that has the class ‘chat-sidebar’ this div contains all of the elements such as currently logged in username along with their Ethereum wallet address. These are pulled across using EJS and displayed using a h3 tag for the username and a p tag for the wallet address with a custom class ensuring that the address is in a smaller font. The current room is also displayed to the user on this bar and below a list of all the rooms

a user has joined. These rooms are listed by again using EJS to loop through the users array of rooms. For each one a button is displayed that shows the name of the respective rooms. When each of these buttons are clicked, the back-end is called and the user is navigated to that room. The div that contains this list of buttons has a max height, if the max height is not enough, the list becomes scrollable with a custom scrollbar set using webkit scrollbars.

Below this list, there is an “Add Room” button, this button opens a popup that contains two segments. The first is a textbox that has the default text “Enter Room ID” this textbox asks the user to enter the room ID of an existing room they would like to join. Below this is a Join button. If the user enters a room that does not exist, the user is prompted that the room does not exist. The other part of the popup is a Create Room button that polls the server for a randomly generated room ID and adds this to the user's room list. When the user successful clicks either of these buttons the popup closes, and the chat page is reloaded with the new room in the list. The popup can also be closed using the exit icon in the top right. The popup is opened and closed by calling the function addRoom(), setting the div that contains it to the display style “block” and then the function closeDialog() which sets the display style to “none” to close the dialog. Below the Add Room button, is a leave room button. This appears if the user is not in the “hall” room as this cannot be removed. Clicking this button removes the room from the users list of rooms.

On the right two thirds of the display is the chat window. This chat window contains all of the messages sent in the currently selected room. Each message has the class of message if sent from another user or myMessage if sent from the current user. Messages sent from other users have a grey background and the text is aligned to the left while messages sent from the current user have a blue background and the text is right-aligned. Each message contains the sender, the date and time sent, and the message itself. These messages are loaded when the room is loaded, new messages are loaded using the WebSockets. The WebSockets listen for any new messages and when a new message is received, check that it has content inside and if so calls the appendMessage() function.

The appendMessage() firstly checks that the message has not already been shown by checking if the id exixts in the DOM. The function also checks if the message is valid by ensuring the body contains content, the channel is correct, and the room is for the currently selected room. Then, the function breaks the message down and gets each component from the JSON response. The message then decompresses the message using the decompress function on the serverside and adds the new message to the chat message window. The function also uses JQuery animate functions to ensure a smooth scroll to show the user the new message.

The final section of the chat page is the send message segment. There is an enter message textbox and a send button. When the send message button is clicked, the server-side is sent the information in the textbox to push to the network.

During the development of the front end, some development issues were identified. Firstly, when the user sends a message, the WebSocket would pick up the new message and display to the user; however, if the user left and rejoined the room, this could lead to duplicate messages being shown. To fix this, when the hasFoundID function is called, a check is performed to ensure the message has not already been displayed.

A second issue that was identified was that when the messages were displayed to the user, sometimes, the WebSockets would identify a new message being sent to the network;

however the message would be null. This could cause issues as when the message was to be parsed, the message could crash the server. To fix this issue, checks were implemented across the message parsing for unidentified elements. This ensured that false messages would not cause issues when displaying to the user.

4.4.2 Back End

The back end of the website utilised Node.js to run the webserver with EJS (Embedded JavaScript) to link to the front end of the website. MongoDB was responsible for holding

```
const userSchema = new mongoose.Schema({
  username: String,
  password: String,
  private_key: String,
  public_key: String,
  mnemonics: String,
  address: String,
  rooms: [String],
});
```

Figure 61: Mongoose User Schema Design

user profiles, this included their username, hashed password, joined rooms, and blockchain wallet IDs. These IDs are required to ensure that the posts to the blockchain come from each individual account, rather than the website itself. MongoDB would be connected via mongoose, an object modelling package that interfaces with Mongo. Creating a Mongoose Schema that held the users' information and created a template for the server to manipulate.

Once the front and back end of the website were functional, the project was hosted on Heroku. By ensuring the website was available online, testing could then be completed by participants in various locations and using various browsers.

When users registered an account, they would have the option to create a username. For data protection and ethical reasons, collecting user data is not viable and as such, randomly generated usernames are automatic. These usernames are generated using a function in the server js. This function utilised a package called “random-words” that has a list of words acceptable for public use. This list does not include profanity. The function to create usernames would call this package and receive two words joined with a “-“ and these words had a max length of 5 characters each for user convenience. Next, a four-digit randomised number would be created and appended to the name. This would be the random username assigned to users.

The server also routed user GET requests for URLs to the correct source file. This would ensure navigation between the login, register, and chat pages were handled. For login, “Passport” was utilised. Passportjs is a middleware for Node.js that ensures authenticated login for the website. This works with sessions, initialising when a user makes a connection and ending it when the user logs out. This ensures users only access their information and can save files to their mongoDB database file.

Interacting with the Aleph.im API was through POST requests to the server. Chronologically, the first time the user will interact with this will be through the /register POST request. This function created a new User using Mongoose register function. This takes the parameters from the user login form on the front end login page alongside the results from the Aleph API call. This call creates a new Ethereum account. This returns Ethereum private key, public key, and mnemonics. Finally, Passport authenticates this user and logs in, directing the user to the chat page.

The next use of the POST request is to send a message. The server checks to ensure the user is logged in using the connectEnsureLogin.ensureLoggedIn function from Passport. Next, the message is passed in; as one of the requirements analysis requirements was to ensure messages were compressed before transit, the package LZString is used to compress the message. Once this is completed, the Aleph API is used once again, to ensure the correct user pushed to the blockchain, the users mnemonics are used to identify the user account, then the message is posted. The message has three elements, the first of which is the address of the sender (Ethereum Address). The second item is the body which is comprised of the compressed message and the users' username. Finally, the details of the post such as room, which api_server the post is going to, the account, and the channel.

Another option the user has is to either join or create a room. To do this, another POST request is used, the requested room is passed through. The new room is added to the users' room list using the mongoose findByIdAndUpdate method. Once a user has set up a new room, they can view the room by sending a “rooms/:roomID” GET method. The user can send this by either selecting the desired room in the roomlist or navigating in the URL to this address. This method initially checks to ensure the room exists using the roomExists() function. roomExists() checks the mongoDB database for the list of active rooms. If the requested room does not exist, the function redirects the user to the hall. If the room does exist, then adds the room to the users list of available rooms and shows the user this room in their room list.

One potential issue with this structure is that as the Aleph API pulls from the public ledger, someone could send a message through their own API call and that message would appear. To solve this, the server checks when messages come through whether or not they fit the model set out. Also, the decompression checks for identifiers before decompressing as to not break attempting to decompress not compressed text.

4.4.3 Process of Implementation Conclusion

The implementation of the project fell under three main components. The first element of the implementation was the decentralisation. The process of implementation researched and understood the Aleph.im API and its interaction layers with the Ethereum network. This acted as a foundation for how the front and back end of the product would be created. Once the method of posting to the blockchain was identified in the Aleph.im API, the front end design started. Using what was identified as a popular design choice from the design/methodology section of the research, the fundamental development process was described. Using traditional web development languages such as HTML, CSS, and JavaScript, the front end could be a responsive and stylish design allowing the end-user to utilise the application with little learning curve. Next, the back end of the website was created, the implementation process detailed which tools were used for this and how each of the functions in the back end worked to ensure a smooth process of for the end-user. By breaking down these functions,

the implementation of the decentralised Aleph.im API was described alongside the use of Passport for user authentication.

Overall, the implementation process broke down the key elements of the website and how the design created in the design methodology was adapted to an actual web interface. Furthermore, this section displayed how previously researched core decentralisation technologies were implemented to ensure the requirements analysis would be satisfied.

5 Testing

The next stage in the project was to complete user testing. The testing would be oriented around the requirements set out in the Requirements Analysis. Each of the requirements in the analysis was categorised by the MoSCoW method. This method split the requirements into “Must Have”, “Should Have”, and “Could Have” priorities. Each of these priorities will be identified and tested with the outcomes thoroughly identified in this section.

Furthermore, user scenarios will be devised to identify routes users may take through the website and goals they may look to achieve while using. These scenarios will also identify if any bugs occur in that route and ensure the user experience is optimal. Once these tests have been completed, users will be polled for a Cognitive Walkthrough. This will ensure that new users can understand the system solely based on the user experience and complete tasks given. Finally, the last survey will be completed to understand the users' experience with the website, any comments they have, and whether they would consider using the website.

Through this testing, the user experience will be understood and ensure that the website fits the functional and non-functional requirements set out in the analysis and acts as a suitable website that users would use.

5.1 MoSCoW Testing

5.1.1 Functional Requirements

Requirement:	Use a decentralised network
Priority:	Must
Result:	Completely Satisfied
Notes:	The Ethereum Blockchain network was utilised to store and send messages.

Requirement:	Compress Messages Before Sending
Priority:	Must
Result:	Completely Satisfied
Notes:	LZ-String Compression library was used to compress messages before posting to the network.

Requirement:	Allow users to send multiple media types
Priority:	Should
Result:	Partially Satisfied
Notes:	Emoji sending is supported; however sub-requirement of image sending has not been satisfied

Requirement:	Be simple to set up for a new user as a traditional messaging client
Priority:	Should
Result:	Completely Satisfied
Notes:	Login page followed design queues from traditional login systems, no extra steps from end user.

Requirement:	Have a responsive and intuitive interface
Priority:	Should
Result:	Partially Satisfied
Notes:	Functionality responds within expected time of <4s. No loading page was implemented

Requirement:	Include video and audio call functionality over the same network
Priority:	Could
Result:	Not Satisfied
Notes:	Blockchain method to send messages is not a viable system for video chat; alternate methods such as P2P connection would need to be implemented.

Requirement:	Include end-to-end encryption on all messages, resulting in a safer and more secure messaging client.
Priority:	Could
Result:	Not Satisfied
Notes:	Room style architecture makes end to end encryption less viable. Research into this for future implantation would need to be completed.

Requirement:	Include group-messaging options with chat rooms for friend or family groups
Priority:	Could
Result:	Partially Satisfied
Notes:	User can create rooms for groups of people is completely implemented, sub-requirement of group calls was unable to be implemented due to network requirements.

5.1.2 Non-Functional Requirements

Requirement:	Have a client that the user can interact with
Priority:	Must
Result:	Completely Satisfied
Notes:	Traditional web development technologies were used to create the client allowing users to access from a web browser. CSS was implemented to scale to an extent for different desktop sizes.

Requirement:	Process messages within a reasonable time
Priority:	Must
Result:	Completely Satisfied
Notes:	Messages sent and pushed to network in under 4 seconds.

Requirement:	Be accessible at any time for users to either send or receive messages
Priority:	Must
Result:	Completely Satisfied
Notes:	Ethereum Network constantly has miners ensuring network uptime, testing over 24 hour period shows zero downtime. Continual monitoring of this over time will be required to ensure uptime is kept.

Requirement:	Allow multiple users to access simultaneously and send messages to any other participant of the service
Priority:	Must
Result:	Completely Satisfied
Notes:	Using Passport to authenticate users, many users can access at the same time and will not interfere with each other.

Requirement:	Adhere to data protection laws and ensure that no user data is stored other than a user-chosen password and a randomly generated username
Priority:	Must
Result:	Completely Satisfied
Notes:	Randomly generated usernames were implemented on account registration. No other user information is collected other than a password which is hashed to be stored on the server.

Requirement:	Be completed by hand in date of 9th April to show demonstration and poster
Priority:	Must
Result:	Partially Satisfied
Notes:	Hand in date for project was modified, completion of the project before the hand-in date was still completely satisfied.

Requirement:	Have appropriate documentation guiding users to utilise the application

Priority:	Should
Result:	Partially Satisfied
Notes:	Documentation was created to show users how to use the application, in-app demonstration was not implemented. This is shown in Appendix G

Requirement:	Have a consistent user interface across the entire application
Priority:	Should
Result:	Completely Satisfied
Notes:	Design research was completed and consistent design was implemented on every page of the website.

Requirement:	Be accessible to users with some disabilities that could affect the use of the application
Priority:	Should
Result:	Partially Satisfied
Notes:	Alt text implemented on necessary objects.

Requirement:	Use Node.js for development and build of back end web interface
Priority:	Should
Result:	Completely Satisfied
Notes:	Node.js was used on server side of website with Express to handle routing. Heroku was used to host instead of linux back-end

Requirement:	Be accessible from connecting to a regular internet browser
Priority:	Should
Result:	Completely Satisfied
Notes:	Traditional web development technologies used to ensure a connection is possible from a regular web browser.

Requirement:	Use a pre-defined network such as Matrix or Ethereum to host the application messages
Priority:	Could
Result:	Completely Satisfied
Notes:	Ethereum network used to host messages rather than the server.

Requirement:	Be built on a custom network, hosted by either home server devices or sandbox Virtual Machines on AWS or similar.
Priority:	Could
Result:	Not Satisfied
Notes:	After research, custom blockchain network would not provide the stability and uptime required to satisfy other requirements.

Overall, these requirements were set out to define what needed to be completed to regard the website as a successful implementation of the original project goal. Using MoSCoW, these requirements were set out into Must, Should, and Could have. All Must Have main requirements were fully satisfied, ensuring the project's core goal was fully implemented. Should have requirements were largely satisfied with some exceptions laid out in their respective breakdowns. Some of the Could Have requirements were met, as their priority was the lowest of the requirements. Not all of these were able to be implemented, such as the design of a custom network.

5.2 Cognitive Walkthrough

5.2.1 Cognitive Walkthrough Goal

The goal of a cognitive walkthrough is to understand whether users who have never used an application are able to understand how to use it without prior instruction. To do this, a user will be posed with multiple tasks that explore different functionality of the website. By allowing the user to complete these tasks without input from an external source, the websites' inherent ability to understand can be gauged and quantified. If the user struggles to complete tasks outlined, the website design and feature set may have to be redesigned or modified; however, if the user can navigate the website and complete tasks without aid, the website design will be proven successful from both a visible and functional point of view.

5.2.2 Cognitive Walkthrough Participant

The participant that volunteered to complete the tasks is a twenty-two-year-old college student who has used many messaging applications but never a decentralised one. The participant has an understanding of popular consumer-based technologies however limited knowledge of more complex systems. As the website's goal is to replace current messaging services seamlessly, a user who has knowledge of how to use these and an indifference to the technology behind them is a prime candidate and a good representation of the general public and target demographic of the website itself. This participant will first view the website and have had no training on what the application is or does. Ensuring the user has no expectations will give a good representation of their first interaction with the website if they were to use it after production.

5.2.3 Metrics Gathered

The user will be measured on four aspects of their response.

- Firstly the user will be measured on whether they were able to complete the task. This gives a good indication of whether the solution is not clear or viable.
- Secondly, the user will be measured on the route they took to complete the task; any deviation from the simplest solution will be noted.
- Next, the user will be measured on the confidence of their task completion. As this is subjective, the user was asked to identify when they completed the task and why they believed they had.
- Finally, the user will be measured on their ability to complete tasks again, for this, the user was asked to complete each task three times. Any deviations in responses will be noted. The users' ability to learn and modify movements based on more use with the website will be measured by completing this.
-

5.2.4 Results

Task	Use application at https://fast-scrubland-37543.herokuapp.com/
1	When provided with a blank PC, the user was able to understand that this was a URL and to open a web browser, the user chose Google Chrome and proceeded to type in the URL.
2	The user typed the URL with no issues other than the length of the URL itself, as Heroku generates this, it is longer than a production URL would be.
3	The user was very confident in how to complete this task, the user noted that they understood this was a URL and how to access the website. The user noted they knew they had completed the task when the website loaded and they were presented with the login page.
4	Each time the user completed the task, they were able to open the website in a similar process. The only variation was when the URL was being typed in, the users' browser auto-filled instead of typing each character individually.
User comments	The user commented on the length of the URL and that is did not correspond to the website, as mentioned in production a custom URL would be selected.

Task	Create an account and log in
1	The user, presented with the Log in page, immediately understood to select the Register an Account page and saw the preset username. The user then typed in a password and clicked Register. Once clicked the website logged in the user automatically.
2	The user immediately understood the simplest way to register an account was to navigate using the on screen button.
3	The user was confident how to navigate to the register page, as the auto-generated username is different to other applications, the user hesitated on this section however, without prompt, they understood and entered the password. The user noted they had completed the task and stated that they understood the log in had been completed because they were navigated to another page and their username was noted on the left.
4	After the first time, the users' hesitation at the auto-generated username had disappeared. The user completed the task faster each time and was able to understand the process more each time.
User comments	The user noted that they did not understand the auto-generated username but liked that it was made up of both words and numbers and not just numbers.

Task	Send a message to two different rooms
1	The user immediately understood how to send a message to the room they were automatically entered into. The user then noted the list of preset rooms on the left and immediately clicked on one of the alternate ones and sent a message to that room.

2	The user took the fastest route to the task; the user immediately understood the room list and clicked on them to navigate to that room.
3	The user was confident in their movements, the sending of the messages was an instant understanding of how to achieve this as the layout was “similar to other websites I have used”. The user noted immediately when they had completed the task and they knew they had achieved it as the message had appeared in each room with the “Me” identifier and the formatting looked different than other peoples messages.
4	Each time the user completed the task, the process was faster. The user also identified there were three default rooms and was able to differentiate between them.
User comments	The user noted that the room list could have different formatting if it was the room they were in as sometimes, although written in the top, it was unclear what room they were in.

Task	Create a room and send a message to that room
1	However, the user hesitated; they identified that the “Add Room” button was an option and selected this. When presented with the “Add Room” page, the user typed into the join room textbox. Once realising this was under “Join Room” the user then clicked “Create Room”.
2	The user took the fastest route by navigation, however used the “Join Room” text box when not needed, this slowed down the process of achieving the task.
3	The user was not very confident in this task. After initially being unsure of what to select to create a room, the user realised Add Room was the required option. Next, the user was unsure of what was required to Create a room. However, although the user was uncertain, they achieved the task and understood when the task had been completed. After which they noted they understood the task was completed as the a new room was in the room list.
4	After the first time, the user understood how to create a room with no hesitation.
User comments	The user commented that they were initially unsure, however, they acknowledged that the misunderstanding with the text box for the create room was clear upon reflection.

Task	Join the room aH8y-4zte-1fb6 and send a message
1	The user was able to add the room using the same method as creating a room.
2	The user took the fastest approach to add this room, understanding to use the text box and add room button to accomplish this first time.
3	The user was highly confident in this; after creating a room, they quickly understood how to join a room. Once joined, they noticed the room was on the left menu and understood to join and see previous room messages. The user understood this was when the task was completed, and they noted that they understood this as the room ID was visible on the left.
4	The user completed this task at ease all three times with different new rooms, although on the third, the room list became scrollable, and the user did not immediately see the added room. This was a cause for hesitation however as a scroll bar had appeared, the user understood to scroll.
User comments	The user commented that the room ID was long and confusing, however acknowledged that as the room ID needed to be unique this may be the reason for this.

Task	Log out of the website
1	The user immediately saw the Logout button and clicked on it.
2	The user took the fastest approach available as part of the website, the user immediately saw and understood what to do with the button.
3	The user was extremely confident in this; they could complete the task very quickly and confidently. They commented that they understood the task had been completed when they saw the login page again.
4	The user completed this task three times, they immediately identified how to log out and proceeded to complete it quickly.
User comments	The user commented that this task was the easiest and they understood how to complete it as the logout button on other websites they used is often in the same location.

The results of this Cognitive Walkthrough show that a user with no prior interaction with the website could navigate and complete tasks that would utilise the core functionality. The user was able to complete all tasks assigned with minor issues and understood upon reflection and repetition the solutions each time. The user comments throughout will be taken into consideration for final edits to the functionality of the website.

One comment made by the user noted that the auto-generated username was initially confusing; research will be conducted into a potential design change to make it clear the user does not have to select a username.

The next change that may be made to the project would be with the “Add Room” button, the user did not seem confident that to not only join but to create a room, this is where they would need to go. The button may be changed to Join/Create Room or similar in order to help this. The button may also be relocated away from immediately under the rooms list to remove the confusion and make it clear that they are different buttons.

5.3 Scenario Testing

The following testing stage is scenario testing; this involves setting out different scenarios that a user may follow when attempting to use the website. This testing is to identify whether the project has any bugs or flaws that may appear while attempting to use the functionality. The following scenarios were performed on the project, the results of these tests are described in each of the following tables. The tables will outline each of the scenarios, their results, and if any bugs were found.

Scenario 1:	The auto-generated username appears on the register page
Description:	This scenario tests the auto-generated username. Each time a new user registers a different name should appear. The name should be comprised of two words with five or fewer characters each. These words should be split by a "-", and then these words should be followed by a random four-digit number.
Results:	Each time a user attempts to access the register account page, a new randomly generated username is generated, and the username textbox is automatically filled with this.
Bugs:	No bugs were found in this scenario during testing.

Scenario 2:	The user can navigate between rooms by clicking the corresponding button.
Description:	This scenario tests to ensure that the list of rooms on the left side menu allows the user to navigate them by clicking each button. This should then remove the chat messages currently shown and replace them with the messages from the room selected.
Results:	The buttons performed as expected, each time a new button was clicked, the correct messages were shown in the message pane and the current room label changed to the selected room.
Bugs:	No bugs were found in this scenario during testing.

Scenario 3:	A room can be created
Description:	This scenario tests to ensure any user can create new rooms, these rooms should have an auto-generated roomID in the form of three sets of four random characters. They are comprised of numbers or letters. These three sets should also be split by "-". Finally, the new room should appear on the left room list menu.
Results:	Each time the create room option was selected, a new randomly generated roomID was added to the list on the room list menu. Clicking on this would open the new room.
Bugs:	No bugs were found in this scenario during testing.

Scenario 4:	A room can be joined
Description:	This scenario tests to ensure any user can join pre-existing rooms. By entering a room code the user should be able to add the room to their current list of rooms. If they have already joined that room, the room should not be added twice.
Results:	When the join room button was selected with no input, the form did not allow the user to join any room. If the room code entered did not belong to a room, the user was alerted that the room does not exist. However, if the room code entered was valid, the room was joined and became available on the left room list.
Bugs:	No bugs were found in this scenario during testing.

Scenario 5:	Many rooms are added
Description:	If many rooms are added, the room list bar should become scrollable as not to extend the page solely based on the rooms joined.
Results:	When the user either creates a room or joins a room, the list of rooms is extended. The room list itself extends to accommodate within the space dependant on the device display size. If more rooms get added, the list becomes scrollable, and a scrollbar will allow the user to select their room.
Bugs:	No bugs were found in this scenario during testing.

Scenario 6:	A room is left
Description:	A user leaves a room that they are currently in
Results:	When the user selects “Leave Room”, that room is removed from the rooms on the left list.
Bugs:	The user does not automatically leave the room; although the room has been left, the user can still see the messages until manually selecting another room or logging out.

Scenario 7:	A non-logged in user attempts to access the chat page
Description:	This scenario attempts to bypass the login page by manually redirecting to the chat page. This is at the “/” route.
Results:	When the user attempts to enter the “/” page, if they are currently not logged in, the website redirects them to the “/login” page rather than loading the “/” route.
Bugs:	No bugs were found in this scenario during testing.

Scenario 8:	A user registers a username that already exists
Description:	This scenario attempts to register a username that already had an account in the database.
Results:	When the user attempts to register, a message will appear on the register page showing that the the username has already been registered and the user is prompted to login instead.
Bugs:	No bugs were found in this scenario during testing.

The results of the scenario testing show that all scenarios were successful in completed their given task. The scenario that attempted to leave a room shows that the room, although left, does not navigate a different room. Therefore the user can still view messages. To combat this, research into improving this design will be made and this minor bug will be patched in the next push. Otherwise, scenario testing shows that the website functionality is as expected and that all main functionality performs under these testing scenarios.

5.4 Design and Implementation Changes

Once the testing had been completed, from feedback some design and implementation changes were made, firstly when a user creates or joins a room. They are automatically redirected to that room. Similarly, when a user leaves a room, they are redirected to the Hall room. Users cannot now leave the hall as this is a core room that is used as a standard.

From a design perspective, some small details were changed. When a user now joins a room, the selected room has a changed design. The room is shown to have a larger space and font size in the room list. Alongside this, the colour of the button has changed. This is to make which room the user is currently in clearer. These design changes are shown in FIGURE. No changes were made to the Login or Add Room pages.

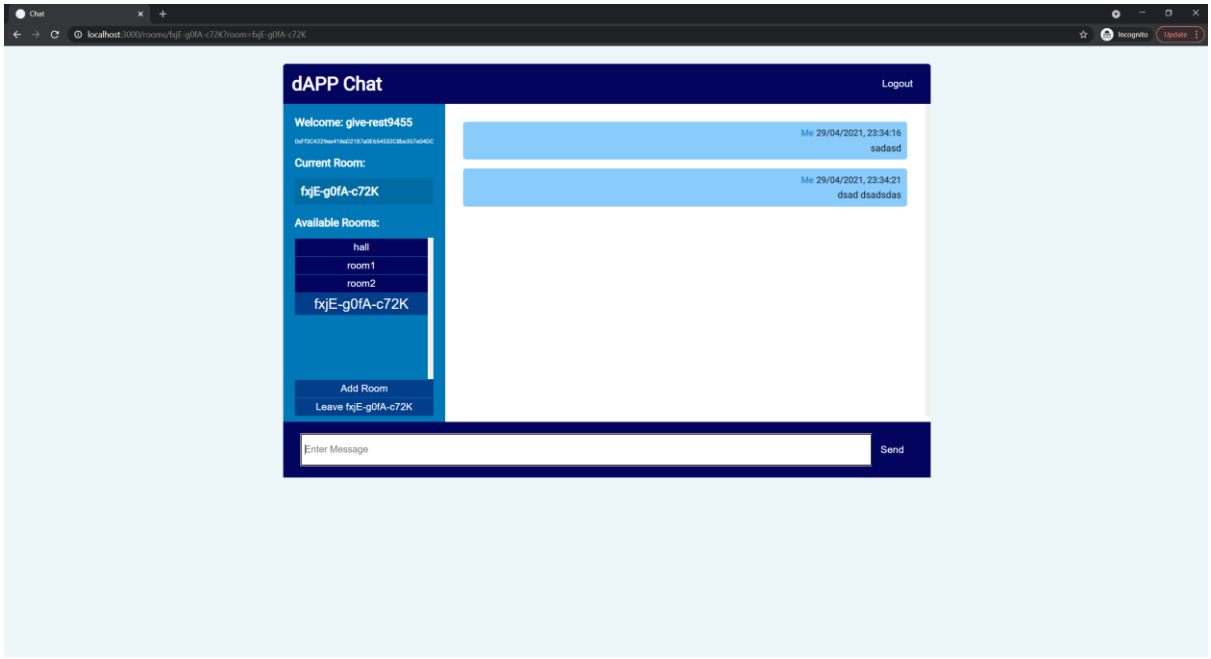


Figure 62: Redesigned Chat Page after Testing

5.4 Testing Conclusion

In conclusion, following the requirements analysis set out in the design and methodology section of the report, the testing proved an essential stage in the development. Three testing methodologies were performed each with a different goal. The first, directly analysed each of the requirements set out in the analysis and aimed to uncover whether they were achieved or not. This is essential as the requirements analysis sets the core functionality and experience targets. The second testing methodology was to run a Cognitive Walkthrough, this helped gauge how the website was in the hands of a regular user in the target demographic. This would ensure the design was not too technical i.e., using technical jargon. Or that the design was not too simple and still allowed users to utilise the functionality. Finally, scenario testing was completed, testing scenarios set out to test the functionality of the website and ensure that there were no bugs while using this functionality. Overall, the testing proved very successful in understanding this, as the results were analysed, modifications to the website were made to accommodate these. These are shown in figure 62, which outlines the final design to the chat page after modifications to the website.

6 Evaluation

6.1 Overview

The project, overall, succeeded in the goal it set out to create a messaging client over a decentralised network. The development process underwent multiple stages to progress from conception to implementation. The website design adhered to industry-standard techniques, utilising user surveys and iterative designs to create an intuitive and aesthetically pleasing design. The implementation of the project followed from the design phase, satisfying the “Must” have requirements set out in the requirements analysis. The implementation also researched alternate methods to complete the task and explained and justified the choices made in the chosen methodology. Finally, the testing ensured a broad range of tests were completed to understand from a user perspective whether the project was viable post-development.

6.2 Design

The design process was an iterative approach involving user surveys and multiple test designs. First, by researching alternative options currently on the market, design queues could be understood and implemented in the project. From there, multiple wireframes were created to understand the core layout the design would take. Next, mock-ups were created of each of these wireframes. Logo designs followed a similar methodology. Finally, these mock-ups were shown to various users using a survey to decide on the final designs. Using user feedback to create and design the interface for the website ensured that the design was appealing to end-users and understood the users' initial needs and wants.

6.3 Implementation

The implementations stemmed directly from the functional and non-functional requirements set out in the requirements analysis. These requirements detailed the minimum functionality for the project to be considered a success. As the “Must” have requirements were all satisfied alongside a large portion of the “Should” have requirements and some of the “Could” haves, the project succeeded in the goal set out in the proposal. The project utilised a decentralised network in the Ethereum blockchain to host the messages sent from the website. Using the leph.im API to interact with this blockchain ensured that the core principle of the website was satisfied; additional functionality such as rooms and login systems were then built from this basis. A secure login system using the “Passport” library ensured that the login system allowed users to maintain personal accounts and message other users. Compression libraries maintained the requirement that messages would be compressed before transit, ensuring the load on the network was minimised. On the front end, the design methodology results were used as a structure, and CSS ensured the layout of the final design matched the mock-ups. With JavaScript to ensure front end functionality felt seamless to the end-user. Overall, the implementation was successful in the goals laid out in the project proposal.

6.4 Testing

The next stage of the project was to ensure that the functionality of the website, combined with the design, created a website that the target market could use. To do this, multiple forms of user testing were created. The first of which directly related to the MoSCoW requirements set out in the requirements analysis. This testing analysed each requirement and noted whether it was satisfied by the result. From this testing, it was clear that the core requirements were satisfied; however, the extended requirements would be moved to future

project developments. From the MoSCoW testing, the next phase was to build a cognitive walkthrough. This user testing method set a user within the target market a set of tasks. The methodology then analysed each of the users' movements and results to the given task to understand the application learnability from new users. The result of this method was also a success, the user was able to complete all of the given tasks with little to no difficulty. From these results, it was clear that the website's complexity did not outweigh the usability, a design paradigm outlined throughout the literature review of an issue that some technical programs have for end-users. Finally, the testing was completed by undergoing scenario testing. This portion of the testing posed some scenarios that users would face while using the website. The goal of this section was to test functionality and user routes to ensure no bugs were present. Once this testing had been completed, it was clear this section was also a success. Some minor bugs, such as the user not automatically leaving the room, were identified in scenario 6. The next stage of the development was to combat this bug and ensure that users were automatically removed from the rooms once leaving. Overall, this testing methodology was vital in identifying this bug and ensuring other user routes were not compromised by bugs. In conclusion, the website underwent three types of testing which analysed the final implementation of the project. The overall consensus from this testing is that the website succeeded in its original goal and satisfied the core requirements set out while also passing the end-user testing set to it.

6.5 Project Aims and Objectives

Once the project reached completion, the aims and objectives set out at the beginning of the project had to be reflected upon. Five objectives were outlined; the first was to design an interface with which users could interact using user feedback to drive the design. From the design methodology section, it is clear this objective was satisfied fully as the main force behind the design methodology section was user feedback. This sculpted the design from feedback and comments from the survey participants. The next objective was to research current solutions and develop an understanding of why they were successful. The literature review section outlined current market options and explained the technologies used from an implementation perspective. Combined with the market research completed for the front end in the design and methodology section, this aim has also been fully satisfied.

The third aim was to ensure that the project was accessible using a standard web browser. The implementation section outlined the core web technologies used to ensure this. As such, this objective has also been satisfied. Further research could be performed to ensure that the website is also accessible from a native app and all mobile devices.

The fourth aim was to ensure that all user messages are stored only on a decentralised network. The process of implementation section outlines the use of the Ethereum blockchain with the Aleph.im API to push and pull from this network. The use of these technologies ensures that this aim was also satisfied in the final product.

Finally, the project aimed to develop a messaging app that could send and receive messages to different users on the system. Combining the entire development process from the design and methodology to the process of implementauon and the testing sections, ensured that the final product in the form of the website, was able to complete this. Using technologies such as Passport and NodeJS to ensure a responsive web interface with the ability for users to securely login to personal account ensured that this final aim has also been satisfied.

Overall, all of the aims and objectives set out at the beginning of the project have been satisfied throughout this project and its development cycle. These aims set a foundation of the goals to achieve from the project and drove development to achieve them.

6.6 Project Limitations

Although each of the requirements set out in the requirements analysis was identified and analysed during the testing. Some functionality of the project was unable to be completed due to various reasons. Firstly, a “could” have requirement outlined that video or audio calls would be a feature that users may want in a chat website. Due to restrictions with the decentralised blockchain Ethereum, a decentralised video or audio call could not take place over this network. As blockchain works by posting to the ledger, a call would not work with this architecture as a continual bi-directional data stream would be required. If a video or audio call system were to be implemented, a separate P2P system would need to be used for these calls rather than the current design.

The following limitation of the project is that it currently uses a pre-defined network. As this network relies on the stability and availability of the Ethereum blockchain, a custom network could remove issues such as this. To combat this, a network solely for communication over this website and potential future sister applications would help ensure the project’s longevity. Another limitation of the project was to encrypt messages being sent across the network fully. Due to the room based architecture used in the final design, encrypting messages end to end encryption was unable to be implemented. Further development of the project would look to ensure that this was a core feature in all messages across the website.

Overall, the website's current state fully satisfies the core requirements set out for this project; however, the next stage in development for this website would be to look into the requirements that were not satisfied as they posed the most significant limitations on the project.

6.7 Legal, Social, Ethical, and Professional Issues

All messaging services have an inherent issue that malicious users could use their service to send or organise illegal activity. As the messaging would not run through any server or message screening, this would have the potential to allow these users to abuse the service for nefarious needs. However, this is an issue that is pertinent to any available messaging service that does not screen the messages of its users.

From a user perspective, when signing up for the website, the service automatically generated a random username. Doing so prevents any GDPR issues of storing user data such as their name or email address. Further to this, any posts to the network are posted using only their Ethereum wallet public address. When the user closes the browser or logs out of the service, their login session is closed, and their messages can no longer be viewed without a new login. The login password is stored on a server and hashed for security purposes, again following guidelines set out by GDPR.

7 Conclusion

7.1 Reflection

The goal set out at the start of the project was to create a decentralised chat application. The literature review outlined in this report discussed and researched current options for this available and the technologies used in the market. This research showed that there are currently many options; however, they do not target the general public but a more technically aware user. Next, the project discussed the design methodology. Using user surveys and research to design a coherent easy to use interface for the project. Moving into development, building on this design to ensure functionality as implemented. Finally, resulting in testing, using multiple methodologies to understand how successful the result was in achieving the original goal. Learning from the testing and adjusting the design and implementation accordingly to address the roadblocks identified.

Each stage of the research to development had a critical reflection, which helped create a structured approach to each next phase. Doing this ensured that the project learned from each phase and understood what that research gained. Research in the literature review ensured that the project would not fall victim to issues identified in the market and ensured that the project would understand successes and failures. The design implementation guaranteed that the final design would be usable by all demographics, while the implementation took the learns from the design and ensured functionality did not impede usability. Finally, the testing section culminated the feedback and results from all of these sections and ensured that the resulting project satisfied the goals set out to ensure success.

In conclusion, the project, overall, completed the goal set out in the title of this project. The functionality of the project successfully stored all user messages on a decentralised network. Each of the requirements set out in the requirements analysis were critically reviewed and the core “must” have requirements were all satisfied. The unsatisfied requirements were not of core functionality that would impede the success of the resulting project. The testing phase ensured that the final project factually succeeded in the goal and satisfied the requirement that end-users could use the website. Each phase in the design process was user-oriented and, using surveys, ensured no bias was present in the final design phase.

7.2 Future Development

Future development of the project could be built around the un-satisfied requirements laid out in the analysis. Firstly, allowing for fully encrypted messaging would be a step towards the website's goal for fully decentralised secure messaging. Further user features such as audio and video calling could also be implemented on top of the existing messaging functionality. Adding functionality such as this could help users of other messaging services adopt the website as their default messaging system. Once a critical mass of users adopts the service, the research could return to a custom network rather than building on the Ethereum network. Moving to a custom network could expand the services of the website further than messaging. New features such as file transfer and image transfer could also be implemented as laid out in the “could” have section of the requirements analysis. Features such as this would further close the gap in functionality between the project and other services on the market. Finally, production could progress into research into a native application for devices

such as iOS and Android. By creating applications dedicated to these platforms, the product could also ensure simplicity and ease of use for social messaging on the go. If the features set out in this section were implemented in a future release of the product, the goal to act as a viable replacement for traditional centralised services would become simpler to market to non-technical users alongside the security and safety benefits of the architecture.

8 References

Aiimi, 2021. *Data Ownership: What is it and why does it matter?.* [Online] Available at: https://www.aiimi.com/insights/the-case-for-data-ownership [Accessed 4 2021].

Bagnoli, F. et al., 2016. Incentive Mechanisms for Crowdsourcing Platforms. In: *Internet Science.* s.l.:s.n., pp. 14-29.

BBC, 2018. *Encryption on Facebook Messenger and other chat apps.* [Online] Available at: https://www.bbc.co.uk/news/newsbeat-43485511 [Accessed 2020].

Berry, E., 2019. *Who owns the content on social media?.* [Online] Available at: https://newsroom.unsw.edu.au/news/business-law/who-owns-content-social-media
[Accessed 2020].

Bhageshpur, K., 2019. *Data Is The New Oil -- And That's A Good Thing.* [Online] Available at: https://www.forbes.com/sites/forbestechcouncil/2019/11/15/data-is-the-new-oil-and-thats-a-good-thing/?sh=160664127304
[Accessed 2020].

Binance Academy, 2020. *What is a 51% Attack?.* [Online] Available at: https://academy.binance.com/en/articles/what-is-a-51-percent-attack [Accessed 11 2020].

Blackmon, M. H., Polson, P., Kitajima, M. & Lewis, C., 2002. *Cognitive Walkthrough for the Web.* s.l., s.n.

Cellan-Jones, R., 2020. *Google hit by landmark competition lawsuit in US over search.* [Online] Available at: https://www.bbc.co.uk/news/business-54619148 [Accessed 2020].

Cloudflare, 2020. *What is DNS?.* [Online] Available at: https://www.cloudflare.com/en-gb/learning/dns/what-is-dns/ [Accessed 2020].

Corbyn, Z., 2018. *Decentralisation: the next big step for the world wide web.* [Online] Available at: https://www.theguardian.com/technology/2018/sep/08/decentralisation-next-big-step-for-the-world-wide-web-dweb-data-internet-censorship-brewster-kahle
[Accessed 2020].

Cornelli, F. et al., 2002. *Choosing Reputable Servents in a P2P Network,* s.l.: s.n.

Dent, A. W., 2008. *A Brief History of Provably-Secure Public-Key Encryption.* s.l.:s.n.

Deterding, S., Dixon, D., Khaled, R. & Nacke, L., 2011. *From Game Design Elements to Gamefulness: Defining Gamification.* s.l., s.n.

Dongdong Yue, R. L. Y. Z. W. T. C. P., 2018. Blockchain Based Data Integrity Verification in P2P Cloud Storage. In: *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS).* s.l.:s.n., pp. 561-568.

Edwards, J., 2020. *Investopedia.* [Online] Available at: https://www.investopedia.com/articles/forex/121815/bitcoins-price-history.asp

Ermoshina , M. K. & Francescaand , H. H., 2016. *End-to-End Encrypted Messaging Protocols: An Overview.* s.l.:s.n.

Freenet, 2020. *Freenet.* [Online] Available at: https://freenetproject.org/ [Accessed 2020].

Geared App, 2020. *KPIs: How to Measure the Success of An App.* [Online] Available at: https://gearedapp.co.uk/kpis-how-to-measure-the-success-of-an-app/ [Accessed 2020].

Gueron, S., Johnson, S. & Walker, J., 2011. SHA-512/256. In: *2011 Eighth International Conference on Information Technology: New Generations.* s.l.:s.n., pp. 354-358.

Jayasinghe, U., 2016. *Centralized vs Decentralized vs Distributed Networks.* [Online] Available at: https://www.researchgate.net/figure/Centralized-vs-Decentralized-vs-Distributed-Networks_fig1_316042146 [Accessed 2020].

jQuery, 2021. *What is jQuery?.* [Online] Available at: https://jquery.com/ [Accessed 26 4 2021].

Kaashoek, F., 2003. *Peer-to-Peer Systems II.* s.l.:s.n.

Kantaria, P., 2019. *The SAFE Network: A dark web-style solution to the online privacy problem.* [Online] Available at: https://www.verdict.co.uk/the-safe-network-a-solution-to-the-internet-privacy-problem/ [Accessed 2020].

Kristol, D. M., 2001. *HTTP Cookies: Standards, Privacy, and Politics,* s.l.: s.n.

Locke, J., 2020. *Social Media Colors 2020.* [Online] Available at: https://www.lockedownseo.com/social-media-colors/ [Accessed 2020].

Lohachab, A. & Karambir, B., 2019. Critical Analysis of DDoS—An Emerging Security Threat over IoT Networks. *Journal of Communications and Information Networks,* 3(3), pp. 57-78.

Loynes, S., 2020. *Element.* [Online] Available at: https://element.io/blog/element-1-7-8-is-out/

Lundgren, H., Gold, R., Nordstrom, E. & Wiggberg, M., 2003. A Distributed Instant Messaging Architecture based on the Pastry Peer-To-Peer Routing Substrate. In: *First Swedish National Computer Networking Workshop.* s.l.:s.n., pp. 8-10.

MaidSafe, 2020. *SafeNetwork.* [Online] Available at: https://safenetwork.tech/ [Accessed 2020].

Marr, B., 2017. *Forbes.* [Online] Available at: https://www.forbes.com/sites/jumio/2020/11/02/why-your-next-vacation-needs-identity-verification/?sh=2625a8de2b83

Matrix, 2020. *Matrix.* [Online] Available at: https://matrix.org/docs/spec/ [Accessed 2020].

Matrix, 2020. *Matrix Clients.* [Online] Available at: https://matrix.org/clients [Accessed 2020].

Meiners, N., Ulf, S. & Seeberger, B., 2010. The Renaissance of Word-of-Mouth Marketing: A 'New' Standard in Twenty-First Century Marketing Management?!. *International Journal of Economic Sciences and Applied Research,* pp. 79-97.

Nakamoto, S., 2009. *Bitcoin.* [Online] Available at: https://bitcoin.org/bitcoin.pdf

Nielson, J., 2012. *Usability 101: Introduction to Usability.* [Online] Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/ [Accessed 2020].

Nielson, J., 2016. *The Distribution of Users' Computer Skills: Worse Than You Think.* [Online] Available at: https://www.nngroup.com/articles/computer-skill-levels/ [Accessed 2020].

O'Reilly, 2021. *What Is a Decentralized Application.* [Online] Available at: https://www.oreilly.com/library/view/decentralized-applications/9781491924532/ch01.html [Accessed 4 2021].

Raj, J., 2019. *Decentralized Internet,* s.l.: s.n.

Rasti, A. H., Stutzbach, D. & Rejaie, R., 2006. *On the Long-term Evolution of the Two-Tier Gnutella Overlay,* s.l.: s.n.

Rasti, A., Stutzbach, D. & Rejaie, R., 2003. *On the Long-term Evolution of the Two-Tier Gnutella Overlay,* s.l.: s.n.

Ripeanu, M., 2001. Peer-to-Peer Architecture Case Study: Gnutella Network. In: *Proceedings first international conference on peer-to-peer computing.* s.l.:s.n., pp. 99-100.

Rittinghouse, J. & Ransome, J. F., 2005. *IM Instant Messaging Security.* s.l.:s.n.

Rosic, A., 2020. *What is Blockchain Technology? A Step-by-Step Guide For Beginners.* [Online] Available at: https://blockgeeks.com/guides/what-is-blockchain-technology/ [Accessed 2020].

Rowstron, A. & Druschel, P., 2001. *Pastry: Scalable, decentralised object location and routing for large-scale peer-to-peer systems,* s.l.: s.n.

Rowstron, A., Kermarrec, A.-M., Castro, M. & Druschel, P., 2001. *SCRIBE: The design of a large-scale event notification infrastructure,* s.l.: s.n.

Roy, R., 2008. *Shard - A Database Design.* [Online] Available at: http://technoroy.blogspot.com/2008/07/shard-database-design.html [Accessed 2020].

Saunders, B., 2020. *Who's Using Amazon Web Services? [2020 Update].* [Online] Available at: https://www.contino.io/insights/whos-using-aws [Accessed 5 2021].

Sihi, D., 2020. Impacts of Blockchain Technology. In: *Advances in Digital Marketing and eCommerce.* s.l.:s.n., pp. 35-40.

Solidity, 2020. *Solidity.* [Online] Available at: https://docs.soliditylang.org/en/v0.7.5/ [Accessed 2020].

Stanford University, 2002. *How does the Internet work?.* [Online] Available at: https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm [Accessed 2020].

Statista, 2017. *Number of mobile messages sent through WhatsApp as of 4th quarter 2017.* [Online] Available at: https://www.statista.com/statistics/258743/daily-mobile-message-volume-of-whatsapp-messenger/ [Accessed 4 2021].

Statista, 2020. *Facebook Messenger - Statistics & Facts.* [Online] Available at: https://www.statista.com/topics/4625/facebook-messenger/ [Accessed 2020].

Statista, 2020. *Most popular global mobile messenger apps as of October 2020, based on number of monthly active users.* [Online] Available at: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/ [Accessed 2020].

The Economist, 2017. *The world's most valuable resource is no longer oil, but data.* [Online] Available at: https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data [Accessed 5 2021].

ThinkComputers, 2013. *History of The Hard Drive.* [Online] Available at: https://thinkcomputers.org/the-history-of-the-hard-drive/#:~:text=In%201991%2C%20IBM%20released%20its,of%20data%20on%20eight%20disks.&text=Hard%20drives%20continued%20to%20become,inch%20hard%20drive%20in%20

1992.
[Accessed 2020].

Tilkov, S. & Vinoski, S., 2010. Node.js: Using JavaScript to Build High-Performance Network Programs. *IEEE Internet Computing,* 14(6).

Viganotti, G., 2020. *The Safe API.* [Online] Available at: https://github.com/maidsafe/sn_api/blob/master/README.md#description [Accessed 2020].

Waters-Lynch, J., 2018. *Why we need a new internet.* [Online] Available at: https://medium.com/typehuman/why-we-need-a-new-internet-713fb980a151 [Accessed 4 2021].

WhatsApp, 2020. *WhatsApp Encryption Overview,* s.l.: s.n.

Wills, N., 2015. *LWN.* [Online] Available at: https://lwn.net/Articles/632572/ [Accessed 2020].

Yu, C.-E., Xie, S. Y. & Wen, J., 2020. Coloring the destination: The role of color psychology on Instagram. *Tourism Management,* pp. 1-12.

9 Appendices

Appendix A – GitHub Repository

The source code for the project was hosted on GitHub at the following link:

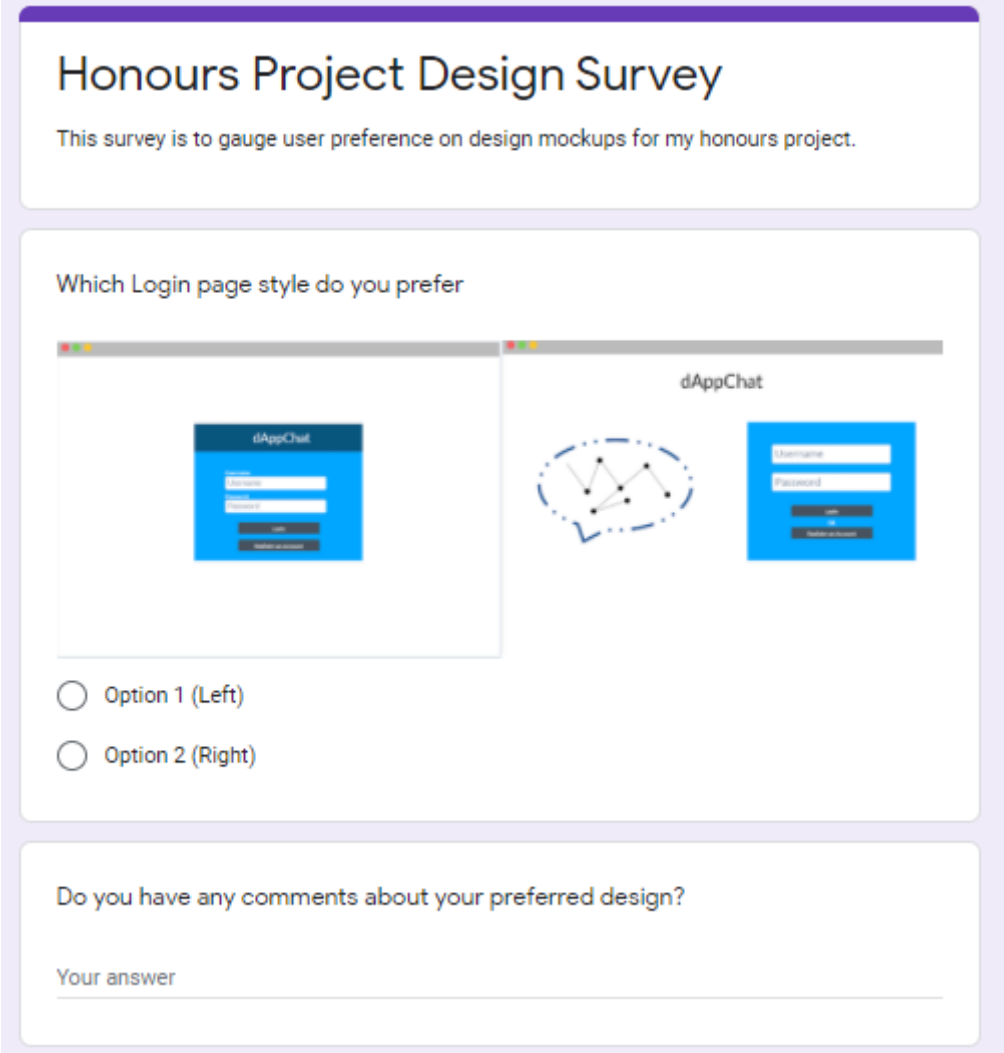
https://github.com/CraigCargill/HonoursProject

Appendix B – Heroku URL

https://fast-scrubland-37543.herokuapp.com/

Appendix C – Design/Implementation Survey Questions

For the design, users were asked for feedback on preference for the design mockups. This survey can be accessed at https://forms.gle/kQk4GWrYZzCHbNweA



Which Chat page style do you prefer

- ◯ Option 1 (Left)
- ◯ Option 2 (Right)

Do you have any comments about your preferred design?

Your answer

Which Add Room page style do you prefer

- ◯ Option 1 (Left)
- ◯ Option 2 (Right)

Do you have any comments about your preferred design?

Your answer

Which colour scheme do you prefer

- ◯ 1
- ◯ 2
- ◯ 3
- ◯ 4
- ◯ 5
- ◯ 6
- ◯ 7
- ◯ 8
- ◯ 9
- ◯ 10

Do you have any comments about your preferred colour scheme?

Your answer

Submit

Creating a messaging client using a decentralised network in order to reduce or replace requirement for servers.

Student: Craig Cargill Supervisor: Dr John Isaacs

Introduction

With the world collecting more data than ever, privacy and security are becoming more and more important to everyday consumers.

A decentralised network aims to solve two main problems with the current implementation of the internet, anti-competitiveness and privacy. When the internet first became a large scale tool, it was built on the technology at the time. Everyday users did not have powerful computers, and the average storage size of a hard drive in a personal computer in 1991 was around 1 gigabyte. (ThinkComputers, 2013) This meant the internet needed to be spread across servers designed to hold a large amount of data. This began the architecture of our centralised network and paved the way for companies to build large storage systems and own the process we use to store and share data on the internet.

Methods

The website was created using VisualStudio Code. The messages are stored on the Ethereum blockchain in order to ensure decentralisation. The Aleph.im API was used to interact with this blockchain. The API provided a framework for hosting messages and creating Ethereum accounts on the fly for users. Compression was handled with lz-string compression before the message is sent and decompressed on receipt. This aids in reduced strain on the network. The design of the app was based around usability for all levels of technical experience. This involved a survey that collected metrics of style and layout.

Project Aim

This project aims to build a chat app that allows two or more users to communicate with each other over a decentralised network. The app will not use a traditional client server model and instead will look into methods to transfer data P2P or using a blockchain. With the goal for a higher level of security and ownership of data.

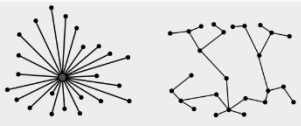


Fig 1: Centralised vs Decentralised network architectures

Figures and Results

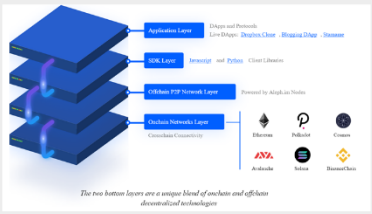


Fig 2: Aleph.im stack structure

Figure 2 shows the Aleph.im stack, this stack shows the Application layer which the messaging app created for this project will live on. And below to the SDK layer the app pulls from. The diagram also shows the on and off chain layers allowing for the application to communicate to traditional clients as well as decentralised.

To create the application layout, a process involving user feedback throughout the design was implemented. To do this, user preferences were gathered from surveys

The first preference for users was whether or not the messaging client should be a desktop app or website. This question was put to users in a survey, the preferred choice between these two would then be taken forward as the final design.

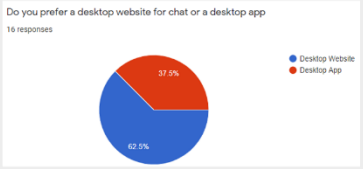


Fig 3: Survey results for client preference

Again, users were asked between different colour palettes that would be used throughout the website. Figure three shows that users were drawn towards the first palette comprised of predominantly blue colours.

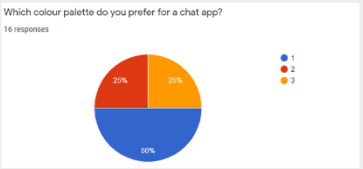


Fig 4: Survey results for colour palette preference

The same methodology was used across other aspects of the website such as logo design and layout.

Based on the feedback from these surveys, the website was build in a beta form shown in figure 5. This is a culmination of preferences shown by technical and non-technical users.

Fig 5: Beta design of website interface

Conclusion

In conclusion, this website aimed to create a working interface with a decentralised network that allowed users to communicate with each other. With surveys completed to ensure that design and platform were suited to all users, the app also aimed to open the gates of decentralised networks to standard users. To this end, users were selected to test the application during and after development in order to ensure integrity of these values.
Next, more testing will be completed as the stability of the network can only be verified with continual use. Surveys will be created to gain user feedback, propose new features, and overall gauge the success of the project.

Acknowledgments

I would like to thank Dr John Isaacs for providing support and guidance throughout this year. For striving to provide the same level of support and availability especially through the pandemic and working from home. I would also like to thank my partner Tricia Wagg for supporting me and providing motivation and care throughout the entire year. Finally I would like to thank my family, for helping me not only throughout this year, but my entire academic and non-academic life.

References

ThinkComputers, 2013. History of The Hard Drive.

ROBERT GORDON UNIVERSITY ABERDEEN
SCHOOL OF COMPUTING SCIENCE AND DIGITAL MEDIA

BSc (Hons) Computer Science

Detailed Project Proposal

First Name:	Craig
Last Name:	Cargill
Student Number:	1604113
Supervisor:	Dr John Isaacs

Defining your Project

1.1 Project title

Help: a brief statement about what you are actually going to do.

Creating a messaging client using a decentralised network in order to reduce or replace requirement for servers.

1.2 Background

Help: Provide the background to your project. This section should highlight the main topics in the area you are going to research. Essentially what is the project about, what has been done before and why is this project important? ~500 words

This research will investigate different implementations of decentralised systems. This will include other chat apps but also larger-scale network systems and file storage systems. Currently, on the Apple App Store and Google Play store, there are multiple apps using a decentralised network however these apps are not mainstream and not clear of their advantages to the general public.

The research will be focused on two main segments. The first area will be into decentralised networks that currently exist and the architecture they use. The next is messaging clients on these networks, the features they offer, the limitations of their platform, and their adoption.

Decentralised messaging clients exist on public devices however they generally have low Daily Active Users, which for a decentralised network is not just a sign of poor adoption but has the potential to kill the platform. Decentralisation faces the paradox of needing new users to host the client/server nodes on their devices. However, they cannot function until reaching a critical mass of users to securely distribute the data. Current attempts at a decentralised network use a steppingstone of decentralised servers acting as server nodes which allow users to rely on the servers until the adoption is high enough to remove the servers.

A focus of my messaging client would be its usability and understanding of the general public. Most current solutions are complicated, clunky, or broken. The aim of this project would be to make a well-designed but also capable solution usable by both everyday users and technically minded people.

Decentralised networks have been an emerging technology in the industry for many years now. The main goal of a decentralised network would be for it to be adopted by all demographics of people. Currently, the technology is limited to those in the "tech sphere". This will cover a larger area of younger users aged 20-25 and industry veterans of ages 25+.

A decentralised messaging client is a concept that is not easily conveyed to non-technical persons. Based on preliminary research, general public adoption of the technology is a leading concern to the viability and growth of the network. Another concern of the technology is the current limitation of lossless compression. Distributing fragmented files across a decentralised network has the inherent issue of data redundancy. Current research shows that decentralised networks need better compression to be a more viable solution.

This technology is important as it is the logical way to take back control of the internet and truly put the data of the people back in the hands of the people. In theory, anyone can develop for a decentralised network, but the benefit is that no one organisation would have control of any platform. Currently, large corporations have complete control over data stored on their servers and by extension have control over their users. Allowing for censoring, oppression, and bias. A decentralised solution would remove the ability for a server-side filter. Users would be welcome to filter on the client side but that would be a conscious choice. Unlike the filtering currently happening.

1.3 Motivation

Help: To whom is this project important? A project must address a question/problem that generates a small piece of new knowledge/solution. This new knowledge/solution must be important to a named group or to a specific client (such as a company, an academic audience, policy makers, people with disabilities) to make it worthwhile carrying out. This is the ***motivation*** for your project. In this section you should address who will benefit from your findings and how they will benefit. ~300 words

> **Example** 1: If you intend to demonstrate that a mobile application that automates class registers at RGU will be more efficient than paper-based registers - the group who would be interested in knowing/applying these findings would be both academic and administrative staff at RGU and they would benefit by time saved and a reduction in their administrative workload.
>
> **Example** 2: You are demonstrating that a particular 3D model design increases realism in 3D environments. The group that would be interested would be games designers or developers of 3D virtual environment applications. The would benefit from producing more realistic environments that could increase sales of their products.
>
> **Example** 3: You have designed a new network topology for IrishOil plc's new Aberdeen headquarters. The interested group would clearly be IrishOil. They would benefit from easier maintenance and improved security of their computer network.

The planned messaging service would largely be aimed at the general public. Most people have a need for communication in both private and business life. The younger or technical demographic more likely to adopt the technology however the goal would be mass adoption of the network.
Currently there are multiple apps available based on decentralised networks, but many are badly designed and difficult to understand. They are also often unstable. Current research indicates that without a suitable incentive, decentralised networks are less likely to be adopted as shifting society to an alternative from what already works is not simple.
In order to provide an incentive, the network needs to be built out and contain the same or similar features as the current internet. There is a known desire for an alternative to current tech solutions such as Facebook, Twitter, Amazon etc. By growing a decentralised network, the general public may be more inclined to try this alternative.
From keeping up to date with tech news, reports, lawsuits. It has become much clearer that an alternative to current tech giants is required. Without changing the fundamental building blocks of the internet, even if a current tech giant loses their power, another will take its place. This shows the need for a network without the possibility of this power being used. By companies, governments, or users.

1.4 Aim & Objectives

Help: Outline what are the main things your project is going to do and what steps or milestones will be used to achieve this aim. The Aim is unlikely to change throughout your project; however, the objectives are likely to adapt to your ongoing research and development. In particular it is highly likely that you may wish to split objectives into sub-objectives as work progresses. A good clear set of objectives give you something to evaluate your final project against.

> **Example** : For the timetable app outlined above
> Aim: To create a functioning attendance application that efficiently automates the taking of class registers.
> Objective 1: study existing register system in place at RGU and identify weaknesses
> Objective 2: research existing automation technology's and identify and evaluate those that may be appropriate to taking in class registers
> Objective 3: Implement chosen technologies to create prototype application
> Objective 4: Conduct user trials to evaluate capabilities of prototype application
> Objective 5: Create a refined application incorporating feedback from user trials

Aim: To create a messaging client that uses a decentralised network instead of a traditional server client network., allowing users to message other users of the network.
Objective 1: Research existing decentralised networks and applications running on that network.
Objective 2: Study current decentralised applications and identify which of those implement features suitable for a messaging client.
Objective 3: Create a prototype of the client using proposed technologies.
Objective 4: Run user trials to gain feedback on features and usability.
Objective 5: Polish the application and implement features or fixes uncovered in user trials.
Objective 6: Create a prototype decentralised network running on servers to imitate clients
Objective 7: Ensure messaging client works on new network

1.5 Key Techniques

Help: Perform some initial research into the area and outline what techniques you my research in further detail here. The techniques you cover here should include references to the papers where you have sourced the information. The techniques mentioned here are very likely to become the section headers in your literature review.

The main techniques used in the research will be based around the development and deployment of a decentralised app and then subsequently the same for the network itself. In order to understand the requirements for this, relevant research papers and websites will be used to develop the network and application.
Decentralised P2P Network architecture
http://www.berkes.ca/archive/berkes_gnutella_freenet.pdf
Decentralised Network Coding
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1405320
Decentralised Messaging App
https://medium.com/adamant-im/how-decentralized-blockchain-messenger-works-b9932834a639

Matrix.org Guides and Documents
https://matrix.org/docs/guides/
Decentralised Networks: The Future Internet
https://link.springer.com/article/10.1007/s11036-018-01211-5

1.6 Legal, Social, Ethical, Professional and Security issues

Help: Here you should discuss any legal, social, profession and security issues that you believe may occur during the course of your project. It is not acceptable to write none in this box, all projects, regardless of focus will have to address issues in one, or more, of these categories. This is an extremely important part of your honours project to which there is no correct answer, this section must be fully discussed with your Honours Supervisor.

Example 1 : In the class register example above – there would be a Legal and Security issue with the gathering and storage of student data. There may be a social constraint as you may be relying on a user to have access to a specific technology. There will need to be consideration of user accessibility.

Example 2 : A 3D model design may have ethical considerations in its evaluation. What if your model made users feel nauseous. Social constrains may again be access to technology or accessibility issues.

Example 3 : You network design need to adhere to specific company policies. You would need to consider the possibility that your design could be wrong, compromising the company's security.

No user data will be stored, each user will be auto-generated a username however any messages or photos sent on the messaging client will be stored.

1.7 Project Plan

Help: This is the project plan as to how you will go about achieving the objectives of the project.

Example: In the class register example above the research plan may involve:
Collecting and analysing paper-based registers in a given class on five occasions.
Identifying the error rate average on these occasions
Researching existing automation techniques
Designing and implementing a mobile application that automatically records attendance in class.
Deploying the application in the class on five occasions.
Identifying the error rate average of the mobile application on these occasions.
Comparison of data and summary of findings.

Look into current decentralised messaging clients vs traditional messaging clients.
Identify improvements from current clients to implement
Design a messaging client to work on a current decentralised network

Research current decentralised networks and their architecture Identify base features and implement them as needed Deploy messaging client on custom network Conduct user testing and polish based on feedback

STUDENT PROJECT ETHICAL REVIEW (SPER) FORM

The aim of the University's *Research Ethics Policy* is to establish and promote good ethical practice in the conduct of academic research. The questionnaire is intended to enable researchers to undertake an initial self-assessment of ethical issues in their research. Ethical conduct is not primarily a matter of following fixed rules; it depends on researchers developing a considered, flexible and thoughtful practice.

The questionnaire aims to engage researchers discursively with the ethical dimensions of their work and potential ethical issues, and the main focus of any subsequent review is not to 'approve' or 'disapprove' of a project but to make sure that this process has taken place.

The *Research Ethics Policy* is available at www.intranet.rgu.ac.uk/credo/staff/page.cfm?pge=7060

Student Name	Craig Cargill
Supervisor	Dr John Isaacs
Project Title	Creating a messaging client using a decentralised network in order to reduce or replace requirement for servers.
Course of Study	BSc Computing Science
School/Department	School of Computing

Part 1 : Descriptive Questions			
1	Does the research involve, or does information in the research relate to:	Yes	No
	(a) individual human subjects		X
	(b) groups (e.g. families, communities, crowds)		X
	(c) organisations		X
	(d) animals?		X
	Please provide further details:		
2	Will the research deal with information which is private or confidential?	Yes	No
			X
	Please provide further details:		
	Users in messaging app will be randomly given a username and the choice of a password. No user information will be stored.		

Part 2: The Impact of the Research			
3	In the process of doing the research, is there any potential for harm to be done to, or costs to be imposed on	Yes	No
	(a) research participants?		X
	(b) research subjects?		X
	(c) you, as the researcher?		X
	(d) third parties?		X
	Please state what you believe are the implications of the research:		
4	When the research is complete, could negative consequences follow:	Yes	No
	(a) for research subjects		X
	(b) or elsewhere?		X
	Please state what you believe are the consequences of the research:		

Part 3: Ethical Procedures			
5	Does the research require informed consent or approval from:	Yes	No
	(a) research participants?	X	
	(b) research subjects		X
	(c) external bodies		X
	If you answered yes to any of the above, please explain your answer: Initial research will involve verbal questions with volunteers. No consent will be taken as volunteers will not have any data taken about them. Later in the research volunteers will be asked for consent to use the messaging client. The users will also give permissions for camera use in the application in order to send messages.		
6	Are there reasons why research subjects may need safeguards or protection?	Yes	No
			X
	If you answered yes to the above, please state the reasons and indicate the measures to be		
7	Has PVG membership status been considered?		
	(a) PVG membership is not required.		X
	(b) PVG membership is required for working with children.		
	(c) PVG membership is required for working with protected adults.		
	(d) PVG membership is required for working with both children		

	and protected		
	If you answered yes to (b), (c) or (d) above, please give details:		
8	Are specified procedures or safeguards required for recording, management, or storage of data?	Yes	No
			X
	If you answered yes to the above, please outline the likely undertakings:		

Part 4: The Research Relationship			
9	Does the research require you to give or make undertakings to research participants or subjects about the use of data?	Yes	No
			X
	If you answered yes to the above, please outline the likely undertakings:		
10	Is the research likely to be affected by the relationship with a sponsor, funder or employer?	Yes	No
			X
	If you answered yes to the above, please identify how the research may be affected:		

Part 5: Other Issues			
11	Are there any other ethical issues not covered by this form which you believe you should raise?	Yes	No
			X
	Encrypted messaging does provide an inherent risk of being used for nefarious means however the application itself is not inherently unethical.		

Statement by Student			
I believe that the information I have given in this form is correct, and that I have addressed the ethical issues as fully as possible at this stage.			
Signature	Craig Cargill	Date	16/10/2020

If any ethical issues arise during the course of the research, students should complete a further Student Project Ethical Review (SPER) form.

The *Research Ethics Policy* is available at

www.intranet.rgu.ac.uk/credo/staff/page.cfm?pge=7060

Part 6: To be completed by the supervisor			
12	Does the research have potentially negative implications for the University?	Yes	No
	If you answered yes to the above, please explain your answer:		
13	Are any potential conflicts of interest likely to arise in the course of the research?	Yes	No
	If you answered yes to the above, please identify the potential conflicts:		
14	Are you satisfied that the student has engaged adequately with the ethical implications of the work? [In signifying agreement, supervisors are accepting part of the ethical responsibility for the project]	Yes	No
	If you answered no to the above, please identify the potential issues:		
15	**Appraisal:** Please select one of the following		
	The research project should proceed in its present form – no further action is required		
	The research project requires ethical approval by the School Ethics Review Panel		
	The research project needs to be returned to the student for modification prior to further action		
	The research project requires ethical review by an external body. If this applies please give details		
	Title of External Body providing ethical review		
	Address of External Body		
	Anticipated date when External Body may consider project		

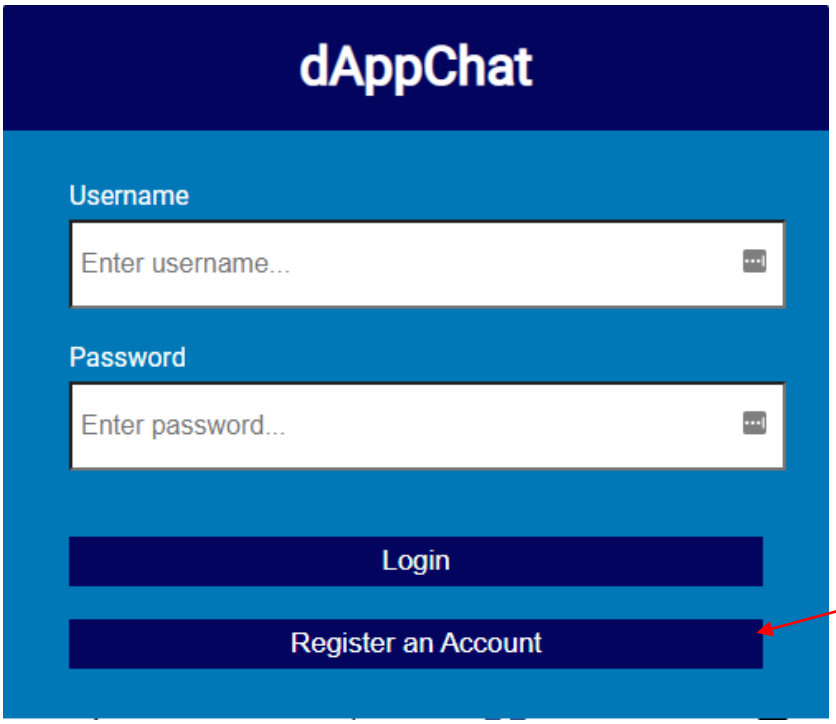
Affirmation by Supervisor			
I have read the student's responses and have discussed ethical issues arising with the student. I can confirm that, to the best of my understanding, the information presented by the student is correct and appropriate to allow an informed judgement on whether further ethical approval is required.			
Signature		**Date**	

Appendix G – User Guide

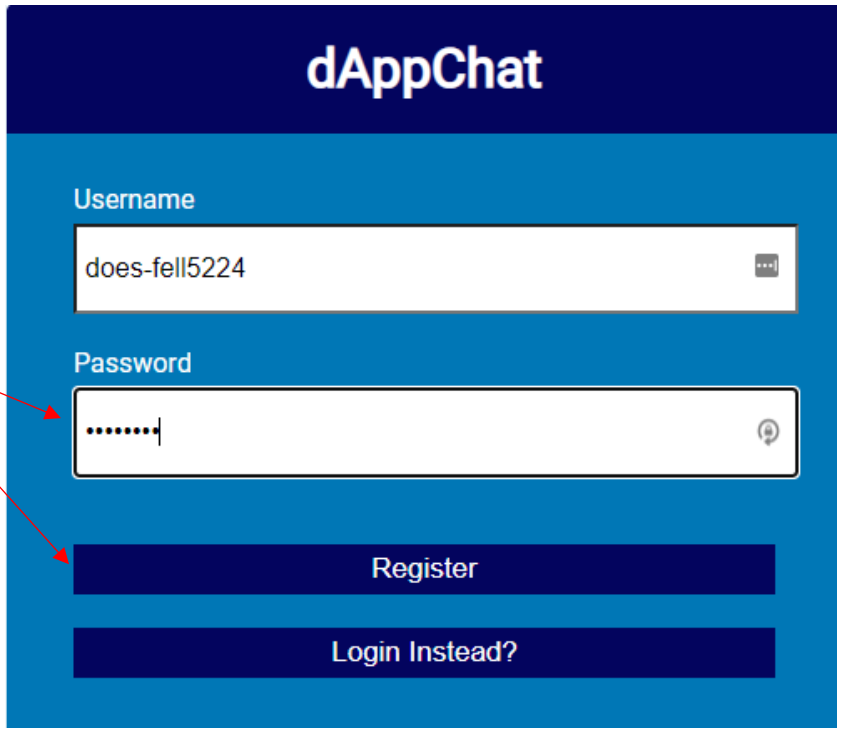
User Guide for dAppChat

This guide outlines some of the key tasks a user may have to complete in order to use the website to its full potential. Each of the tasks is outlined in the header and graphics are used to show the optimal route.

Register Account and Login

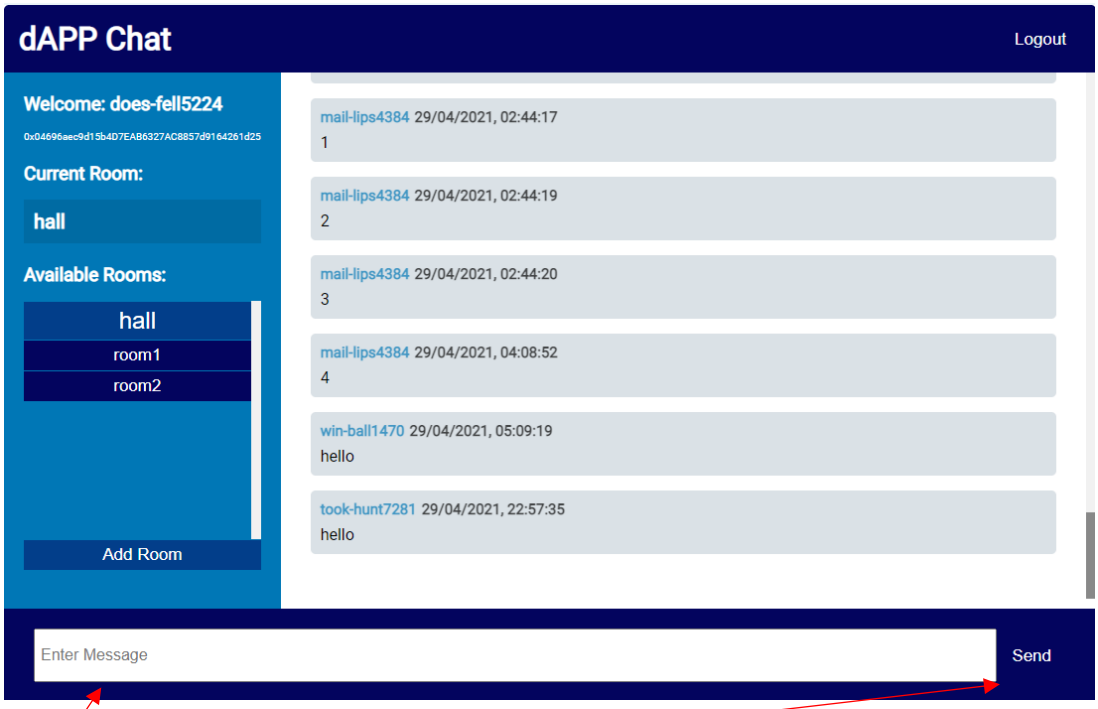


Click Register an Account



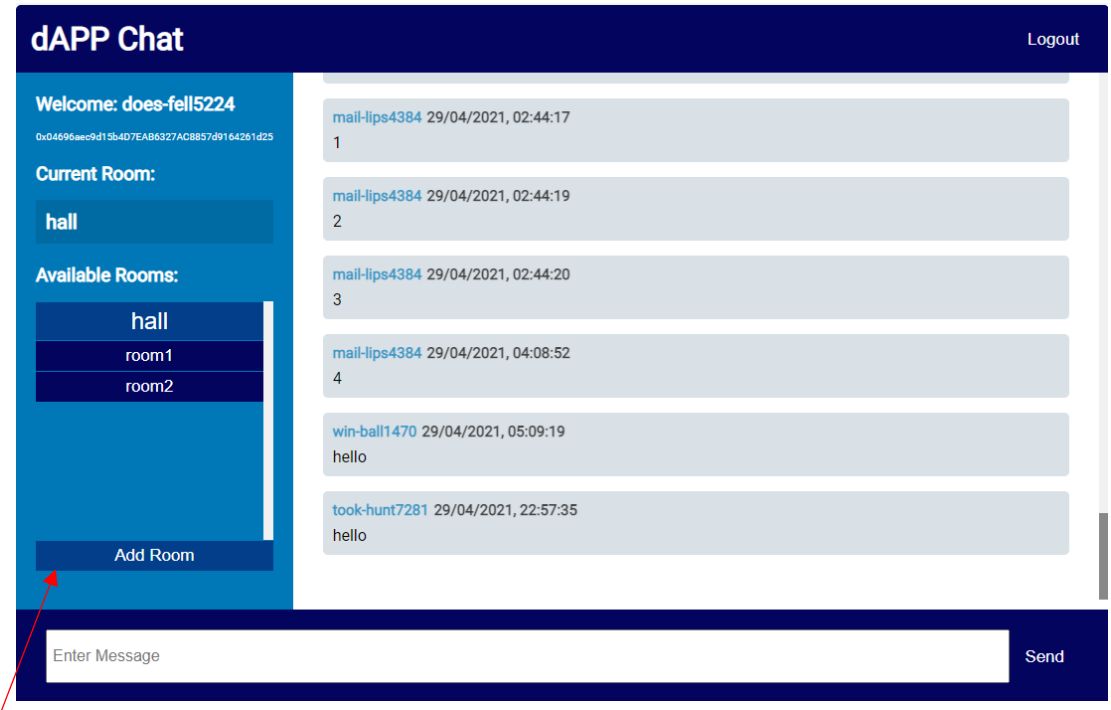
Fill out a password you will remember

Send Message

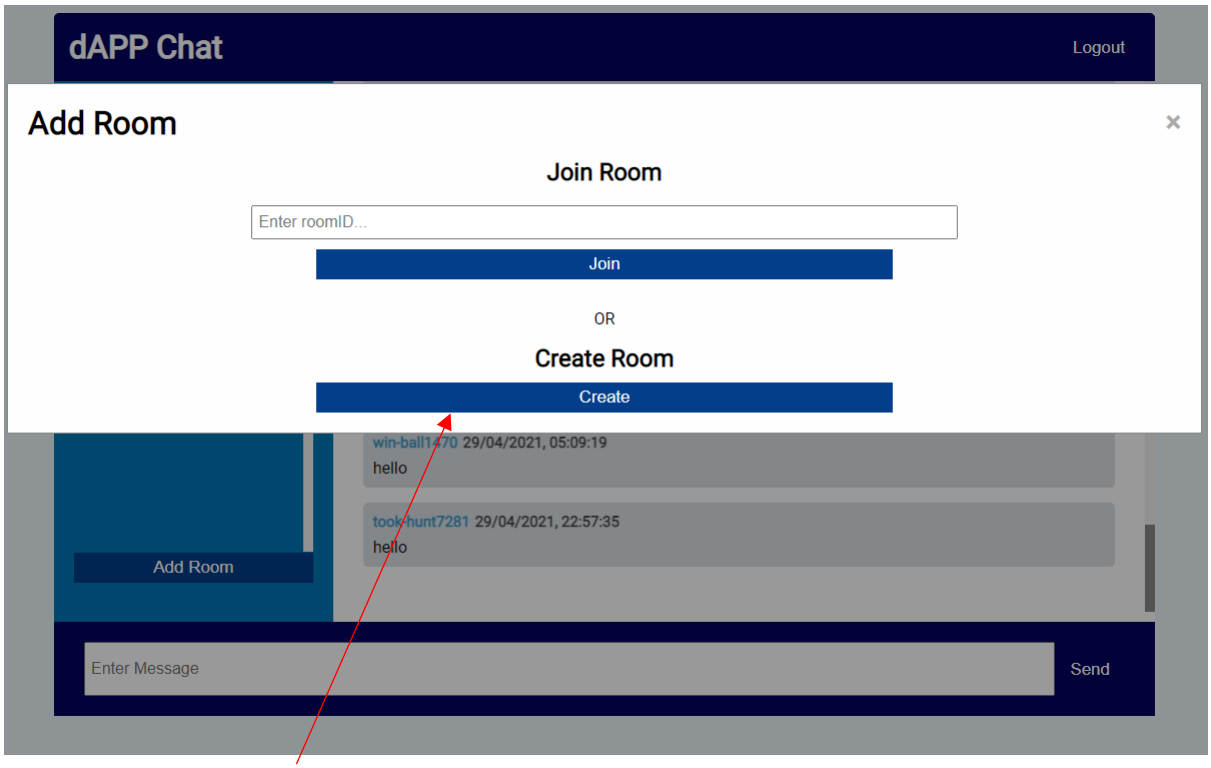


Type message and click send

Create a Room

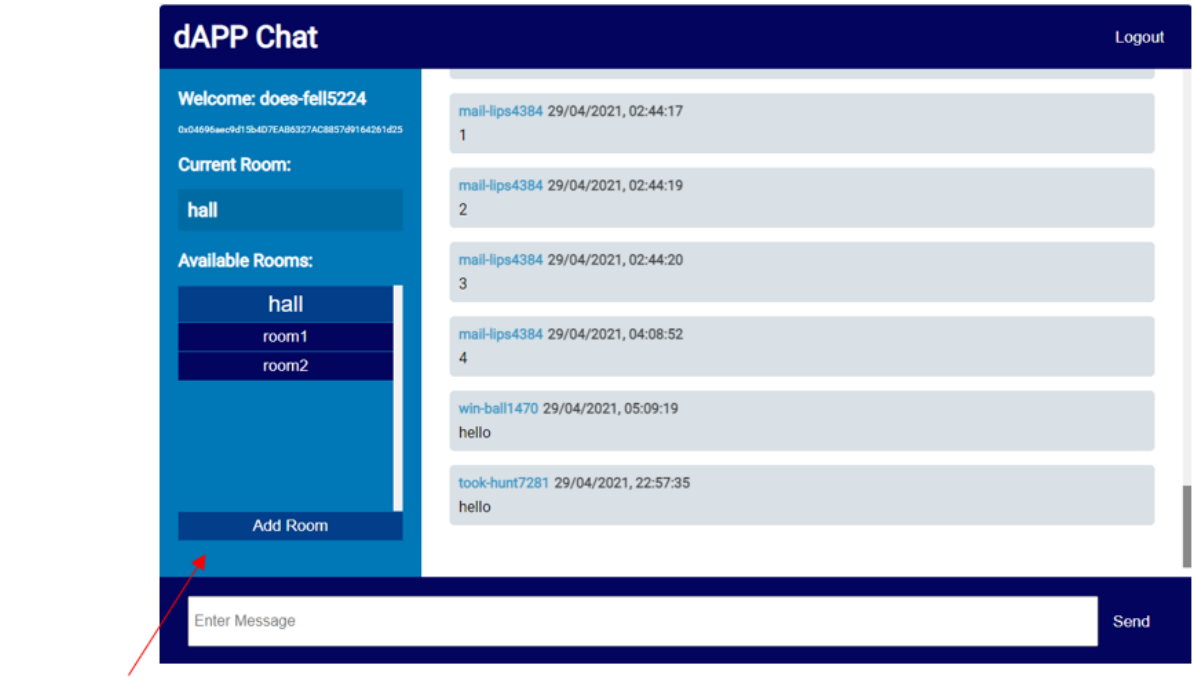


Click “Add Room”

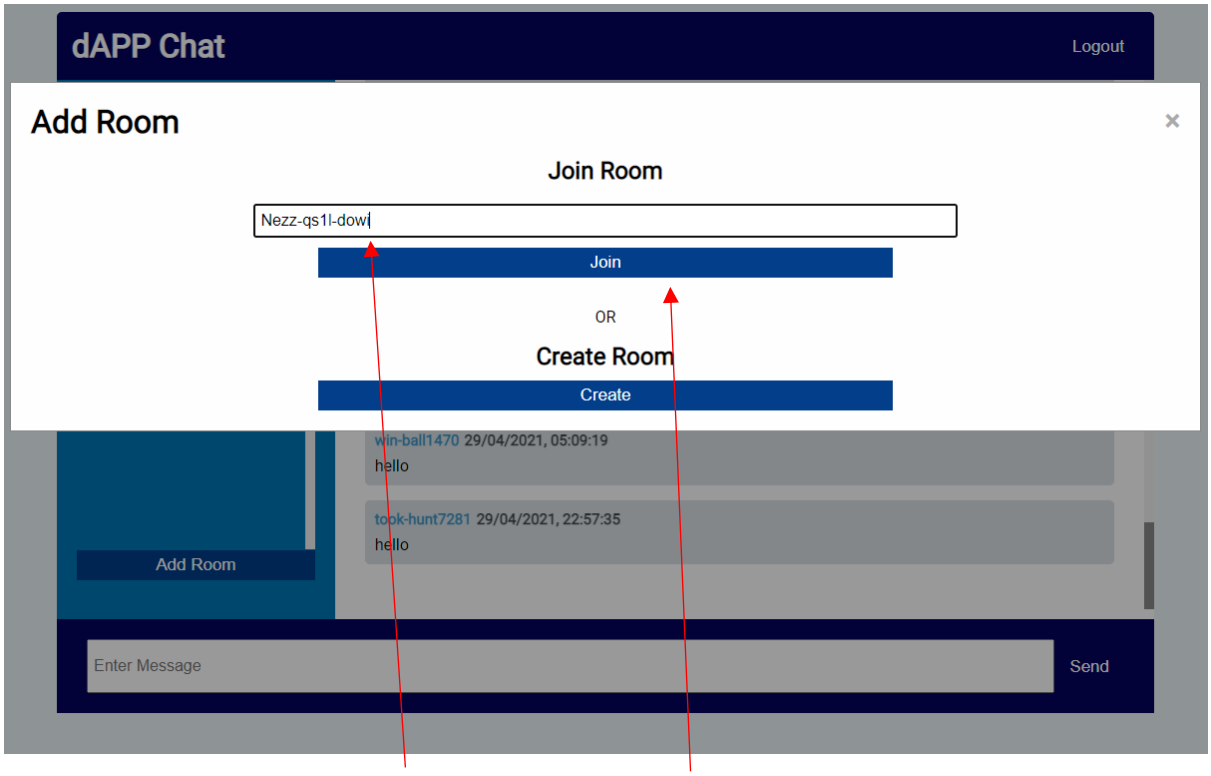


Click “Create” to create and join a new room

Join a Room

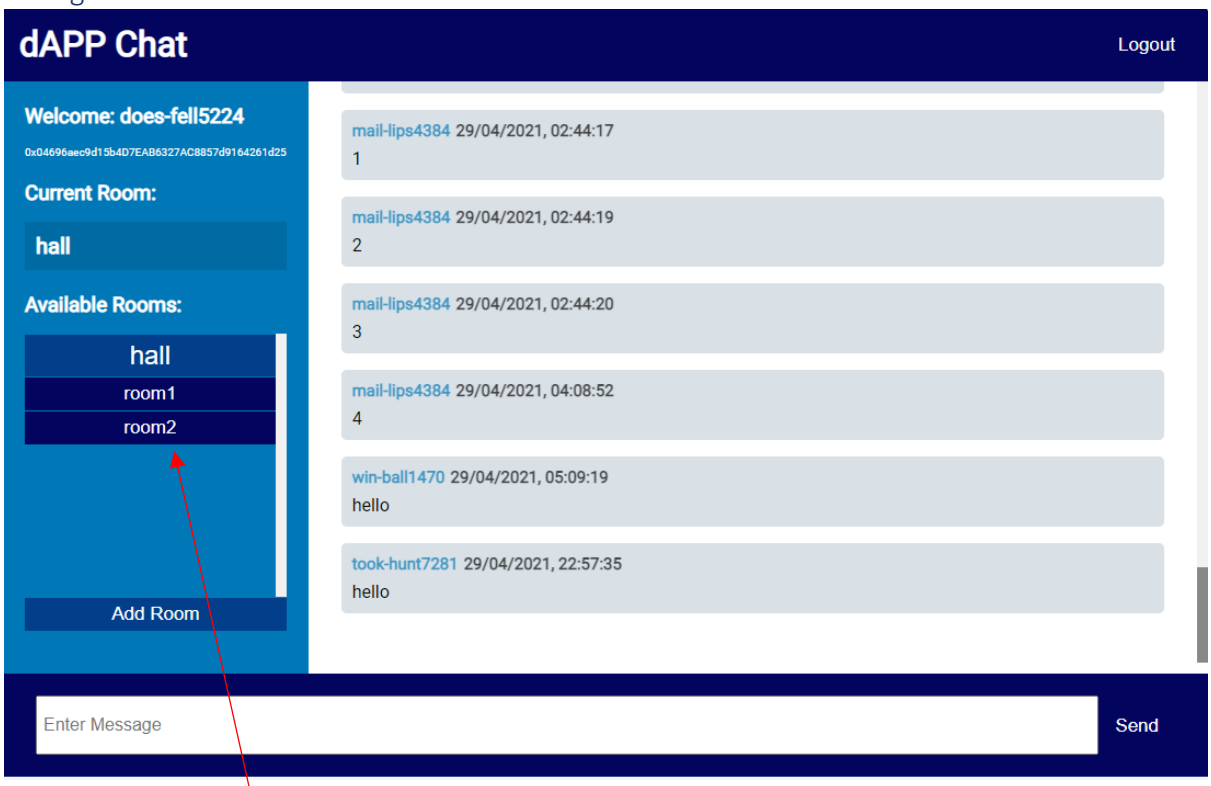


Click “Add Room”



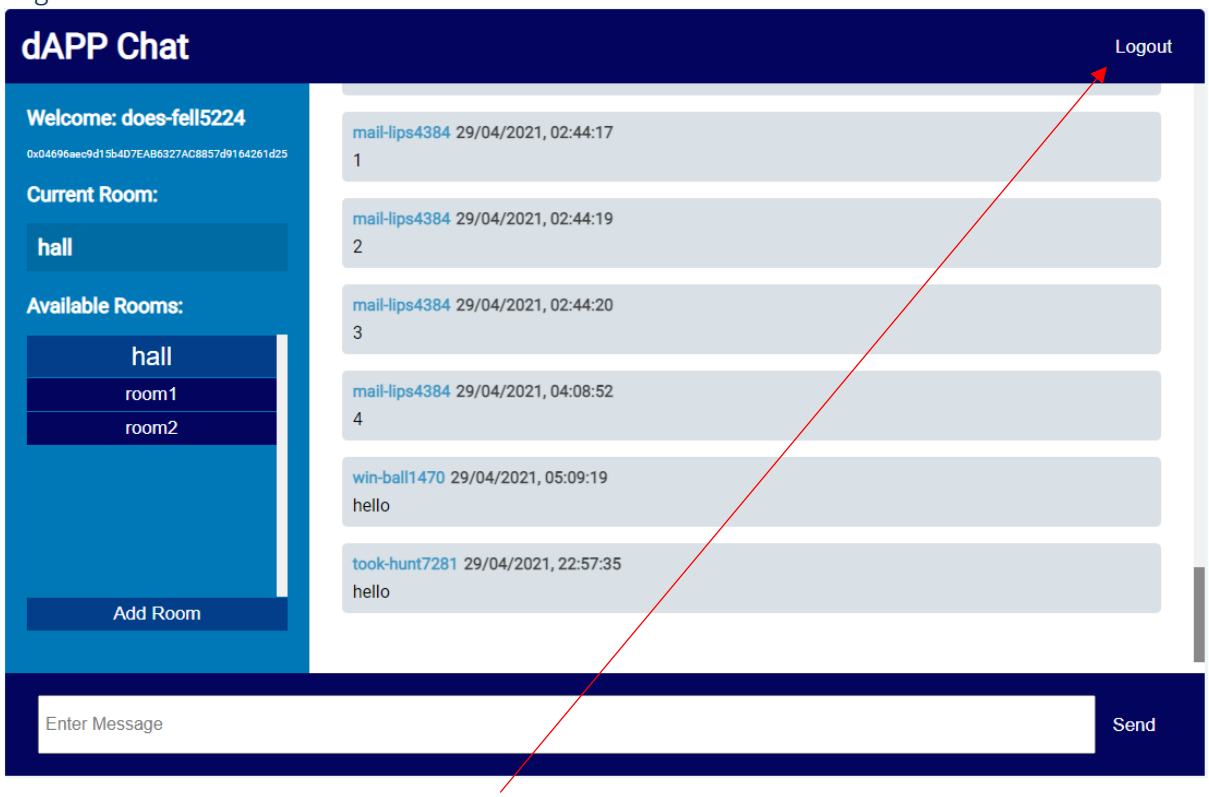
Type the Room ID into the Join room box and click “Join”

Change Rooms



Click which room you want to change to using the room list

Log Out



Click the Log Out button in the top right

10 Figures