Goldsmiths Meet Ups (GMU)

IS52018C/IS52018S/IS52018F: Software Projects / Computing Project 2 Report Part Two (2022-23)

MEET UPS/GET TO KNOW EACH OTHER APPLICATION

By Amin Dayri, Ionut Boboi, Ruben Nhez, Simran Johal, Tomi Akinyemi

1 – INTRODUCTION

The Problem and Our Solution

Our project seeks to address the challenge that many Goldsmiths' students encounter when trying to find like-minded people to connect with. It is not uncommon for students to feel overwhelmed when trying to navigate the vast array of social groups and activities available to them, and as a result, many find it challenging to connect with individuals who share their interests and hobbies; this was found out in our initial market survey [Appendix].

To bridge this gap, our team is working on an innovative application that allows users to input their preferences and interests, such as their hobbies, ethnicity, and languages. We know our application will be innovative and unique because this is what most stakeholders thought when they participated in our initial market survey [Appendix]. The system would then use this information to suggest group chats or individuals who share similar interests, providing a much-needed boost to the students' social lives. We say our app is much-needed due to the fact that 100% of the stakeholders that took our initial survey said this is an app they very much would use at Goldsmiths University.

We believe that our application will fill a gap in the market for a messaging platform that provides tailored recommendations based on individual interests. The system will have a feature that automatically analyse users' inputs and match them with groups or individuals that have similar interests, saving them time and effort.

One of the key features of our application is that it is designed to be user-friendly and intuitive. Upon signing up, users will create their own personal account, complete with an email and password and we even added a 2FA feature, to log in to the system. They will then be prompted to complete their profile page, where they can add information about their interests and preferences. By having a personal account, users will be able to search for other individuals based on similar interests, as well as be searched for by others, providing a seamless experience for all.

The goal of this project is to develop a social networking app that helps Goldsmiths' students connect with individuals who share similar interests and hobbies. The app will be designed to provide tailored recommendations based on individual preferences, allowing users to find like-minded people easily.

The Project Scope

We have created a list of components that will be a part of our focus to give to our stakeholders. These key elements will provide the users an understanding of what our project will hopefully include by the end of its development cycle.

The minimum viable product (MVP) for this project will include the following features:

1. User authentication: Users will be required to create a personal account with their email and password to access the app.
2. Two-factor authentication: Users will be required to sign in with a 2FA method.
3. Profile creation: Users will be prompted to complete their profile page, where they can input information about their interests, hobbies, ethnicity, and languages.
4. Recommendation engine: The app will use the information provided by users to suggest group chats or individuals who share similar interests and preferences.
5. Messaging system: The app will include a messaging system that allows users to communicate with each other directly.
6. Search functionality: Users will be able to search for individuals based on their interests and preferences.

The MVP will focus on providing a seamless user experience that allows students to connect with like-minded individuals quickly and easily. The app's design will be clean and user-friendly, with a straightforward layout that enables users to navigate the app without difficulty.

Future iterations of the app could include additional features such as a calendar function for organizing social events and an algorithm that suggests new groups based on user behaviour. However, these features will be outside the scope of the MVP and will be considered for implementation in later phases of the project.

2 – PLANNING

Our Starting Plan

We as a team of coders, started by holding a series of meetings to define the project's scope and objectives, as well as the specific requirements it must meet. During these meetings, we worked closely with the stakeholders and end-users, to identify the key features and functions that the software must include. This involves gathering detailed information about the user requirements, the technical constraints, and the desired outcomes of our project. Our meetings were decided to be weekly, or whenever we were in dire need of help from each other.

Once we had defined the scope and requirements of the project, we decide on an appropriate development methodology, considering factors such as the complexity of the project, the level of risk, the size of the team, and the level of involvement of the stakeholders.

We then create a detailed project plan that outlines the timeline, milestones, and deliverables for each phase of the project. This plan is regularly reviewed and updated throughout the development process to ensure that the project stays on track and that any issues or risks are addressed in a timely manner.

As user experience was a key factor for our project, we involved stakeholders in regular feedback and testing cycles to gather feedback and ensure that the software meets the user's needs. This may involve running beta tests, conducting user surveys, or engaging in user experience research to gather insights into how the software is being used and how it could be improved.

Our Stakeholders were defined early on during the project proposal: Goldsmiths' students. However, we wanted a more personal audience, so we decided to ask people we already knew at Goldsmiths to also test our project. This allowed for a more distinctive and trustful reaction, whilst still following the guidelines of our target audience.

The Development Techniques

Now that our requirements and objectives were established, it was time to further agree upon as a team of the development methodology we were going to work on. We had decided in our project proposal that we would follow the Agile Methodology. This was because this methodology had key characteristics that would help us create our project.

Below are the main reasons for the choosing of Agile development:

- Stakeholder involvement: Agile development places a strong emphasis on involving the stakeholder in the development process. This means that stakeholder feedback is

incorporated throughout the development cycle to ensure that the product meets their needs and expectations.

- Iterative development: Agile development is an iterative process that involves multiple cycles of development, testing, and feedback. This allows us (the development team) to quickly adapt to changing requirements and deliver a working product in short timeframes.
- Continuous delivery: Agile development also emphasizes continuous delivery, which means that the development team delivers a working product in small increments, rather than waiting until the end of the development cycle to release the entire product.
- Cross-functional teams: Agile development requires cross-functional teams that include individuals from different disciplines, such as developers, testers, and designers (which is some of the roles we split between us). This allows for more effective collaboration and communication between us as team members.
- Adaptive planning: Agile development places a greater emphasis on adaptive planning rather than following a strict plan. This means that we continuously re-evaluated the project plan and adjusted it based on changing requirements, feedback, and priorities.
- Emphasis on quality: Agile development also emphasizes the importance of producing high-quality software. This is achieved through continuous testing, collaboration, and feedback throughout the development process.
- Flexibility: Agile development is flexible and can be adapted to different project requirements and team structures. It also allows for changes to be made to the project plan at any stage of the development process.

Overall, agile development is a stakeholder-focused and iterative approach to software development that emphasizes collaboration, flexibility, and quality. By following these key aspects of agile development, we can deliver a product that meets the needs of the stakeholder and respond to changing requirements in an effective and efficient manner.

Team Planning and Roles

During the project proposal making stage, we found out that our team was compromised with many skills all in different areas. This was a positive finding as this meant we could each be set a role through our skills that could aid the process of creating our project.

The roles and skills of each team member are summarised below:

Report Handler - Amin Dayri: Has been involved in a project where they assumed responsibility for formatting the project document and managing the final report. Their duties involved organizing and presenting project data, ensuring consistency in the document's style and format, and working collaboratively with their team to create a professional and polished final report. Through their efforts, the individual demonstrated a strong attention to detail and dedication to producing high-quality work.

Front End Developer – Ionut Boboi: Main role to design and develop the user-facing portion of the project, including the homepage and other related components. Also, focused on creating an intuitive and engaging user experience, while also ensuring that the site was responsive and accessible across a range of devices.

Project Lead/Lead Developer – Ruben Nhez: Had the role of primary developer who overlooked the development of the project as well as develop the back end and middleware code. Also provided market research and created extensions to the project from this research.

Secondary Developer – Simran Johal: Main role was to help aid in creating and coding the project, set requirements and further provide research to the team to guide the project to become a stable and working product.

Head Front End Developer - Tomi Akinyemi: Key role was to design and develop the user facing end of the project as well as help in any part of the project where tasks needed completing. Furthermore, was in charge with researching and drafting mock ups of the designs of the application from start to finish. Also contributed to the Final Report.

To summarise we all had our main roles: Ruben and Simran worked on the middleware and backend creation, while Ionut and Tomi were designing and coding the frontend. Amin oversaw the report and made sure it was formatted correctly and ensured all work was captured in the document. We all provided some input on the research and helped keep the report in a formal and tidy manner.

Meetings and Project Time Management

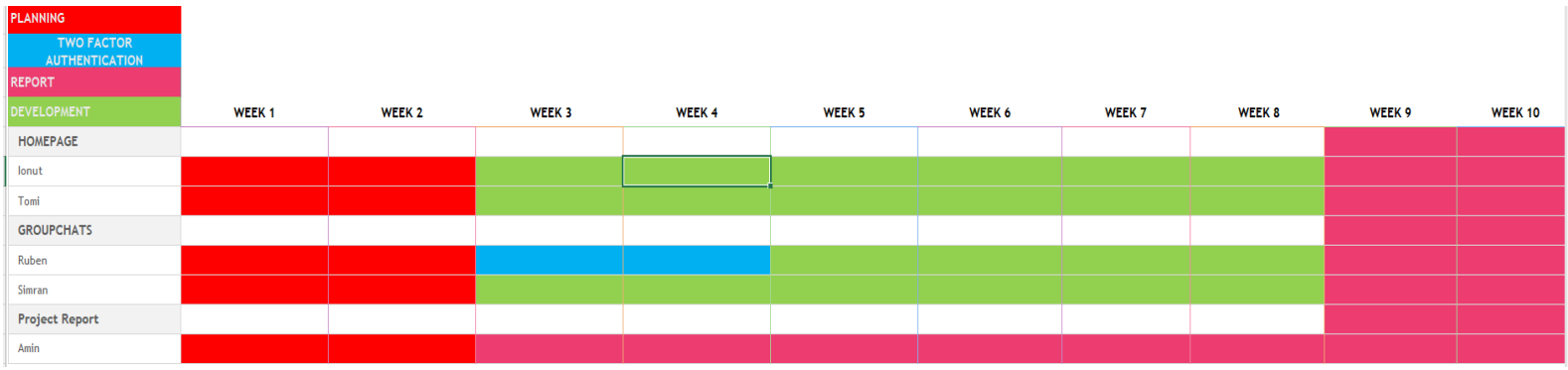
Project Update: Meeting on Week 1 (30 Minutes):

At the onset of our project, we conducted a brief meeting to determine the roles and responsibilities of each team member. Our objective was to divide the tasks efficiently to ensure timely and successful project completion.

We identified that dividing ourselves into two groups of two, with one person responsible for report writing, would be the most effective approach. Ruben and Simran were assigned to the group chat feature, with Ruben specifically responsible for implementing the two-factor authentication (2FA) functionality. Ionut and Tomi were assigned to the homepage, while Amin was tasked with report writing.

Below is the initial time management plan that we created during our first team meeting:

Made with Microsoft Excel



The tasks involved in this project include developing a graphical user interface (GUI), designing the application, creating a table, and working with Microsoft Excel. The progress we made over the course of ten weeks was documented in the log provided above. The first two weeks were dedicated to planning, while the next six weeks were focused on the development of individual tasks. The last two weeks were focused on report writing.

Further details on each task can be found below. Our team remains committed to meeting the project's objectives while adhering to our deadlines.

The home page was created effortlessly; however, our work was not being met in course with the time plan shown above. Due to the fact this was a very new concept to the developers (in terms of creating group chats), we fell behind schedule and were struggling to create the different aspects that were promised in our original MVP. The group chat developers had help from the frontend developers but even still some of the features (which will be explained later) were still not working.

Due to the fact we ran into many problems through our development cycle, we managed to only get feedback from our stakeholders on the features that were working and could not gain much insight on the features that were missing.

Our project was in the end not completed by our original deadline and was still not fully complete by the final deadline.

Key Applications Used

Through the process of creating our project, we had to use many applications to ensure all work was being created and well documented. We also needed to make sure all our work was able to be passed around to everyone in the group so that we could all provide input into the project efficiently. Some of the key applications we used to make sure this was possible are listed below.

To help us manage our time, we used Microsoft Excel which is the application we used to create our project timetable. This provided us a way to know if we were keeping track of the

time, we had left to complete each iteration of our project. This also helped us identify each member of the group roles and made sure we knew what we were all doing.

Microsoft 365 online was used especially for Microsoft Word, as this was the file where our document was being created. This was an easy choice of an application to use due to the fact that everyone had access to it, and it was easy to use. It ensured that everyone was able to see and expand the project report.

The application we used to write our code in was visual studio code. We chose this source code editor as everyone in the team knew how to use it. It was very simple to get the source code and edit it. We could use many programming languages and see which was best fit for creating our project with this editor.

GitHub was used to manage all the software through the development and testing phase. This hosting service allowed for every member in the team to work together on the same piece of software for the project. Each of the iterations of our work was saved on this website.

As we needed to communicate, we had to choose an application we all knew well. We used two main chatting services to speak to each other online when we had run into a problem and needed help. Snap Chat was used for quick chatting and progress updating. We also used Discord for sending images and small files to each other. Both these applications helped us overcome problems and communicate as a team.

3 – RESEACH AND ANALYSIS:

Research Introduction

As we were about to begin creating our application, we needed to know features that were needed to be included in our application. We decided the best way to collect this information was to create a survey which we would share out to our stakeholders.

Our survey would be handed to Goldsmith's students (who are our stakeholders) as well as close friends who also go to Goldsmiths University. We found out that there were some questions that helped us have a definite decision into the features we would add and there were some questions that didn't give us a clear understanding about what we needed to prioritise.

First, we checked if all the people taking the survey were attending Goldsmiths, and 100% said yes. Almost 30 people took part doing this survey.

A question that was helpful in our insight to know what to add to the application was "Does our application need two factor authentication?". Over 80% of participants said this was a feature that would be useful in our application, and so we wanted to focus on developing this feature.

However, there were two questions that did not give us clear answers, as there was a split in choices from the participants. As we were going to add a recommendation system for group chats in our app (meaning the group chats were public), we wanted to make sure this was still a feature the stakeholders wanted. Around 54% said group chats should be public, 11% said groups chats should be private and 36% said there should be a way for group chats be hidden from other users. This was unclear as group chats had to be public for the recommendation system to be added; this system was the initial main feature from our proposal. Even though there was a split in this question, we were still determined to stick with our initial idea as the majority still voted for it.

Another question that was similar to the previous question was to find out how users should be added to group chats. 54% of survey users said the user should be able to use the recommendation of group chats system and join if they would like to, whereas 46% said users should only be invited to group chats. As this was a close split, it didn't really give us an insight into how the joining group chat system would be like.

Eventually, we as a group decided to keep working with the initial group chat recommendations system as well as have all group chats be public for users to join. We believe this would be a better way for users to meet new people at Goldsmiths which was what around 63% of people wanted our app to be for.

The survey questions and the percentage results can be found in the Appendix.

Identifying Our Stakeholders and Developers

Stakeholders:

Goldsmith's students are our main and only stakeholders. Our application is aimed at specifically students who struggle meeting new people in person. The app is meant to act as a safe place for students to meet and make friends with people online.

Consumers:

Our consumers are the people who have the rights to use our product. We have designed our application to attract our stakeholders. One way is making sure our app is free and no charges are placed on our consumers. Our app is targeted at people aged between 18-30, I.e., students at Goldsmiths University. Our application will not be designed to be accessible to people outside of the stated target area.

Developers:

We are the developers who oversee developing and maintaining the product. Main focus is to reduce workload for other developers within the team as well as the stakeholders. We aim to create sustainable code that will require minimal maintenance in the future.

Testers:

Our testing methods include both of Alpha and Beta testing. We as developers will test the product and so will our stakeholders. This will massively reduce the number of problems that occur when the product is fully released.

Design Heuristics

The main goal of heuristic evaluations is to identify any problems associated with the design of user interfaces. Usability consultant Jakob Nielsen developed this method on the basis of several years of experience in teaching and consulting about usability engineering. These heuristics can be applied during the design process or as a tool for evaluating and improving existing interfaces. They provide a set of guidelines for creating user-friendly interfaces that are intuitive, efficient, and error-free. We had a look at Jakob Nielsen’s design heuristics.

Below is a table demonstrating Jakob Nielsen's design heuristics [1]:

Design Heuristics	Explanation of the heuristic
Visibility of system status	The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
Match between system and real world	The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
User control and freedom	Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
Consistency and standards	Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
Error prevention	Even better than good error messages, is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
Recognition rather than recall	Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
Flexibility and efficiency of use	Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
Aesthetic and minimalist design	Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
Help users recognize, diagnose and recover from errors	Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
Help and documentation	Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Heuristic Evaluation

As we develop our chat app, we will be using Jakob Nielsen's heuristics for user interface design to create an app that is both usable and effective. These heuristics were created by Jakob Nielsen and Rolf Molich in 1990 and have since become a widely used framework for evaluating and improving user interfaces.

The first heuristic, visibility of system status, is critical in our chat app. It is essential to keep users informed about their status within the chat, such as whether a message has been sent or not. We will make it clear to the users of who the messages are from by adding a name indicator. The third heuristic, user control and freedom, will give users the power to have control over their chat experience. We will provide users with the ability to send messages or leave conversations. Additionally, our app will allow users to easily back to previous screens.

The fifth heuristic, error prevention, will help prevent users from making mistakes in the app. I will put error prevention measures in place, such as preventing users from sending empty messages or accidentally deleting entire conversations.

The sixth heuristic, recognition rather than recall, will make it easy for users to find conversations and contacts. I will display relevant information about each conversation, such as the last message sent or received, to make it easier for users to remember important details.

The eighth heuristic, aesthetic and minimalist design, will ensure my chat app has a clean and simple design that prioritizes the most important information. The app will avoid cluttering the screen with unnecessary information or features.

4 – REQUIREMENTS

System requirements

Users can use a browser to view the web application known as the MVP. Any widely used operating system and browser should to have the capability ready to use it without any issues.

The application isn't demanding for any hardware above the minimal specifications that a modern computer should have. This covers the needed CPU and GPU. The application has a responsive web layout, so it should function on any frequently used screen size such as smartphones, tablets, computers etc.

Due to the application's basic design and basic layout, we do not anticipate lengthy loading times. However, extremely poor internet links might affect usability such as increasing load times.

User requirements

User Account Creation:

- Users can create a personal account using their email and password.
- Users will be required to complete their profile page with information about their interests, hobbies, ethnicity, and languages.
- Users will be prompted to verify their account using 2FA method.

Profile Management:

- Users can edit and update their profile information.
- Users can view their own profile and the profile of other users.
- Users can upload a profile picture and cover photo.

Recommendation Engine

- The app will suggest group chats or individuals who share similar interests and preferences based on the information provided by the user.
- The app will use an algorithm to analyse users' inputs and match them with groups or individuals that have similar interests.

Messaging System

- The app will include a messaging system that allows users to communicate with each other directly.
- Users can initiate a conversation with other users or reply to incoming messages.

Search Functionality

- Users can search for other individuals based on their interests, hobbies, ethnicity, and languages.
- The app will provide filters to refine search results by distance, age, or other criteria.

User Privacy and Security

- The app will have security measures in place to protect user data, such as encryption and data backups.
- Users can control their privacy settings and choose what information to share with others.
- Users will be notified of any changes or updates to the app's privacy policy.

User Experience

- The app will have a clean and user-friendly design with a straightforward layout that enables users to navigate the app without difficulty.
- The app will be optimised for different screen sizes and devices.
- The app will provide helpful tips and instructions for users to use the app effectively.

Non-functional requirements

Security:

- Authentication: The application must have a robust authentication system to ensure only authorised users can access it.
- Encryption: The application must use encryption to secure user data, including personal information, messages, and media files.
- Access control: The application must have proper access control measures in place to prevent unauthorised access to user data.
- Data protection: The application must use appropriate measures to protect user data against theft, loss, or damage.

Usability:

- User interface: The application must have an intuitive and easy-to-use interface that enables users to navigate the app effortlessly.
- Consistency: The application must maintain a consistent design and layout throughout all screens and menus.
- Clarity: The application must have clear and concise instructions and prompts to guide users through the app's features.

- Responsiveness: The application must be responsive and fast, providing an efficient user experience.

Accessibility:

- Assistive technologies: The application must be designed to support assistive technologies, such as screen readers, to make it accessible to users with disabilities.
- Colour contrast: The application must have sufficient colour contrast to ensure readability for users with visual impairments.
- Alternative text: The application must provide alternative text descriptions for images and other media to make it accessible to users with visual impairments.

System:

- Scalability: The application must be designed to scale up to handle a growing user base and increasing data volumes.
- Reliability: The application must be reliable, with minimal downtime and a backup system in place to ensure that user data is not lost.
- Performance: The application must perform efficiently, with minimal latency and quick response times.
- Compatibility: The application must be compatible with different operating systems and devices to ensure it is accessible to all users.

Ethics:

- Privacy: The application must respect user privacy and adhere to relevant privacy laws and regulations.
- Transparency: The application must be transparent about how it collects, uses, and shares user data.
- Accountability: The application must be accountable for any data breaches or misuse of user data.
- Non-discrimination: The application must not discriminate based on race, gender, sexual orientation, religion, or any other protected characteristic.

5 – DESIGN AND DEVELOPMENT

The Application Concept

Our concept application compromises of many features to improve the easiness of meeting new people at Goldsmiths. The application we are designing consists of different group chats where students can find like-minded people. There will be a system where users can locate other users' profile and see the things they are interested in/like. Users will then be able to join certain group chats based on what they like.

This was the main concept we wanted to follow through our development cycle. However, as we didn't know our coding capabilities to the fullest, we wanted a back-up plan in case something went wrong e.g., lack of time or coding incapabilities. Through our early initial survey stated in the research & analysis chapter, we found that there was an almost split decision in whether group chats should be invite-only or public. This meant that even though we were still going to focus on implementing the recommending chat system, our back-up was just to have a normal group chat where it was invite-only and for Goldsmith's students only.

Below we will focus on the prototype features we developed and designed for our application and why we chose them.

Home Page Prototypes and Iterations

The homepage is an essential part of any website or application as it's the first point of contact for users. This report will outline the planning, development process, and testing of the homepage of a website. Additionally, it will discuss the reasons behind certain colour schemes used and why testing is crucial.

Before starting the development process, we researched various popular websites such as Discord, WhatsApp, and Slack to analyse how their homepage is presented. We found that most of them showcase their features and try to sell their app to potential users. Specifically, we liked the colour tones used on Discord's website and found the way WhatsApp split their homepage into sections, each with its own information, to be effective. From Slack, we gathered inspiration for the homepage's design layout. Based on this research, we decided to implement similar components on our website's homepage.

Based on the wireframes from our proposal [Appendix], we started the development process by creating a homepage that showcased the app's key features, provided easy navigation, and an attractive layout.

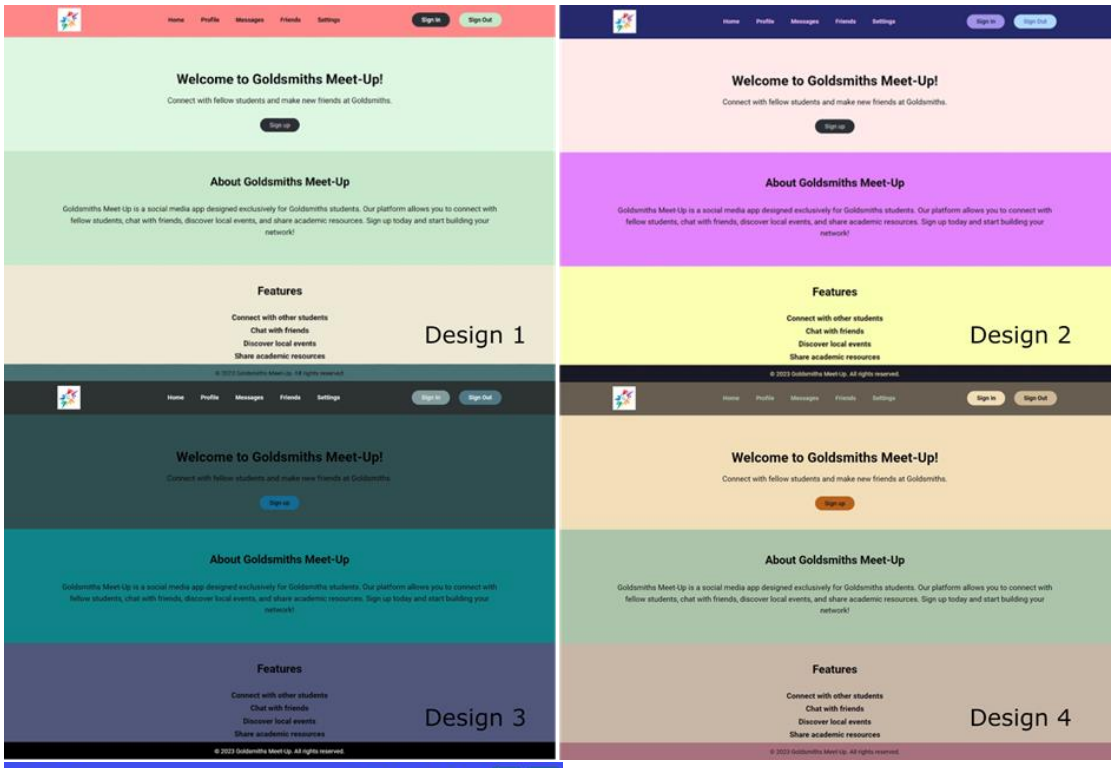
As we developed the homepage, we created five different prototypes with varying colour schemes to determine which one was the most popular. The prototypes were distributed to our stakeholders (Goldsmith's students), and they were asked to choose which design they found most appealing (from a small survey). After collecting and analysing the responses, we found that prototype number five, which featured a specific colour scheme, received the

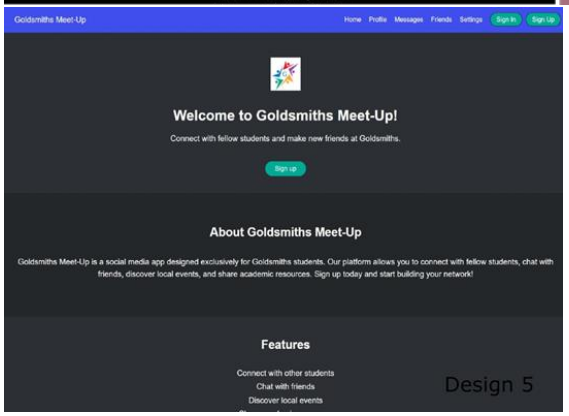
most votes. Based on these results, we decided to implement this colour scheme into the design of the homepage. This survey helped us ensure that the design of the homepage was user-friendly and visually appealing to potential users.

We used blue and grey (colours chosen from survey) as the primary colour schemes as blue is known to evoke trust, loyalty, and professionalism, while grey provides a sense of balance, stability, and neutrality. Using PHP, CSS, and JavaScript, we translated the wireframes into a functional homepage. We focused on responsiveness, accessibility, and performance and tested the design on multiple devices and browsers to ensure compatibility and responsiveness.

Survey can be found in Appendix.

These are the five iterations of the prototypes we made for the home page:

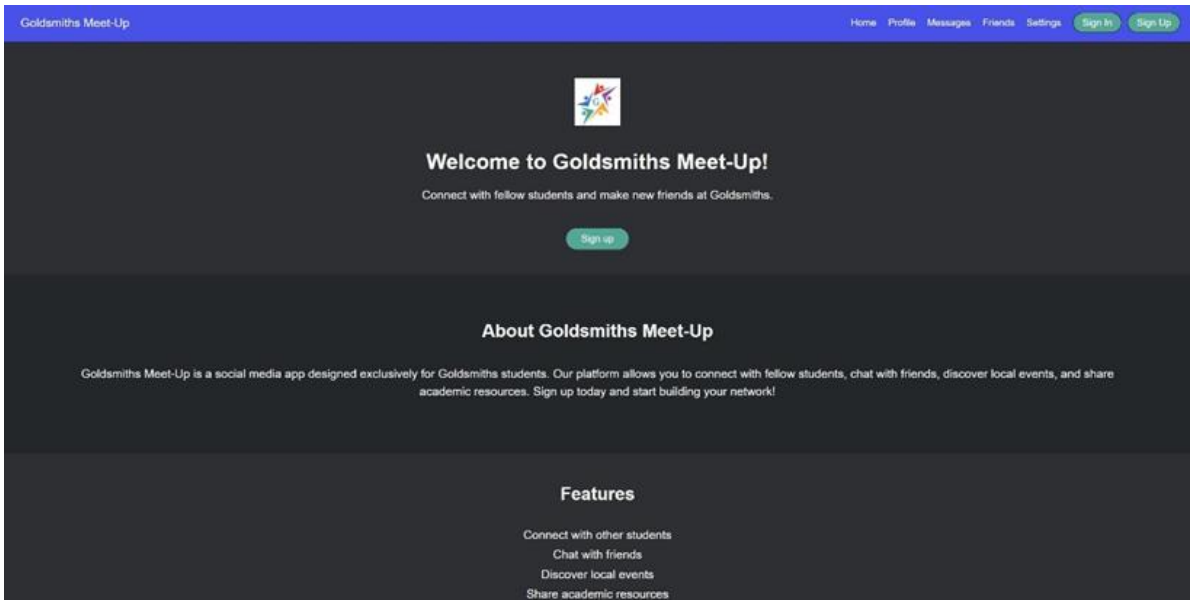




In conclusion, the development of a homepage involves careful planning, designing, and testing to ensure that it effectively communicates the app's key features and benefits. Colour psychology plays a vital role in creating a visually appealing design that attracts users, and

testing is crucial to ensuring the homepage's functionality, usability, and accessibility. We also made sure the home page was as simple as possible as this is what the majority of our stakeholders wanted from our research & analysis survey. We believe we have followed these key concepts to creating a home page that expresses what our application is about.

Final design of our home page is shown below:



Home Page Future Work

We believe the final iteration of our home page is perfect and do not plan on updating it in the near future.

Group Chats Prototypes and Iterations

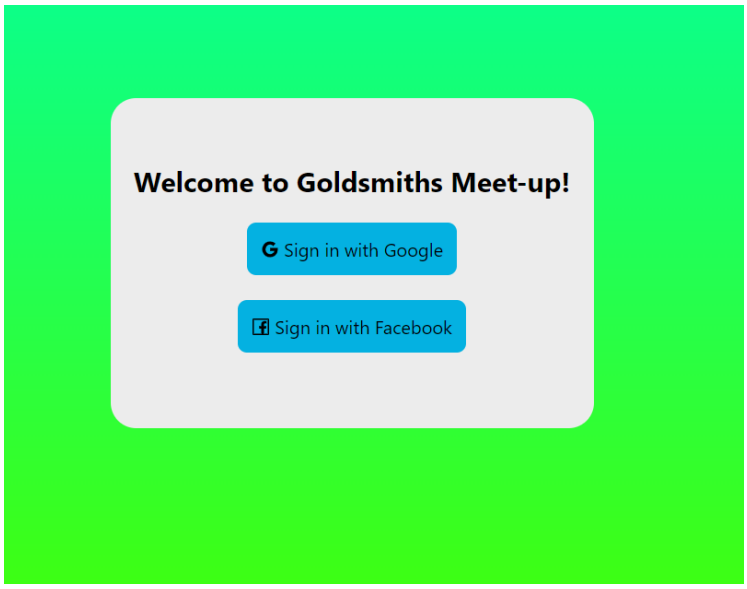
Group Chat Prototype 1 and Research

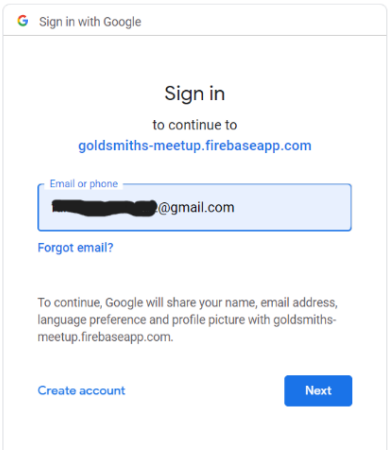
During our first bit of searching for creating our group chats, we found a video on YouTube that created a chat application that uses JavaScript, Firebase and Chat Engine. The video is by JavaScript Mastery [2]. As we were struggling on finding a place where to start, we decided to use the source code, which is linked in the video, and work from there. The Chat Engine [3] is an API which makes it easy to build chat services. As this was everyone in the group's first time building a chat application, we thought that it would be easier to start off by using an API that simplifies the troubles of creating the chat app. We knew it would take a lot of time for all of us to create an app from scratch, so as a group we decided to first begin with free source code and learn about the behaviours of a chat app from there. So, this means our first iteration would be mainly about improving our coding skills and knowledge of creating a chat app from pre-written code. The app used Firebase [4], which is a Backend-as-a-Service (BaaS) app development platform that provides hosted backend services such as a real-time database, cloud storage, authentication etc, and it would take care of the login part of our system by making sure the user is using their own email as you would have to be logged into your email to access the application. Java Script would be used to connect

between these two by calling these APIs, making sure they are working well together and help build the application.

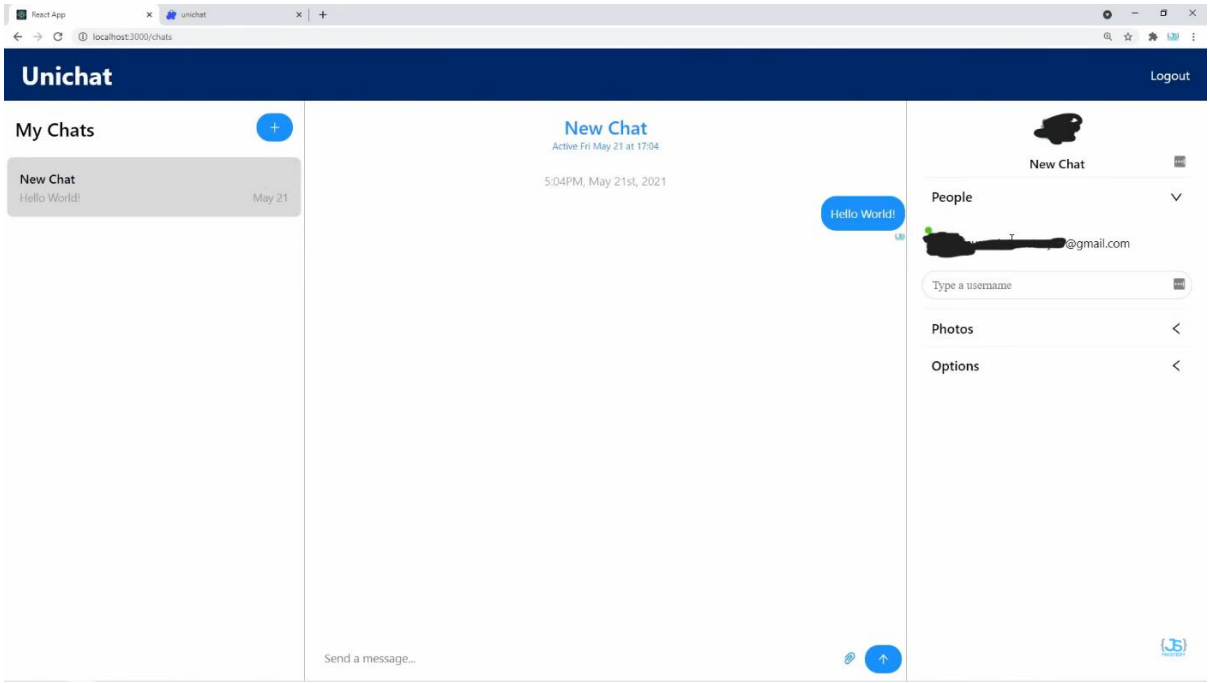
Our iteration 1 design can be seen below. Although it was a good starting design it wasn't quite what we were looking for as the only way you could add someone to the chat that you created would be by clicking on someone in the people section and the other person would automatically be added. Therefore, there wasn't much of a choice on whether or not you wanted to be added to a random chat as once you were invited by someone, it would automatically add you to the chat. The initial colour scheme used was also not the same as for the home page; however, this was an initial group chat, so it did not matter during the time.

The sign-up page is shown below for iteration 1:





The group chat page when signed in is shown below for iteration 1:



Summarising the Pros for Iteration 1:

- Chat generated by an API therefore we didn't have to spend lots of time implementing a chat as it was pretty much done for you.
- You can send GIFs and they would be stored in Photos so you can always go back to it without needing to scroll up.
- You can send files to other users in your chat.
- Simple and easy design to understand (this was stated in our MVP)

Summarising the Cons for Iteration 1:

- You can't join a group chat unless your added by someone else.
- If someone clicks on you with the intention of adding you to a chat, you'll be automatically added so you don't have much of a choice if you want to join or not.
- APIs are an external dependency that we rely upon for our project yet have virtually zero control over
- GIFs may be inaccessible as a file format, may create clutter on both physical and virtual memory space.
- The design doesn't follow the same patterns as our home page, the group chat and home page seem like two different applications
- Uses too many APIs, can be confusing and difficult to add features in future

Group Chat Prototype 2 and Research

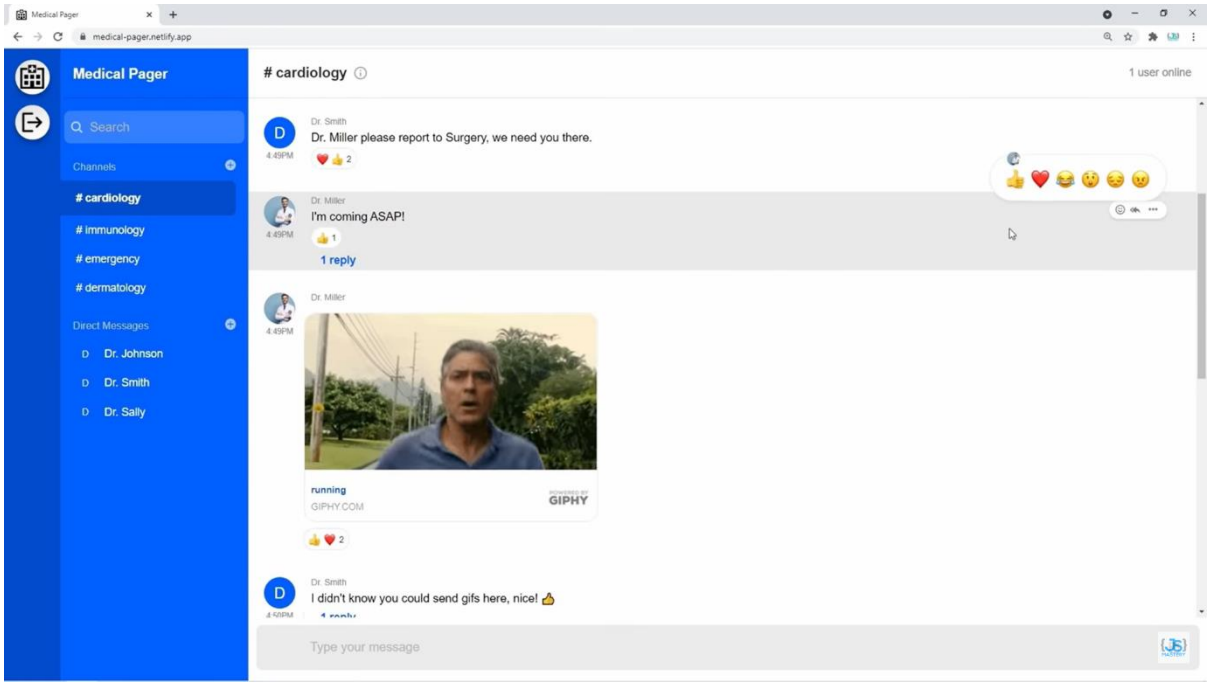
Now we wanted to start working and implementing the skills we learned from prototype 1 and create the second prototype. We didn't want to keep working on prototype 1 as we were still missing a few features such as the recommendation system and profile page, and we didn't know how to implement them in an already built group chat app. We found a video [5] by JavaScript Mastery, where we found more starter source code that helped us

begin the with the second prototype. It was a chat app which uses one API called Stream [6] and would allow our users to create different text channels similar to discord where they could make a chat based on interests, ethnicity, hobbies etc. This is one of the features that was stated in our concept; we wanted different chats based on different hobbies/interests. The API Stream powers chat messaging and activity feeds for billions of global end-users across thousands of different apps it has a rapidly ship in app messaging with highly reliable chat infrastructure and feature-rich SDKs which would improve the overall in-app conversation, engagement and retention, it would take care adding users to chats, GIFs, notifications, authentication, etc. This is why we chose to change the API for the second iteration, as it seemed to us it was better for chat messaging (reasons stated above), and we wouldn't have to take care of two APIs like the first iteration.

Why we decided to create more iterations, and didn't choose this design:

Although this was an improvement in terms of features added to prototype one, we were still missing a few features like the recommendations system and profile page. We knew we had to create at least one more prototype to try adding the missing features, so we chose to leave this design behind and start on the next iteration. Another reason we didn't choose this as our final prototype was because we realised the design was too similar to one of our competitors Discord [image comparison in Appendix], and we wanted our user interface to feel unique and not the same as our competitors.

Below is an image showing the group chat for the second prototype:



Summarising the Pros for Iteration 2:

- Channels available to all users so everyone can add chats based on what they are interested in; this was one of the features the stakeholders voted on in our research survey [Appendix].

- You can use GIFs and interact with people's messages such as "like" a message.
- Users are able to use emojis, wasn't in our MVP, but we believe it is a plus for our application
- Done by an API therefore not long time spend coding a chat since it's created and managed by the API
- We have moved from 2 to 1 API from iteration 1 to 2
- Direct messages meaning you can choose to have a 1 on 1 conversation or more.
- Simple design layout

Summarising the Cons for Iteration 2:

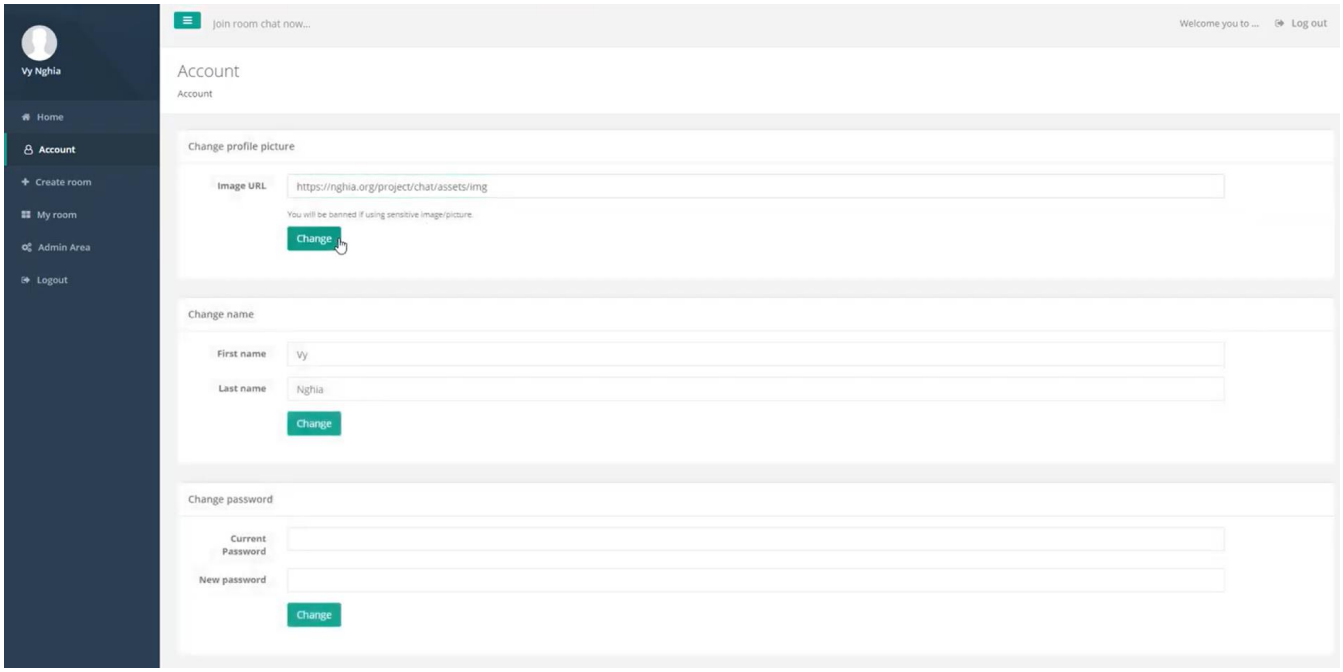
- Depending on the number of users the channels can be flooded with many chats.
- Not much privacy unless in a 1 to 1 conversation, as all chats are public
- The design layout is too similar to one of our competitors "Discord"
- APIs are an external dependency that we rely upon for our project yet have virtually zero control over

Group Chat Prototype 3 and Research

The third prototype was inspired from a YouTube video [8] where the template uses PHP, JavaScript, SQL, HTML and CSS. Since most of our developers knew how to use PHP, SQL, HTML, CSS and JavaScript, we knew this would be a template which we would understand. PHP is known for creating dynamic web applications and to interact with databases. Since we were creating a chatting application which would use databases, we knew we had to incorporate this language into our work to make it easier for ourselves. Originally, we were only planning to use JavaScript, SQL, HTML & CSS. This application first works by the user having to register which then they'll be added to a SQL table in our database. The same table will save their profile then the user would be able to login then change their account information such as name, last name password etc. The user can also create a room and then if someone would like to join their room, they would need to input the room id then wait for the request to be approved by the owner of the room before they can join and conversate.

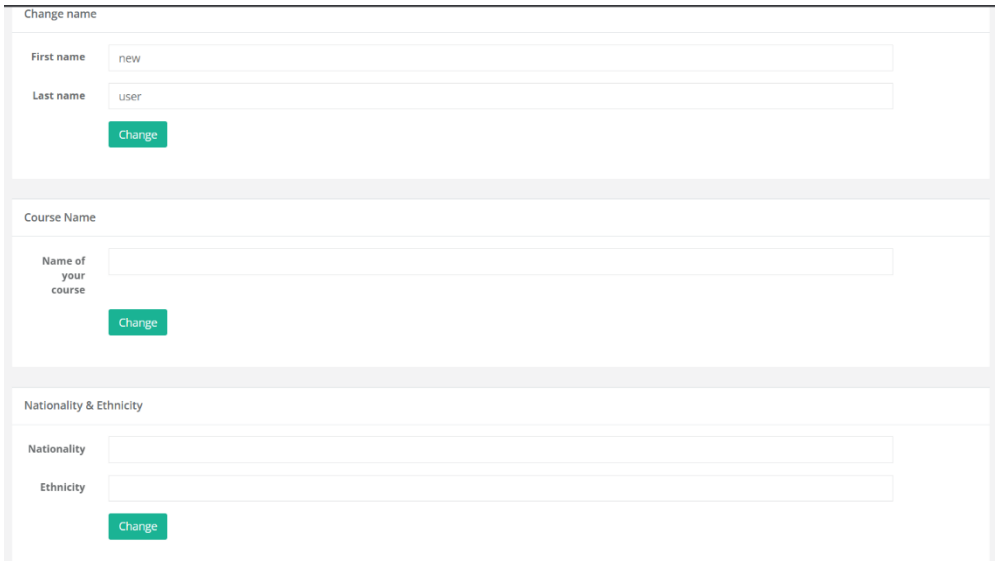
Below shows the images of the third prototype:

Account page:

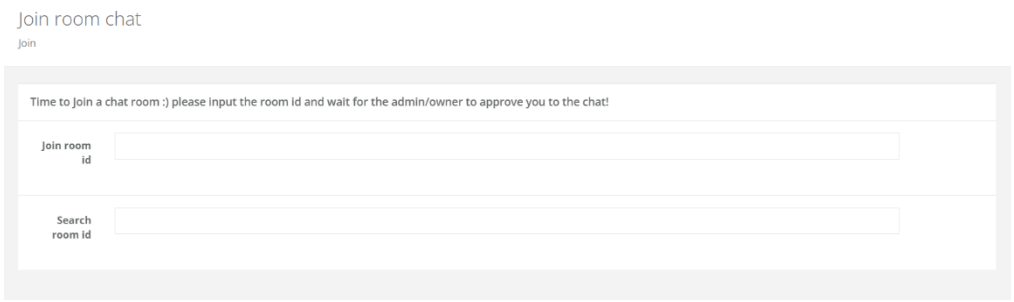


Registration Form:

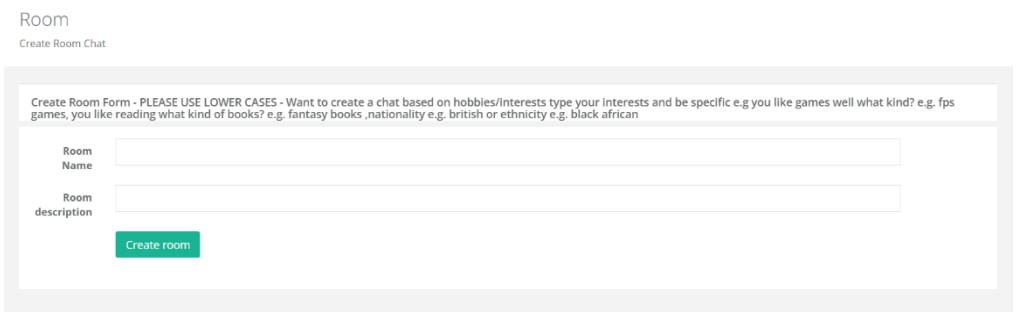
User Profile:



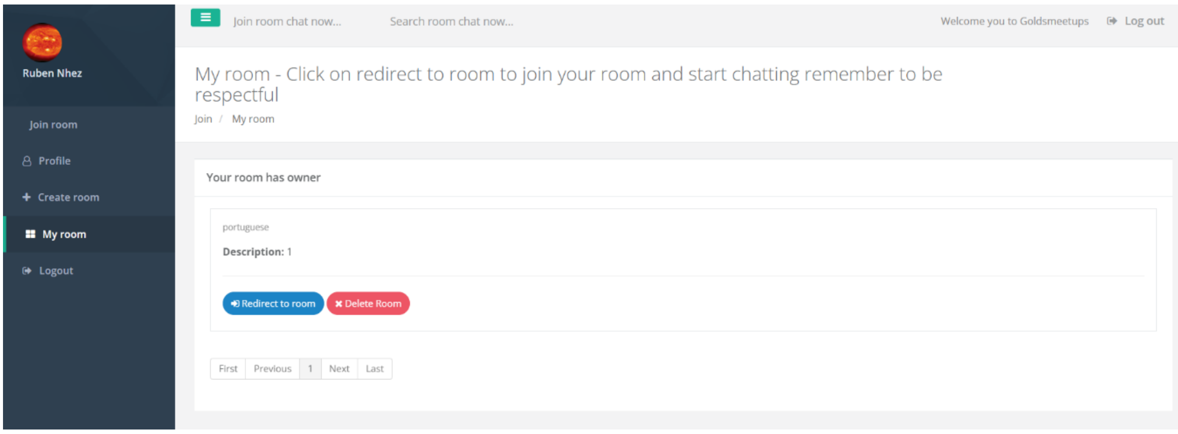
Join chat with simple description on how it works:



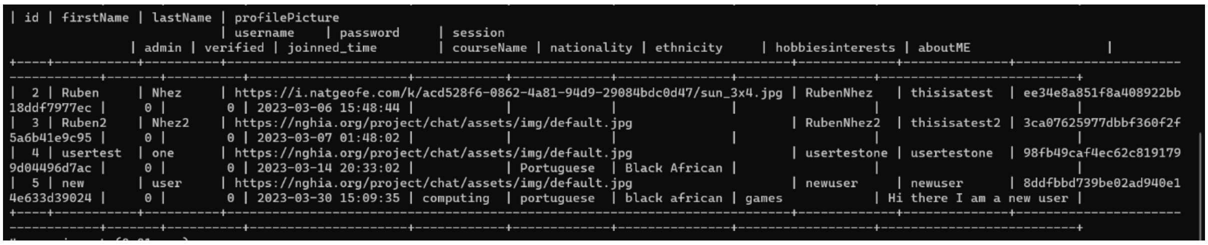
Create room and description on how it works:



My room and simple description on how it works:



Changes to SQL table for user:



```
| id | firstName | lastName | profilePicture
                            | username     | password       | session
                  | admin | verified | joinned_time           | courseName | nationality | ethnicity     | hobbiesinterests | aboutME                  |
| 2 | Ruben     | Nhez     | https://i.natgeofe.com/k/acd528f6-0862-4a81-94d9-29084bdc0d47/sun_3x4.jpg | RubenNhez   | thisisatest  | ee34e8a851f8a408922bb
18ddf7977ec |     0 |        0 | 2023-03-06 15:48:44 |             |             |               |                  |                          |
| 3 | Ruben2    | Nhez2    | https://nghia.org/project/chat/assets/img/default.jpg                     | RubenNhez2  | thisisatest2 | 3ca07625977dbbf360f2f
5a6b41e9c95 |     0 |        0 | 2023-03-07 01:48:02 |             |             |               |                  |                          |
| 4 | usertest  | one      | https://nghia.org/project/chat/assets/img/default.jpg                     | usertestone | usertestone  | 98fb49caf4ec62c819179
9d04496d7ac |     0 |        0 | 2023-03-14 20:33:02 |             | Portuguese  | Black African |                  |                          |
| 5 | new       | user     | https://nghia.org/project/chat/assets/img/default.jpg                     | newuser     | newuser      | 8ddfbbd739be02ad940e1
4e633d39024 |     0 |        0 | 2023-03-30 15:09:35 | computing   | portuguese  | black african | games            | Hi there I am a new user |
```

What we changed in the app:

We are making use of the login/registration and log out system; nothing has changed except registration, we added text boxes to store the course that our users are studying, their nationality, ethnicity and an about me section so that we can store the information in a SQL database and use it later for their profile. In the SQL we added courseName, nationality, Ethnicity and aboutMe to the user table to store the user’s information. We also changed home and turned it to a join room instead with a simple instruction on how to use it at the top, my room was also added with a simple description to tell the user how to see their chat; account was changed to a profile and text boxers were added to display information about our users and they can choose to change it if they want whether that’s their “about me section”, name etc. Lastly in create room a function was added so that if there is an existing room with the name as what the user intends on creating, for example, say you want to create a chat for your nationality and you’re Spanish; if you try to create a Spanish chat and one already exists it won’t create a chat and instead it should display the room id for the existing chat. The project doesn’t have the CSS as of now, but we intend to change it in the next iteration.

As stated, users are able to create their own chats based on what they want whether that is their hobbies, interests, ethnicity or nationality and other people are only able to join by inputting the chat room id number and even then, they would have to be approved by the admin/owner of the chat room.

Group Chat Surveys

We as a team decided that iteration three presented an application that met almost all features (features not met covered in next chapter) that were stated in our concept and

MVP. We believe it represents the best version of what we wanted in the first place (during proposal).

However, the most important thing we needed to know was if our stakeholders felt the same way. We design a small survey that showcases our three prototypes and asked a handful of our stakeholders which had the best design, and if all features that were promised were added or not. We also asked if they would use this app.

Through our survey we found out that just like the developers, prototype three was the most liked among our stakeholders with a chosen percentage of 90.9%. We also found out that 63.6% of stakeholders thought that we added most of the features that were promised in the original concept. We asked them what was missing, and we only received one answer: the recommendation system (covered in the next chapter). However, even though some of our stakeholders thought some features were missing, 100% said they would still use our application to commute with Goldsmith's students.

This survey can be found in the Appendix.

Group chat Future Work and Conclusion

Even though we managed to add many of the features into our application that was stated in our concept and MVP, we were still missing a major feature that was missed out during the development stage: the recommendation system. This was the main feature that was the most anticipated in the iteration process and was the attribute that would off made our application feel very unique; it was stated in the initial market survey that this concept was unique by our stakeholders [Appendix]. The reason for this mishap, was due to two factors, lack of time and lack of knowledge from the developers in this area of coding group chats. As we didn't know where to start, our time management was all over the place, and we had to focus on the main feature which was to make sure that the group chats were functional. Our knowledge wasn't the greatest in creating this app, so we did indeed have to rely on pre-written free to use source code. We as a group know we have over promised and under delivered, however we believe it is not all bad as the group chat works and, in our research and analysis survey [Appendix], we found out that the 54% preferred chats to be public whereas 46% preferred it to be private. This shows that there is almost a 50-50 split and if group chats are private, we wouldn't be able to implement the recommendation group chat system. Since we had to make them private and only joinable with a special key ID, we did in fact please almost 50% of the other stakeholders who wanted the chats to be private.

However, as our initial promise was to add a recommendation system and we as a team have realised our mistakes, we feel motivated and intend to add this missing feature in the near future and update our app to fulfil the addition of the recommendation system.

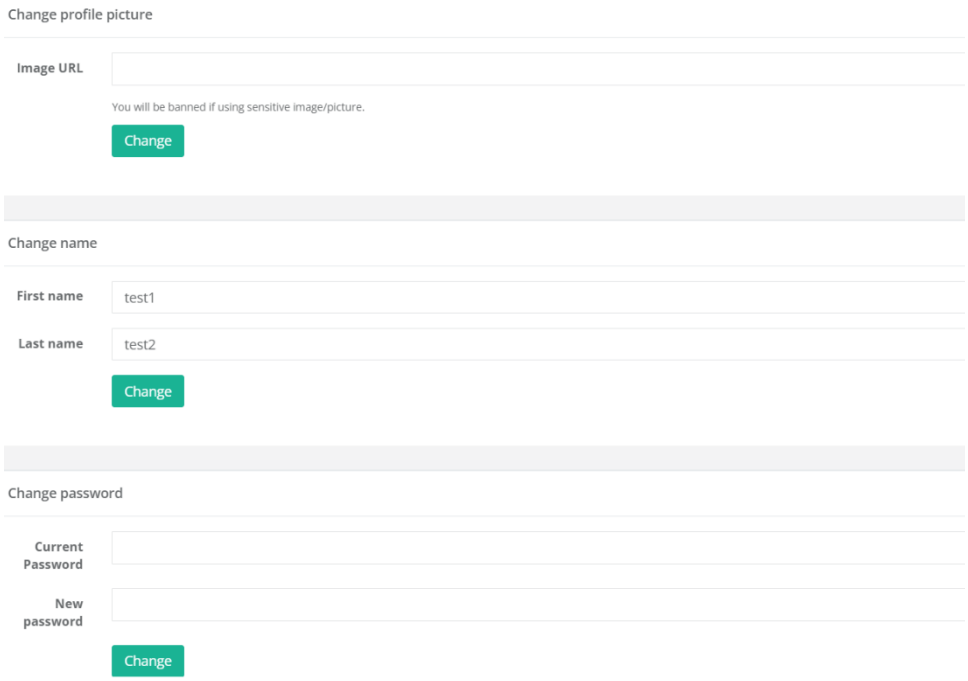
Profile Page Prototypes

When we started our profile page, we wanted there to be a section where users can express themselves and tell a bit about themself. This was mandatory, as our initial concept was

about users being able to showcase their, for example, hobbies, interests and ethnicity and finding group chats with similar aspects.

Our starting prototype of our profile page was very minimalistic, and only had a few features. It only had features that could change name/surname, profile page and change password.

Below is an image of initial profile page:



As you see, our profile page does not contain the features which we wish for. We had to implement a way for users to fill our different boxes. Another feature we wanted was there to be a way for users to change these boxes, in case e.g., they no longer have an interest in something or if they change courses.

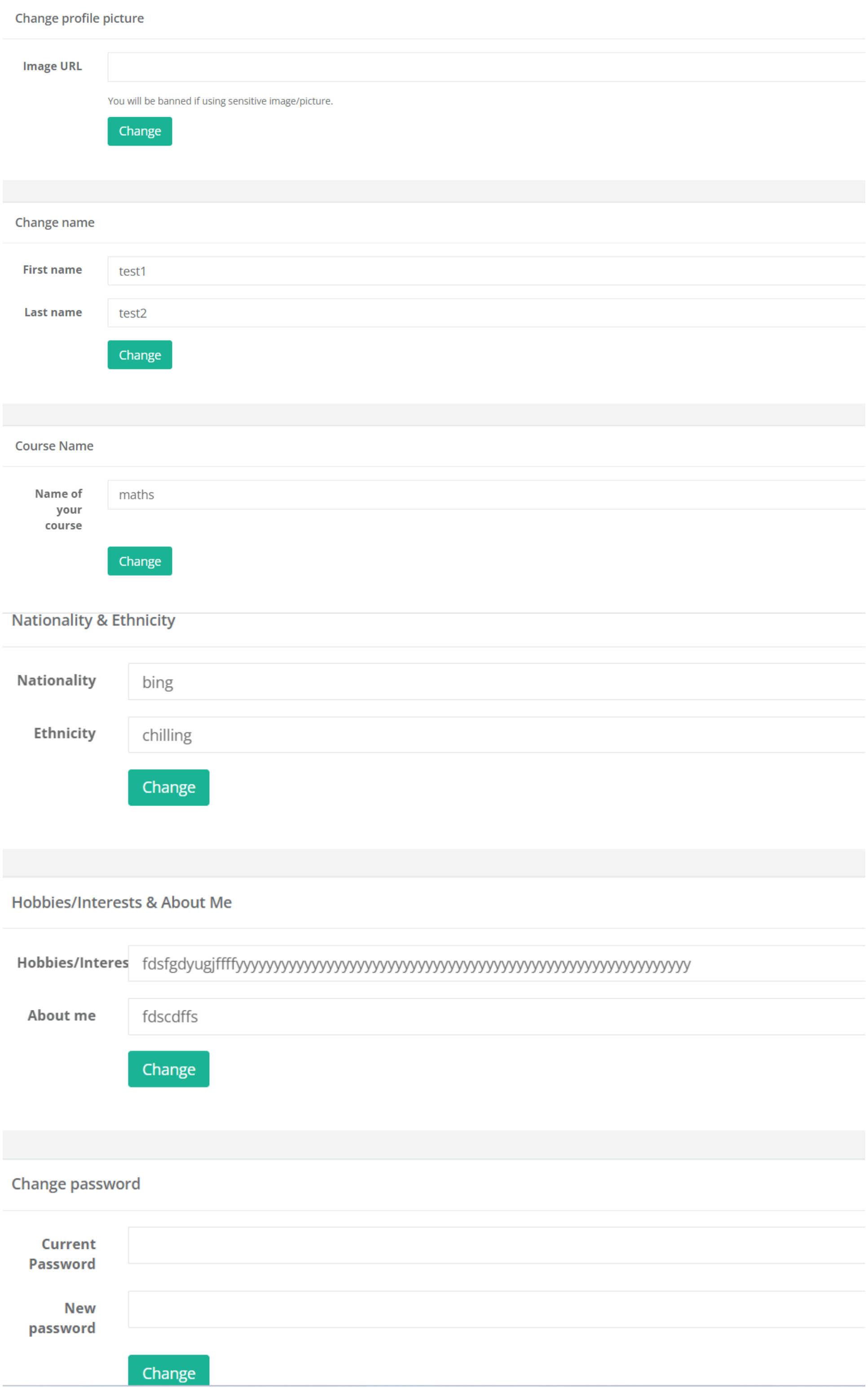
The development team implemented these features; however, we still didn't know if anything was missing. The ideas from our initial concept were added, but we needed outside feedback to see if the profile page was at the best, it could be. We created a small survey just for the profile page to see if users liked the design and features added.

We asked if users like the design of the profile page and 80% said yes and 20% said no. We asked the people that didn't like it why, and they said it was a little too bland. They would have preferred a few more features like a picture posting page and video sending page. Due to time constraints, we couldn't think of a way to implement this in time. However, we are determined to think about these points in the future.

We asked users if they thought they needed anything added as well. The majority said yes, and the reason for this was because they realised you could only type a few words in the profile page. This was an error that was missed by the developers; our app was focused on a way for users to express themselves, and it would be impossible to if they could only write a few words. Due to this, we bumped up the numbers in the "about me" section to 1000

characters. We didn't want to risk going higher as we did not know if this would cause performance issues. We thought that 1000 characters was enough to write a small paragraph about yourselves. [The profile survey can be found in the Appendix]

Below is an image showing the final design of profile page (without CSS added):



Since we want this feature to be a mandatory attribute, we added these fields to the register page, so you must fill them in initially before accessing our app. This is one of the reasons

you can change the details on account page, as you might not know what to write on register page.

See register page with the feature added:

Two Factor Authentication Prototypes and Iterations

A Two Factor Authentication, or 2FA [9], is an extra layer of protection used to ensure the security of online accounts beyond just a username and password.

A 2FA is a good way to make sure that the user is giving us correct information when necessary. Using a 2FA will reduce the chance of fake users to be inserted into our system as only users that own that email can create an account. This will reduce the likeliness of bot accounts. Our project “Goldsmiths Meet Ups” will be using an email verification which takes the email that the user inputs and the system will send a random generated code to the user’s email which the user then has to input back into the app. Then the system will determine whether the two match each other.

While researching on the internet for email authentications using JavaScript, we stumbled upon a YouTube video by AshisCoding [10] using an SMTPJS which is a free library you can use for sending emails from JavaScript and we had to set up a server in a website called Elastic email [11] which takes care of sending the emails and keeps a records of the emails and messages it sends.

Our 2FA works by taking the email inputted by the user then the user will receive a pop up stating that an OTP code email has been sent to their spam email and then a text box will appear asking for the OTP code to be inserted and followed by a verify button. If the user writes the wrong OTP, a display message stating that the OTP is incorrect will show, else if correct, a message stating the OTP is correct is shown. To see how this works please check the “Testing” chapter.

As previously stated on our secondary Market Survey [Appendix 1] our users were a fan of any colour with a high contrast (55.9%), hence I decided to use a yellow for the background and a nice blue for the button to make the design of the Two Factor Authentication easy to see.

Below is the design for the Two Factor Authentication:

Below is a code snippet of how the two-factor authentication works. The code works by generating a random number on line 9 followed by an email body on line 11 to state the structure of what the email would look like. Then in “Email.send” on line 16, we call the private secure token from elastic email which will take care of sending emails to our users. Then the rest of the code under neath checks whether or not the OTP code is correct and its outputs.

```
8
9   let otp_val = Math.floor(Math.random()*10000);
10
11  let emailbody = `
12      <h1> See Your Goldsmiths Meetups OTP Down Below</h1> <br>
13      <h2>Your OTP is </h2>${otp_val}
14  `;
15
16  Email.send({
17      SecureToken : "aa7ff2e0-316f-4ca6-b9c0-33669adc7760",
18      To : email.value,
19      From : "rnhez001@campus.goldsmiths.ac.uk",
20      Subject : "This is the from Goldsmiths Meetups",
21      Body : emailbody
22  }).then(
23      //if success it returns "OK";
24    message => {
25      if(message === "OK"){
26          alert("OTP sent to your email "+email.value + " Please Check your Spam mail");
27
28          // now making otp input visible
29          otpverify.style.display = "block";
30          const otp_inp = document.getElementById('otp_inp');
31          const otp_btn = document.getElementById('otp-btn');
32
33          otp_btn.addEventListener('click',()=>{
34              // now check whether sent email is valid
35              if(otp_inp.value == otp_val){
36                  alert("OTP correct your email has been verified");
37              }
38              else{
39                  alert("Invalid OTP try again");
40              }
41          })
```

Two Factor Authentication Future Work

Even though we managed to create the two-factor authentication system, we couldn’t implement it into the actual group chat app. This was due to the fact that we wanted to come back and implement it during the end of the development cycle, however we ran out of time due the focus in implementing the actual group chat system. Since we also didn’t exactly know how to implement it, it would have taken much longer for us to find out how. We believe for the future it would not be too hard, however since we changed the login system from email to password, we would have to change that system. We decided to make

the user login with username instead of email as this is what most stakeholders wanted [Survey in Appendix], however we as developers believe it would be safer and more secure with using an email and two factor authentication; so, in the future we plan on adding this feature.

6 – SYSTEM DEVELOPMENT & TESTING:

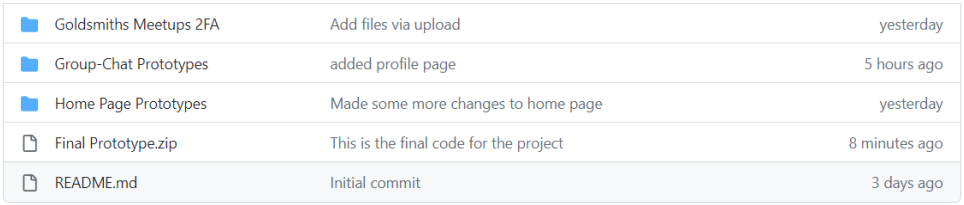
Throughout this chapter we will cover our testing strategies as well as the source code used to develop our application.

GitHub

We used GitHub to store our prototypes and share code with each other. This allowed us to maintain the code and enable version control. We as a group are not the most confident with GitHub and not all of us knew how to use it during the starting stages of the development process. We didn't know how to create branches, so we created folders where we put each iteration of the prototypes. Before everyone knew how to use GitHub, some members of the team preferred to email source code to each other.

Due to this bad decision, we couldn't locate our prototype 2 code and so the repository only consists of iterations of prototype 1 and 3. We also moved all our code to a second repository (the one being submitted) due to the first having account issues.

In the image below you can see the several folders we created to store our source code:



The "Goldsmiths Meetups 2FA" stores our two-factor authentication system which we could not implement in time. The "Group Chat Prototypes" shows all the prototypes we made for our application. The "Home Page Prototypes" shows all the home page iterations we created during the development process.

Since we were novices when using GitHub, we weren't very effective at using its key features such as raising issues and fixing bugs within GitHub's IDE.

Our log files will be submitted along with the report and source code separately.

The link to our repository [15]

Front End Development

When we started working on the front-end side of the chat app, we had to consider important factors [12] when selecting the features and technologies we wanted to implement. Our main priorities were to optimize user experience by creating a fully functional home page that had a simplistic design for user visuals impair abilities [13] [14].

Our initial concept wanted our framework to be built out of JavaScript, HTML & CSS, because everyone in the development team had working knowledge with these coding languages/ mark-up languages. However, as our group-chat app was now being developed using PHP [reasons stated in Development chapter], we had to change the front-end files to

PHP. This was easy as all we did was change the .HTML extensions to .PHP. This now meant we could like our front-end files to the middleware and back-end files. Not all our front-end developers had knowledge in PHP, but they were all aware of this change and managed to learn the basics of this databasing language quickly (Ionut and Tomi).

The front-end was completely built by our front-end developers Ionut and Tomi, and they were in charge of building the home/landing page.

We used PHP tags which decided what to display to the user. For example, if the user was on the login page, it would display a label for “account”, “create room” and “my room”. When the middleware knew where the user was, it would display the user with this information. The same can be seen for “isAdmin” where it would display the labels for “Admin Area”.

The PHP tags can be seen below:

```
<?php if($inLogin): ?>
<li class="<?= $accountMenu ?>">
    <a href="account.php"><i class="fa fa-user-o"></i> <span class="nav-label">Account</span></a>
</li>
<li class="<?= $createRoomMenu ?>">
    <a href="create_room.php"><i class="fa fa-plus"></i> <span class="nav-label">Create room</span></a>
</li>

<li class="<?= $myRoomMenu ?>">
    <a href="my_room.php"><i class="fa fa-th-large"></i> <span class="nav-label">My room</span></a>
</li>

<?php if($isAdmin): ?>
<li class="<?= $adminArea ?>">
    <a href="secret_admin.php"><i class="fa fa-cogs"></i> <span class="nav-label">Admin Area</span></a>
</li>
<?php endif; ?>
<li>
    <a href="<?= $_LOGOUT_FILE ?>"><span class="nav-label"><i class="fa fa-sign-out"></i> Logout</span></a>
</li>
<?php endif; ?>
```

Using PHP to decide the rendering of HTML to the user was a decision we made as a team. This allowed us to create multiple files (which are named according to its function), rather than keeping it all the decisions and scripting in one html.

Elements that kept repeating were put into a separate file, so that we wouldn’t have to keep repeat coding them. For example, the nav bar on the home page as well as the nav bar for the group chat page were kept in separate files so we didn’t have to rewrite them for each page. This allowed us to save time and keep the nav bar consistent across the home page and group chat pages.

Nav bar for home page:

```
<nav>
    <ul class="menu">
        <li class="logo"><a href="#">Goldsmiths Meet-Up</a></li>
        <!-- <img class="logo" src="assets/logo.png" alt="logo"> -->
        <li class="item"><a href="#">Home</a></li>
        <li class="item"><a href="#">Profile</a></li>
        <li class="item"><a href="#">Messages</a></li>
        <li class="item"><a href="#">Friends</a></li>
        <li class="item"><a href="#">Settings</a></li>
        <li class="item button"><a href="#">Sign In</a></li>
        <li class="item button"><a href="#">Sign Up</a></li>
        <li class="toggle"><span class="bars"></a></li>
    </ul>
</nav>
```

Nav bar for group chat page:

```
<li class="<?= $homeChatMenu ?>">
    <a href="<?= $_HOME_FILE ?>"><i class="fa fa-home"></i> <span class="nav-label">Home</span></a>
</li>

<?php if($inLogin): ?>
<li class="<?= $accountMenu ?>">
    <a href="account.php"><i class="fa fa-user-o"></i> <span class="nav-label">Account</span></a>
</li>
<?php endif; ?>

<li class="<?= $createRoomMenu ?>">
    <a href="create_room.php"><i class="fa fa-plus"></i> <span class="nav-label">Create room</span></a>
</li>

<li class="<?= $myRoomMenu ?>">
    <a href="my_room.php"><i class="fa fa-th-large"></i> <span class="nav-label">My room</span></a>
</li>

<?php if($inLogin): if($isAdmin): if($adminArea !== "active"):?>

<li class="">
    <a href="secret_admin.php"><i class="fa fa-cogs"></i> <span class="nav-label">Admin Area</span></a>
</li>
<?php else: ?>
<li class="active">
    <a href="secret_admin.php"><i class="fa fa-cogs"></i> <span class="nav-label">Admin Area</span> <span class="fa arrow"></
    <ul class="nav nav-second-level collapse" style="">
        <li><a href="#room_control">Group Control</a></li>
        <li><a href="#user_control">User Control</a></li>
    </ul>
</li>
```

We believe we have created a reasonable user interface as developers, and all the features that have been implemented have been tried and tested and work in the app. We will test our UI with our stakeholders in the testing chapter.

Middleware Development

Next was the development of our middleware. Our middleware must issue the functionalities to the user. It is written in PHP within PHP tags. As you see in the image below, our code makes queries to our database, hence why some lines are written in SQL. Some functions will use mysql extension to execute a SQL query against a database.

Our middleware is designed to interact with both the front-end and back-end. The front-end will request information from the middleware and the middleware will receive answers to the request from the back end.

For example, returning a user's info by username:

```php
function searchUser_bUsername($db, $username){
    if(!empty($username) && !mysqli_error($db)){
        $sql = mysqli_query($db, "select id,firstName,lastName,profilePicture from user where username = '$username'");
        if(mysqli_num_rows($sql) > 0){
            $result = mysqli_fetch_array($sql);

            return $result;
        } else {
            return;
        }
    } else {
        return;
    }
}
```

Upon receiving an event from the user on the app's frontend, such as a login attempt, the middleware component translates the user's input into a SQL query and sends it to the database. The query asks the database to check whether the login information matches the information stored in the database. The middleware then handles any exceptions returned by the database. If the query returns a valid result, the middleware logs the user in and directs the frontend to the group chat page, which is then rendered for the user.

```php
if(!empty($_POST["username"]) && !empty($_POST["password"])){
    $username = (string)$_POST["username"];
    $password = (string)$_POST["password"];

    $sql = mysqli_query($db, "select * from user where username = '$username' and password = '$password'") or die(json_encode(array("
```

We have tried to make our application as robust as possible. This means we have tried to make our website work on as many different browsers as possible. We believe we have done this, as we have tested that our website works on Google Chrome, Microsoft Edge and Safari (we haven't found a browser that our application doesn't work on). We also have tried to implement as many fail cases as possible. This means that in the case of an unexpected error, the website should give out some type of output explaining what has happened to the user. One example we have implemented is when a user inputs incorrect login details. We have made sure that a message pops up on screen explaining the situation to the user. In this situation, the user is prompted by the message "Login failed. Please try again!".

```php
} else {
    $loginStatus = array("success" => false, "message" => "Login falied. Please try again!");
}
```

Back End Development

Finally for our back-end development, we decided right from the start (the proposal) that we would be using MySQL to create and maintain our databases. Our team has the most knowledge in MySQL and it has very fast query performances, hence why we chose this database management system.

Our database has been normalized to ensure no data is repeated. Data such as usernames cannot be repeated, as usernames must be unique. Our tables also have unique names and are labelled in a way so that future developers can understand what is in each table.

Below is an image of our database and our tables:

Tables_in_goldsmeetups
ban_list
chat_room
messages
request_join
room_member
user

As we see, we have six tables all used for their own purposes. We have used primary keys and foreign keys to link our tables. For instance, ID is a primary key in user and foreign key in ban_list. The table below shows the primary key in the user table.

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
firstName	text	NO		NULL	
lastName	text	NO		NULL	
profilePicture	text	NO		NULL	
username	text	NO		NULL	
password	text	NO		NULL	
session	text	NO		NULL	
admin	tinyint(1)	NO		NULL	
verified	tinyint(1)	NO		NULL	
joinned_time	datetime	NO		NULL	
courseName	text	NO		NULL	
nationality	text	NO		NULL	
ethnicity	text	NO		NULL	
hobbiesinterests	text	NO		NULL	
aboutME	text	NO		NULL	

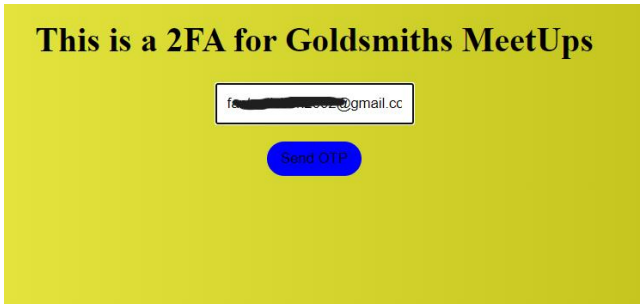
Two Factor Authentication Unit testing

While creating our two-factor authentication, we as developers had to test if the functionality of our two-factor authentication system worked. This is called unit testing, and we decided to use this testing method as we were in the process of checking small pieces of code, and seeing if they worked as intended.

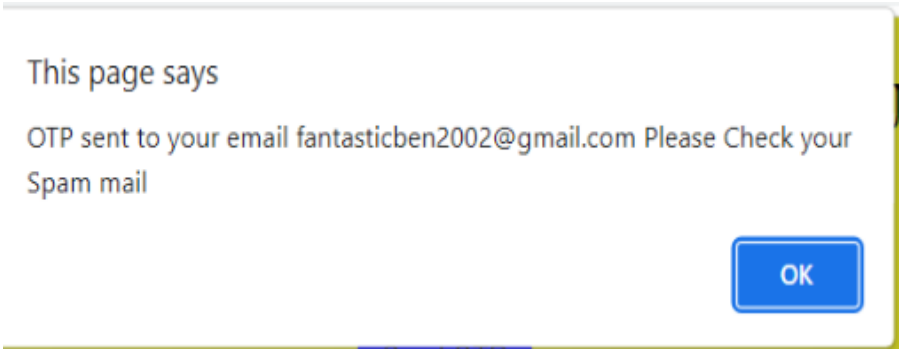
As stated, we decided to use this testing method on our two-factor authentication and see what happens in different use cases of the system.

Use-case 1 – Trying to verify an email address, by first entering a wrong OTP code, then the correct code:

Step 1: Input Email:

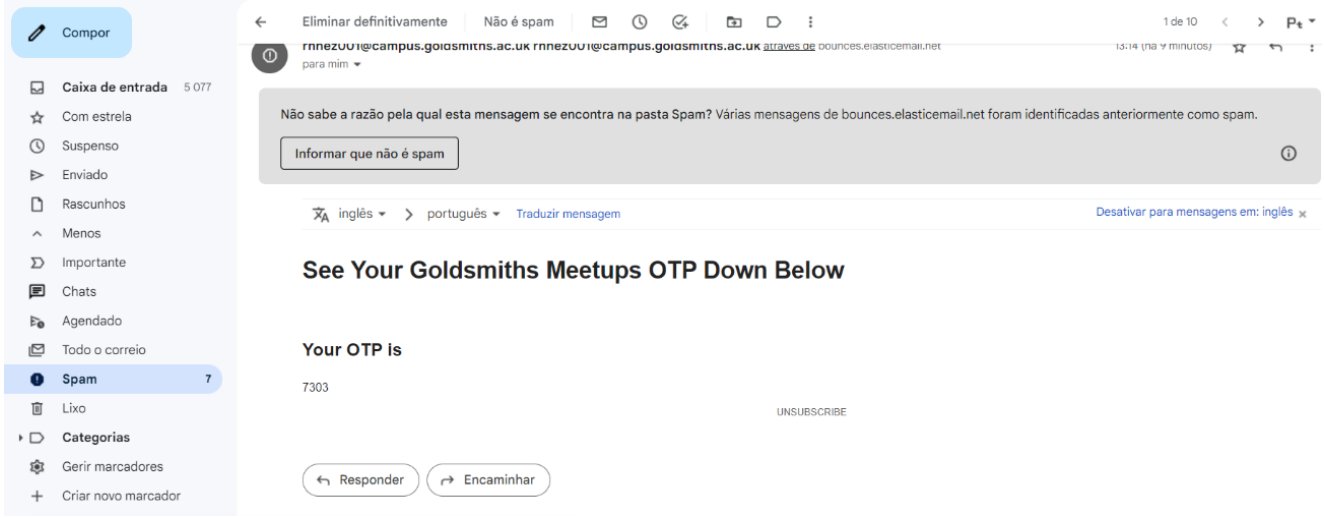


Step 2: Message is displayed saying an OTP code has been sent to your email address and to check spam folder:

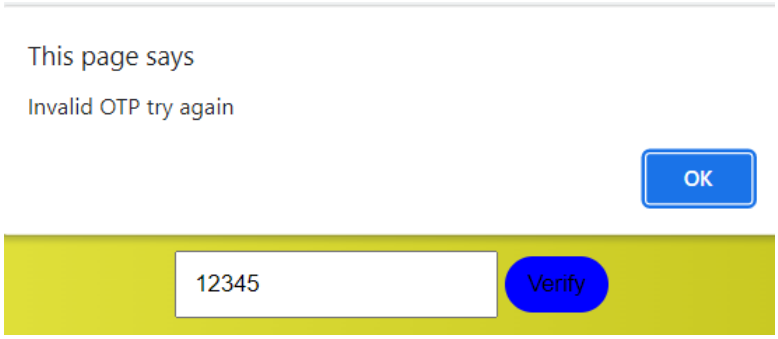


Step 3: A verify OTP box will appear:

Step 4: Check spam email for the OTP code



Step 5: Input Incorrect OTP to check the error message pop-up:



Step 6: Input correct OTP to check the valid message pop-up:

The unit test conducted on our two-factor authentication system, shows that the system has passed all the tests, because this was the intended output on this use case.

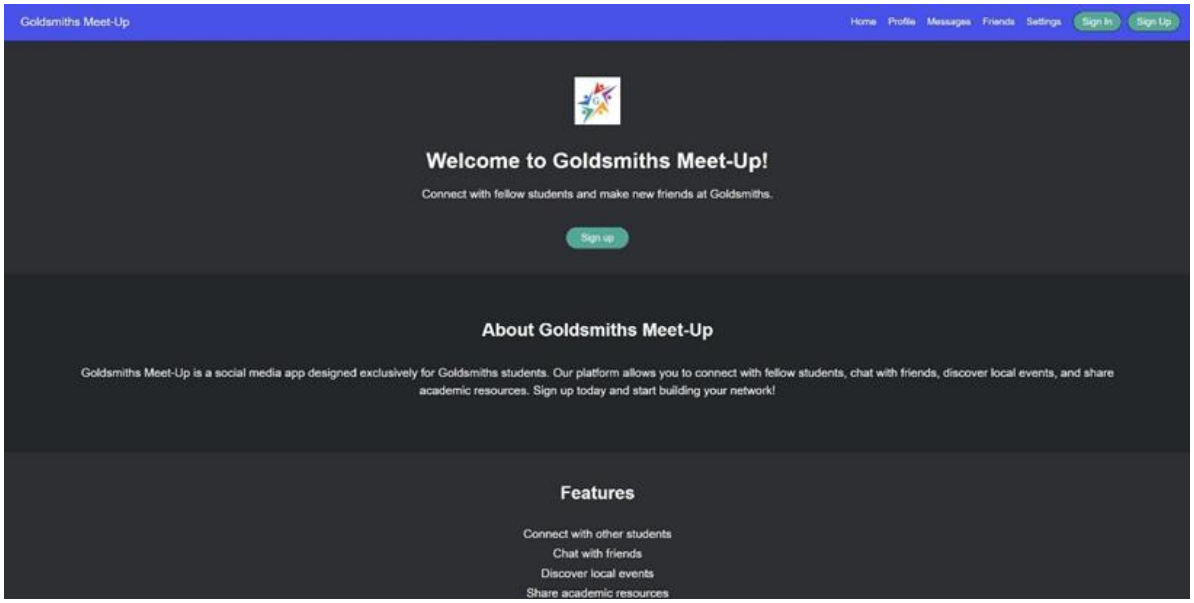
As stated in the development chapter, our authentication system makes use of Elastic Email API. For context some of the features that the elastic email API is doing in the background is; taking a record of when the OTP email was sent; checks whether user has engaged with the message, how many emails were sent and opened; keeps history of each email.

The images of Elastic Email interface can be found in the [Appendix].

Home Page Unit Testing

During testing, we ensured that the website was error-free and responsive by checking it on various screen sizes, and thoroughly checked the site for typos, grammar errors, and proper punctuation across all pages. We also checked the consistency of the font formatting and ensured that there weren't different fonts for the same section e.g., three different fonts for one navigation bar. Through these various testing methods, we were able to ensure that the homepage met the requirements of the project (stated in the concept) and was visually friendly to less abled users (for example, we didn't have a hefty design and kept it simple in terms of both design and colour scheme; our stakeholders stated in our survey that they found the design simplistic [Appendix]) and accessible.

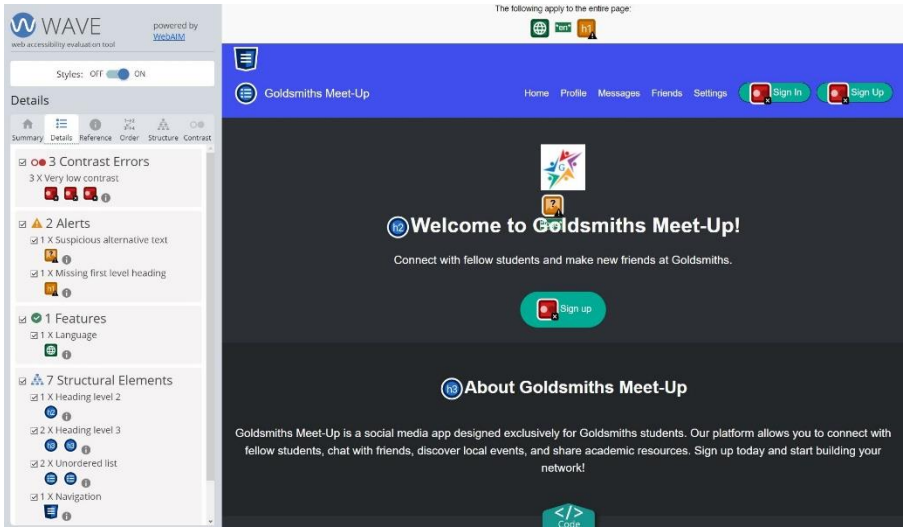
The image below shows our home page on a desktop:



The image below shows our image on a phone:

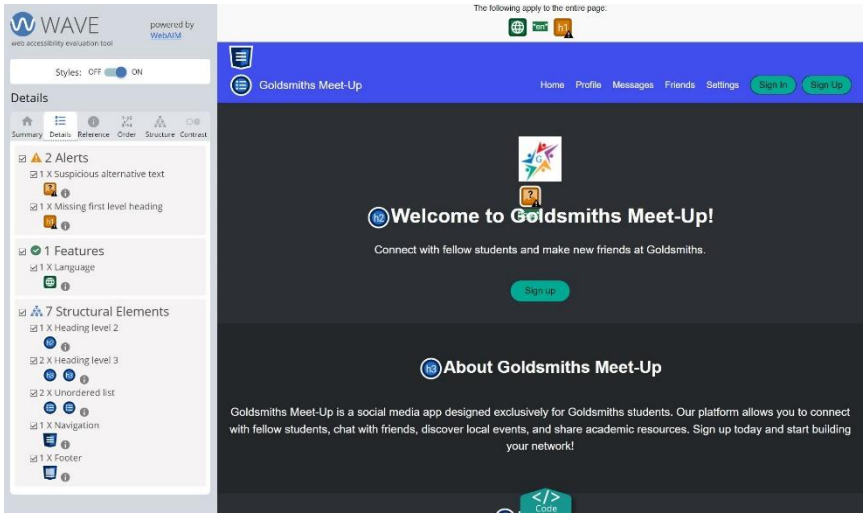
Accessibility Test on Home Page

We conducted an accessibility check on the home page of the application using an accessibility checker tool. The tool reported three errors related to low contrast in the buttons, which could have made it difficult for users with visual impairments to distinguish the buttons from the background.



To address these issues, we modified the font colour of the buttons to black, which significantly improved the contrast between the buttons and the background. This change ensured that the buttons were more accessible to users with visual impairments and provided a more user-friendly experience for all users.

We then re-ran the accessibility checker, and it reported no further errors or issues related to contrast.



Stakeholder & User Tests

As our main motive was to create an application specifically to make Goldsmith's students social lives feel more comfortable and easier to meet people, we knew we needed user feedback to check if our system was indeed following our motive. Our app had to be easily accessible and usable by all our users. To test if our app was doing this, we decided that our three prototypes should be put through accessibility and usability tests.

However, due to the fact that we weren't very time efficient during the development stage, we couldn't create a summative evaluation for all our three prototypes. The reason we didn't have time wasn't only due to time inefficiency, but also because of factors such as mid-term assessments, course work and some of our team members had personal problems which we cannot discuss in this report. Due to these factors, we couldn't create tests for all prototypes. So as a team we decided that it would be best to only conduct tests for our final prototype.

We had close friends at Goldsmiths as well as other random students carry out our tests to see if there were any usability issues. We had no more than ten students carry out the usability tests.

Our usability tests were to check if the users could function our application, whereas our accessibility test showed whether users could function the system without help from anyone else.

Usability Test on Final Prototype

The usability tests were mainly to see and understand if users could work around the application the same way as the developers. Due to time constraints, only a few questions were asked and recorded by our tester team.

Q1) Can you login to the application?

This was a trick question as you had to first register an account before logging into our chat app. All testers found this out and found how to reach the register page. They all registered and managed to login to the application.

Q2) Can you find the logout button?

All testers immediately found the logout button and managed to log out of their accounts.

Q3) After logging in, do you know what page you are on?

Most users immediately knew what page they were on (home page), however not all users were quick enough and said it wasn't clear enough.

Unfortunately, as this test was conducted right before the deadline to submit, we couldn't go back and make the home page clearer. However, in the future we have decided this would be a feature we would improve and make it clearer to the users.

Q4) Can you locate the profile page?

All users were quickly able to locate the profile page.

Q5) Can you locate the page to create a chat room and create a room to chat in?

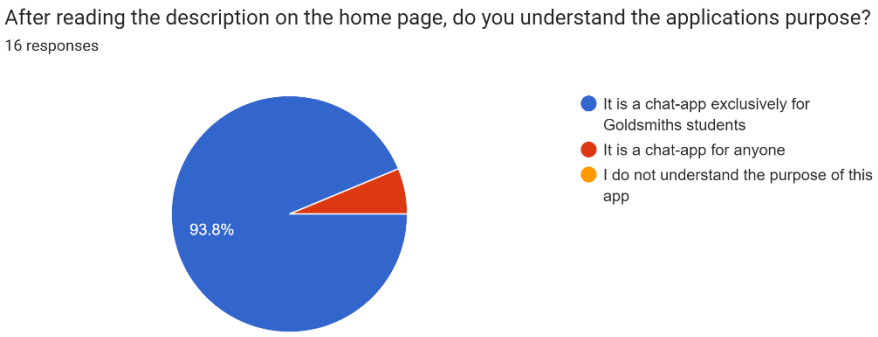
All testers were able to find the “create room” button on the side bar and were directed to create a room page. They then gave the room a name and description and pressed the button that created the room for them.

Accessibility Test on Final Prototype

Our accessibility test was to see if users could use our app without supervision. The survey was given to the same testers as the usability testers, and we also included a few more testers from Goldsmiths. Our survey was created using Google Forms, just like all our other surveys. We provided the testers with images with the survey to make sure they knew what each question was about. As with the usability form, are accessibility form wasn’t too big due to the time we had left to complete this assignment.

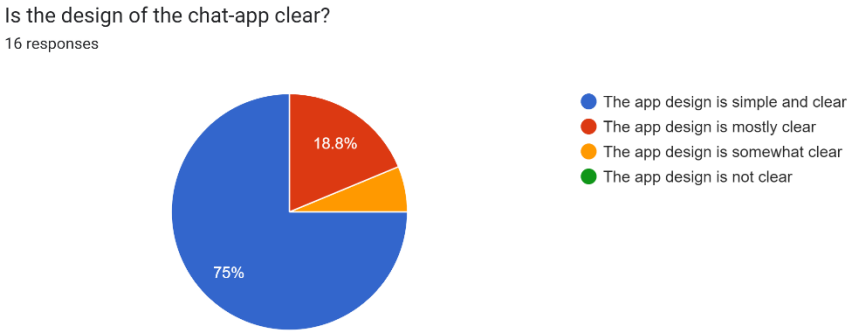
Q1) After reading the description on the home page, do you understand the applications purpose?

Almost all our users knew (93.8%) that our application was a group-chat/messaging application exclusively for Goldsmith’s students. However, 100% of testers knew that our app was indeed a chat-app. Since we knew that the majority knew what the app was about, we didn’t need to change anything on the home page.



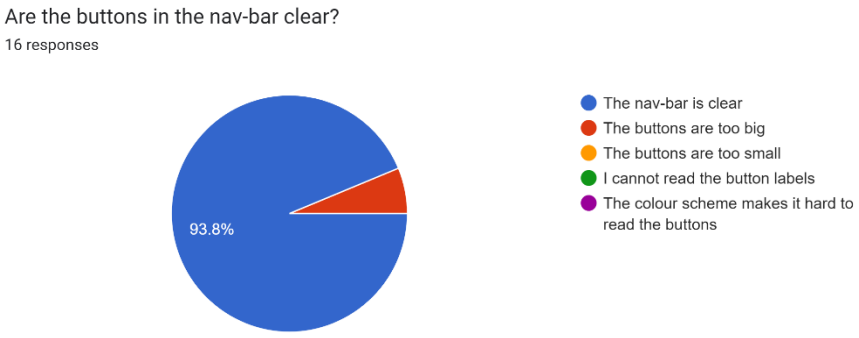
Q2) Is the design of the chat-app clear?

75% thought that our application was clear in terms of design, whereas the other 25% thought it was mostly or somewhat clear. Due to our incompetencies, we didn’t ask what they didn’t found clear in our app design. However, as again the majority thought it was clear, we didn’t plan on changing anything in terms of the design.



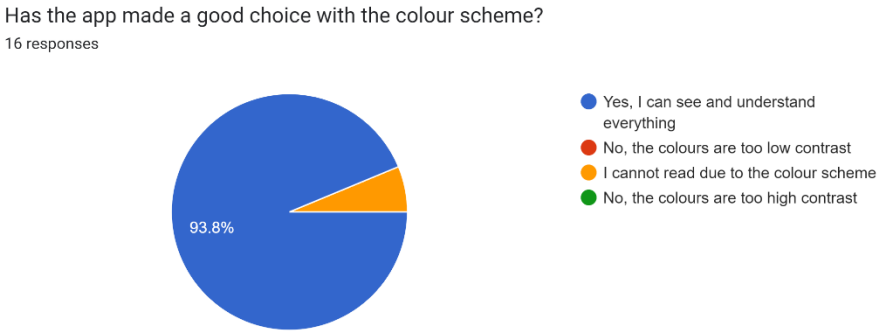
Q3) Are the buttons in the nav-bar clear?

This was a question to see if our nav-bar was clear and if users were easily able to see the buttons and their labels. 93.8% of our survey testers thought the buttons were clear and a small percentage thought they were slightly too big, due to this we only reduced the nav-bar buttons by a few pixels and didn't change them too much.



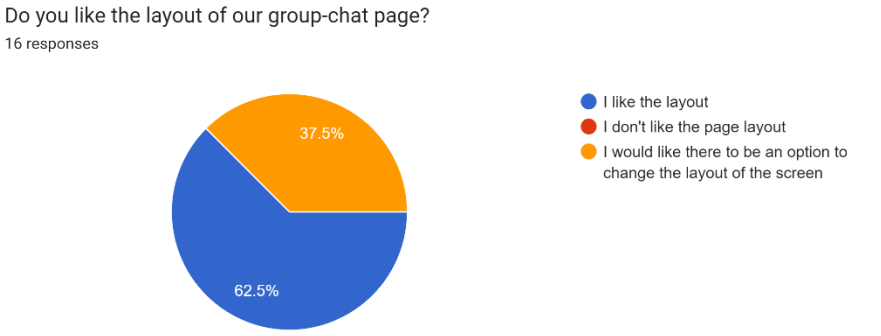
Q4) Has the app made a good choice with the colour scheme?

Our question was to see if the colour scheme reduced visibility when looking at our app. We wanted our app to be accessible to all users, even users with some type of visual impairments or colour blindness. A small portion of our testers, who cannot be named, had either some type of colour blindness or they needed visual assistance (glasses). 93.8% said that the colour scheme was clear and everything on the screen easily readable. The rest of the percentage however said they could not read due to the colour scheme. Since, we did not see the results of the survey during the time, we did not know which one of our stakeholders said this. As the majority thought the colour scheme was readable, we had decided as a team that it was not appropriate to change the scheme at this point, but we would think about this during future developments; so it is clear for all users.



Q5) Do you like the layout of our group-chat page?

This question was asked to see if the layout design was liked by our users. Our results showed that 62.5% of users thought that they did like the layout. However, the rest of the percentage thought that it would be a good idea if there was an option to change the layout of the screen. This was another feature where we didn't have time to implement as the whole front-end code had to be rewritten and reorganised. We thought this idea was useful as it would give the user more freedom and allow them to change the layout the way they liked, so we thought this was a feature which would be added in the future.

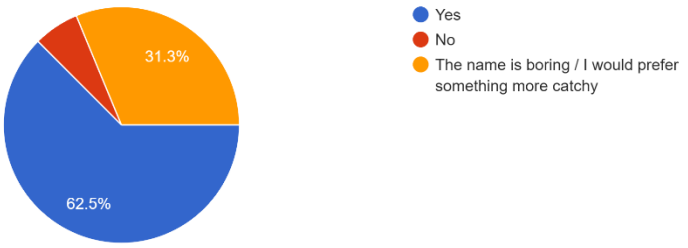


Q6) After reviewing our application, do you think the name "Goldsmiths Meet-Ups" fits the application?

The last question was asked to see if a user would know what the app is from the name and if the name fits the application. The majority of testers (62.5%) thought the name fits the application, whereas the rest thought the name was either boring or not catchy. The name may be too sophisticated and wordy, but the majority thought it was fine so we decided that we would not change the name.

After reviewing our application, do you think the name "Goldsmiths Meet-Ups" fits the application?

16 responses



7 – EVALUATION:

Evaluation Introduction

As we have come to an end in creating our group chat app, we knew we as a team had several strengths that contributed to the success of our application, as well as some areas that could be improved.

One of the greatest strengths of our team was the use of Bootstrap to help with CSS. This is a great decision as Bootstrap is a widely used framework that offers a variety of pre-made styles and components, which can significantly speed up the development process. By leveraging Bootstrap, our team was able to create a professional-looking and consistent user interface throughout the application. For example, we had some small details of animations that could be seen on e.g., the home page where the buttons lit up when hovered.

Another strength of our project was making the application responsive to work on all screens. This is an important aspect of modern web design as more and more people access websites and applications on mobile devices. By making our application responsive, we ensured that users could have a seamless experience regardless of the device they are using.

Our team's focus on accessibility and user-friendliness is also commendable. By designing with accessibility in mind, we were able to create an application that is more inclusive and can be used by a wider range of people. Similarly, by prioritizing user-friendliness, we ensured that the application is easy to use and navigate, which can lead to increased user engagement. We measured these two attributes through surveys, where we found many positives as well as improvements we could make in the future [Found in accessibility chapter].

However, even with all these strengths, our most important achievement was being able to create a seamless experience for Goldsmith's students online where they can chat with each other and create friendships based on having similar qualities.

Despite all these strengths, there were a few areas where we as a team could work on. For a team with not much experience with making applications, we found it quite difficult to stay on the same page. We all worked at different paces and had to learn new techniques in different time periods. Since no one in or group wanted to be the person with the least input or knowledge we were influenced by each other into stayed motivated. We had to use a variety of techniques into completing this project and this was one of our biggest challenges, since we weren't always able to complete certain tasks due to a lack of knowledge. For example, our recommendations system [covered in "Future Work" chapter] was a big factor into meeting our initial concepts and MVPs, but we could not implement it due to time constraints and lack of knowledge in this area of expertise.

Since this was also a first for using GitHub for most team members, they weren't always able to use this system and were forced to send emails. This was a mistake, as some of our code was lost during this process.

Overall, we believe we have been able to develop our skills in this process of creating our project and are motivated into doing something like this again.

Future Work

In this chapter we will explain what we have created and our future aspirations for this project.

Our initial aim for this project was to create a web-based application that had the features of a regular group chatting system with our own unique features. This app would be exclusively made for Goldsmith's students. It would take in information such as hobbies, interests and ethnicity and would recommend group chats that have the same or similar attributes. This approach meant we would need a system that would take in information and produce an appropriate answer (the group chat).

We started off with a template that had the prime aspects of a group chat. Our job was to improve this into making it similar to our concept. We started off by adding a home page where we showcased our premise to the user. This page had options to access web pages like messages and profile. We were happy with the home page idea we created but we were missing a few features from our concept. For example, we had to remove the "friends" tab due to the fact that you could not add any friends. You could create 1 on 1 rooms with someone, but you couldn't actually add them as a "friend". As stated, this was a concept we wanted from the very beginning of development, however due to our bad time management we couldn't have this as a priority as we wanted the bulk of our chat functioning. We believe this is a feature that can easily be added in the future as we have already created userIDs and usernames to link people to become friends.

Another feature, that we can add in the future is a separate settings menu. Like the friend's menu, we had to remove this due to other priorities when creating our app. We linked some of our settings features into the profile page. For example, our change password feature was originally meant to be in a separate settings folder but was in the end placed in the profile page. We believe this should be added in the future to our application.

Furthermore, the two-factor authentication system is also a system which we would really like to add to our app as in our MVP, security was one of our priorities. We wanted to create a system where users would have to verify emails, so the application doesn't get botted. This would have provided the app to be a much safer place as you knew only people who verify their email would have access to it.

However, the feature that we regret the most missing out on was our recommendation system. Since this was meant to be the biggest feature that would make our application unique from others, we really wanted to add it. But due to our bad time management and lack of knowledge in this area, we didn't know how to do it. We wanted our users to be recommended group chats based on their own interests but could not find a way into doing it. Since our development of our app had finished, we thought of a way that when the user enters key words that match with a group-chats name it could recommend it. We think this

is a possible way into adding a recommendations system and our motivated into completing this in the future. The system we have in place at the moment, is to create a group chat then sending a key to another user for them to have access to it.

We know that we have accomplished many things when creating our group chat, but we also know that we have over promised and under delivered in some areas. This is due to the lack of knowledge and underestimating the amount of work that was loaded on to us during this process. This is an element we now know, and we take into consider for future projects.

Teamwork

Overall, we believe we have worked to the best of our ability and know that we have produced an application that has its main features working. We are satisfied that we were able to work together as a team. We had some chemistry when performing some tasks and had some team dynamic. We were lucky everyone's strengths were in different areas within the team. This meant we could give out different tasks to different members of our team based on what they were good at and their strongest points.

We had zero problems in terms of “fights” or “problems”, and we were never frustrated with each other.

Having said this, we know that not everything was perfect. We weren’t always there for each other when something went wrong. This was very unprofessional, but this was caused to factors outside of the project, such as other work needing to be done or being not motivated during a time of need.

Throughout midterm and towards the end of term, it was hard to keep everyone in track and it became harder to know where everyone was at in the project timeline. We had some members that became very ill or had personal problems, where they could not work for a period. These hiccups through the process of creating our project became quite common, and other members in the team had to take over work from someone who couldn’t do it during that time. This was one of the main issues and reasons we couldn’t implement all our promised features into the app.

Since this was a sort of “first time” type of project, not all members knew the circumstances we were setting ourselves up to when we created our initial concept. For instance, we didn’t know how difficult it would be to implement a recommendation system, and we only found this out during the time of development.

We now know that we have over promised in terms of features being added and underperformed into adding them. We know we were affected by outside factors, but we also should’ve known from the start what we were putting ourselves through.

Overall, we were a team that synergised well, covered work from each other and tried our best into creating an application we are proud in. We think that with a little more time our application would’ve been much closer to a finished product.

8 – CONCLUSION:

Our greatest challenge was keeping track of time and completing all features that were promised. We delivered an application that did not meet all MVPs, but managed to implement some features that would give home into meeting them in the future. Saying, this we believe we have created an application that still meets the main function of a chat app and think it can easily be improved in the future.

We have missed out on features like recommendations system and friends list and think there are some features we can improve like the current group-chat system. We should've created a safer application with the two-factor authentication, but we could not implement that too.

We believe that the reasons this has happened is because time management and outside factors. We had some members that were not able to complete work during a period of time because of, for example, personal issues and illness. Due to this, some features are missing from our home page as we had to focus on implementing the main functions of the group chat system.

Including all our disadvantages, we believe throughout the rest of the development stage, we worked well as a team and developed an application that showed the main functionalities of a group chatting app.

One of the major factors our app did well, was user feedback. We created many small surveys during and at the end of the development stage and managed to add some of the features the users wanted, which created a a better system overall.

9 - User Guides

When opening our application, you will enter the home page. From the home page you will press “sign up”, this will take you to the register page. You may register there by entering name, password and details about yourself. After that you will have to login. Login with the same details as register, but only with username and password. Now you on the home page of the group chat app. From here you can change profile, see your rooms, join a room or logout of your account. Go to create room and create a room, then go to “My room” (on the side bar) and redirect to your room. You can see a room ID at the top of the screen, send this room ID to another user, and they can go to home page (on side bar) and enter the room ID. After going to “my room” then “redirect to room”, scroll down and there is a chat bar where you can type to your friend.

How to make code work on local system

Step 1: First download “Final Prototype.zip” from https://github.com/Simj2002/Chat-App-Project.git

Step 2: Download mySQL, create an account, then create a database called “goldsmeetups”.

Step 3: In the Final Prototype zip file, there is a “vn-room-chat.sql” file. Open that then in mysql, first do “USE goldsmeetups”

Step 4: Then start copying and pasting every section of SQL in “vn-room-chat.sql” into the mysql command line.

Step 5: Then download XAMPP. Locate where you downloaded XAMPP, enter folder, then enter “htdocs” and paste the code folder from our repository.

Step 6: Open the XAMPP control panel then press start next to “Apache”.

Step 7: Open a browser and type localhost/GoldsMeetUp/index.html

Step 8: You are now in our chat-app, and you are free to use it.

References

[1] https://www.nngroup.com/articles/ten-usability-heuristics/

[2] https://www.youtube.com/watch?v=Bv9Js3QLOLY

[3] https://chatengine.io/

[4] https://firebase.google.com/

[5] https://www.youtube.com/watch?v=MJzbJQLGehs

[6] Stream

[7] https://www.google.com/search?q=discord+ui&rlz=1C1CHBF_en-GBGB974GB974&sxsrf=APwXEdc0_qUAeU_MBZv2Xvt_eKGP1XbrnQ:1680463345748&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjw4Z_m9Yv-AhVOdcAKHWaPARQQ_AUoAXoECAEQAw&biw=1536&bih=714&dpr=1.25#imgrc=6e3ShHJSrVvlYM

[8] https://www.youtube.com/watch?v=4bt1sfviKCY

[9] https://authy.com/what-is-2fa/

[10] https://www.youtube.com/watch?v=diPDHvhUXHU

[11] https://elasticemail.com/

[12] https://bootcamp.berkeley.edu/resources/coding/learn-web-development/what-does-a-front-end-web-developer-do/

[13] https://www.telerik.com/blogs/top-5-reasons-websites-with-simple-design-are-better#:~:text=Simple%20Design%20Enhances%20User%20Experience&text=So%20it%20gives%20the%20user,skim%20content%2C%20according%20to%20Adobe.

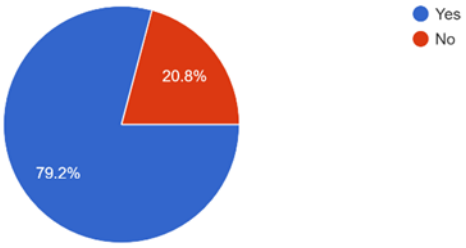
[14] https://www.henshaws.org.uk/top-tips-for-making-websites-and-apps-accessible-for-people-with-sight-loss/#:~:text=Websites%20and%20apps%20should%20be,as%20a%20basic%20human%20right.

[15] GitHub Repository: https://github.com/Simj2002/Chat-App-Project.git

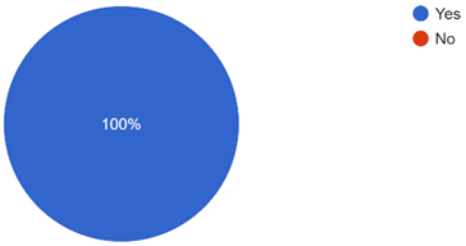
Appendix:

Initial Market Survey – Introduction

Do you struggle to meet new people at Goldsmiths?
24 responses



Do you think a concept like ours is something that is needed at Goldsmiths?
24 responses

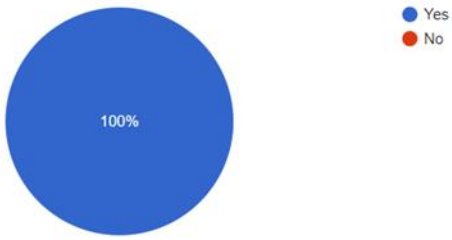


Is our concept unique or is there one already in place at Goldsmiths?
24 responses

Our application concept is to create an app for Goldsmiths students where they can create a personal profile page, and have the system recommend group chats or people who have similar interests, hobbies or languages as stated in your profile. You can then try socialising with people who you think you might end up being friends with. Is this app something you would use at Goldsmiths?

Copy

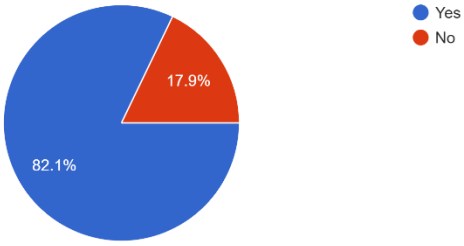
24 responses



Research Introduction Survey

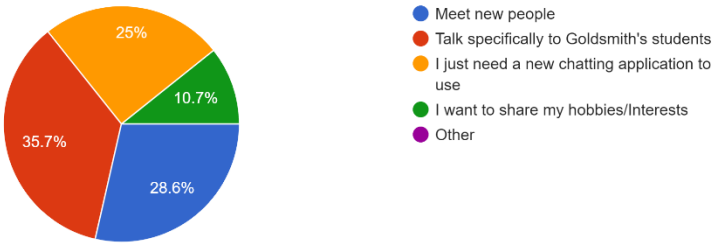
Does our application need two factor authentication?

28 responses



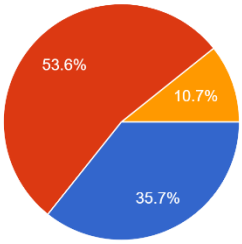
What is the main reason you want to use our application?

28 responses



Should all group chats be public or private?

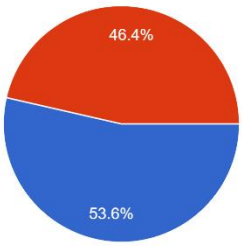
28 responses



- I want there to be a way for group chats to be hidden
- All group chats should be public
- All group chats should be private

How should people be added to group chats?

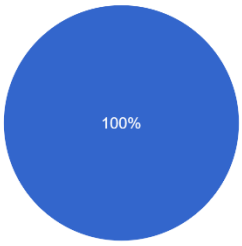
28 responses



- They can choose if they want to join
- They must be invited to the group chat

Do you attend Goldsmiths University?

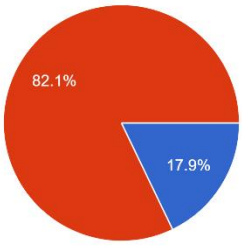
28 responses



- Yes
- No

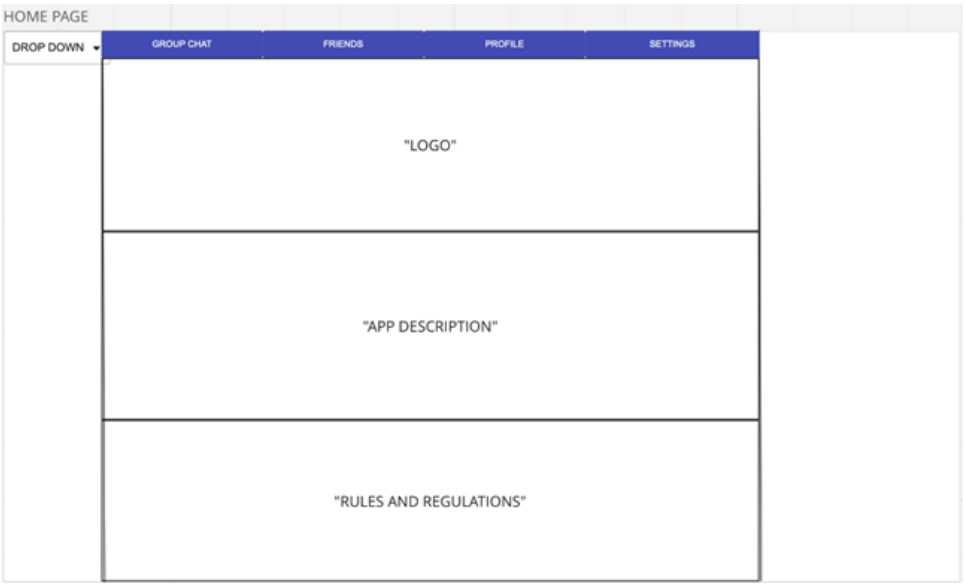
How should our application look like?

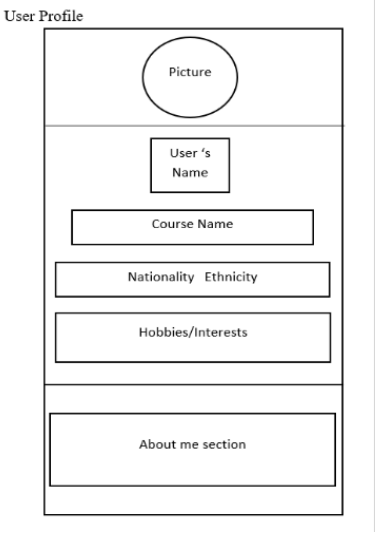
28 responses



- I want there to be a lot of different parts on the screen
- I want the UI to be simplistic

Wireframes

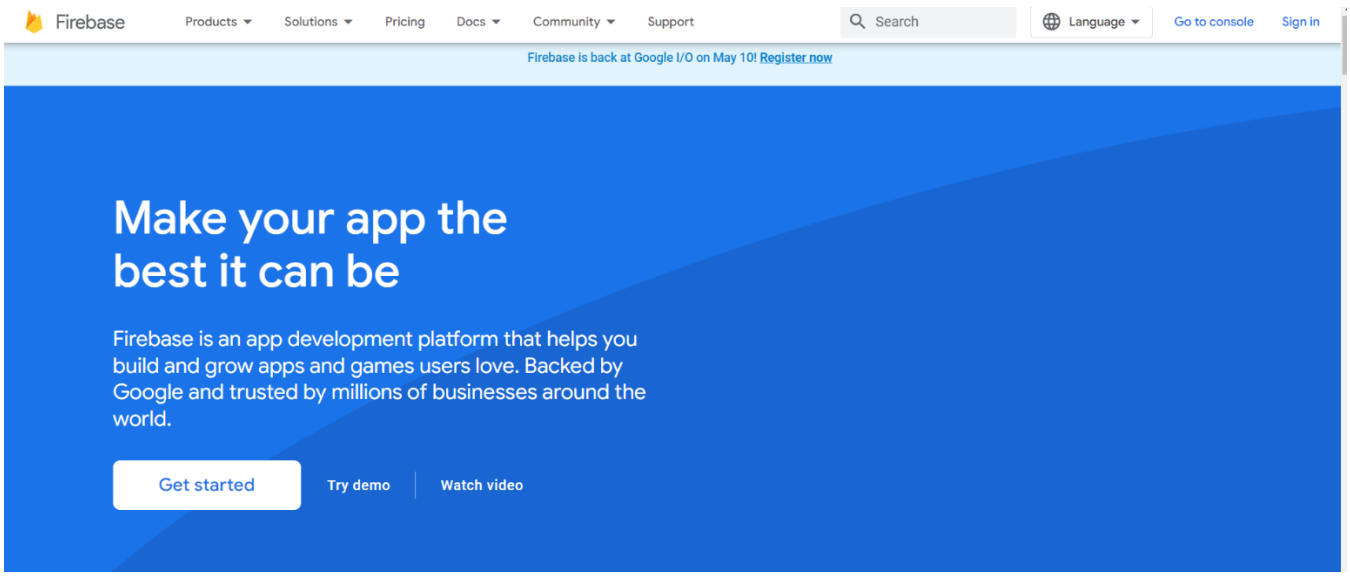


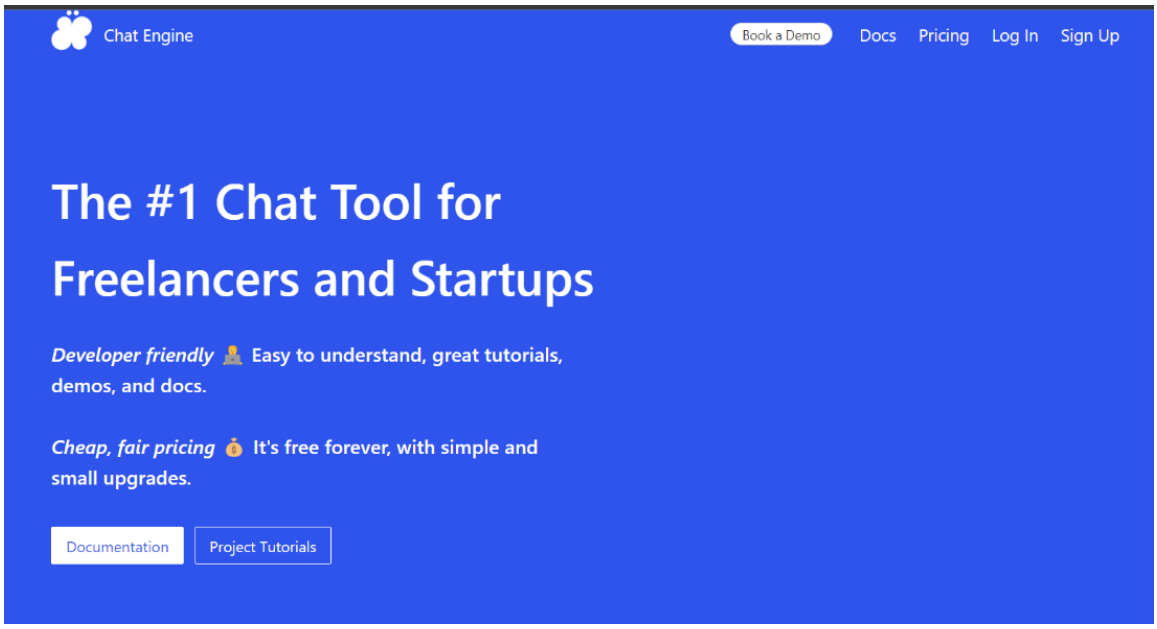


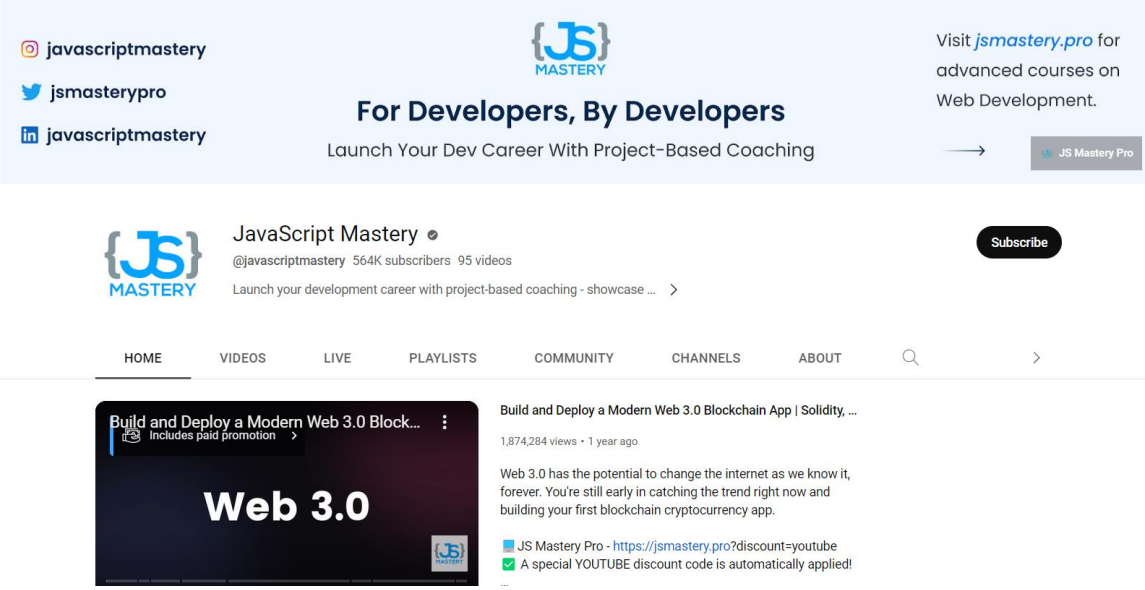
Home Page Survey

Group Chat Iteration 1 Images

These are the websites we used to create iteration 1 of group chat.

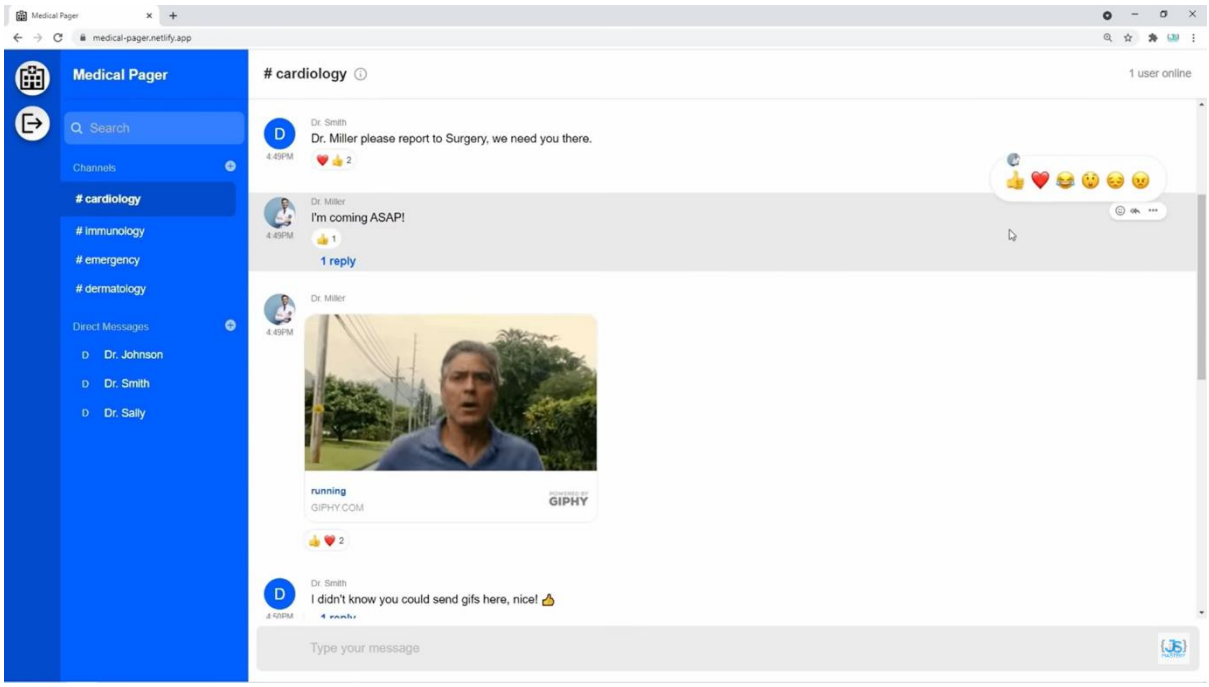




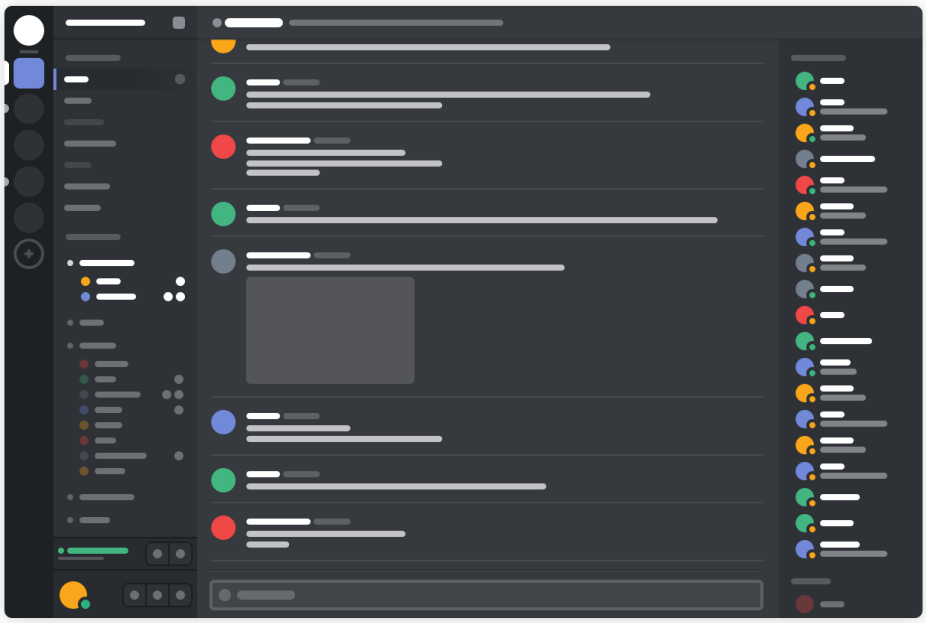


User Interface Comparison Discord Vs Our App

Our application user interface:



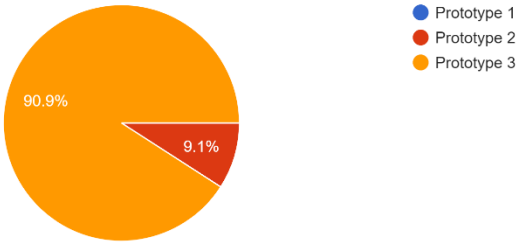
Discord user interface:



Group Chat Survey

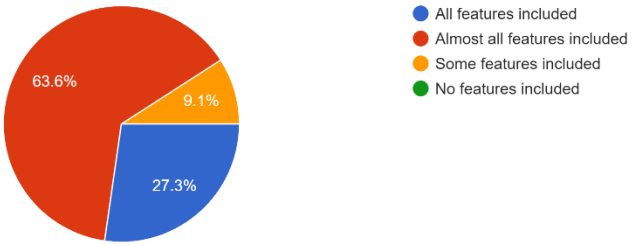
Which prototype has the best design and looks the best as a group chat app?

11 responses



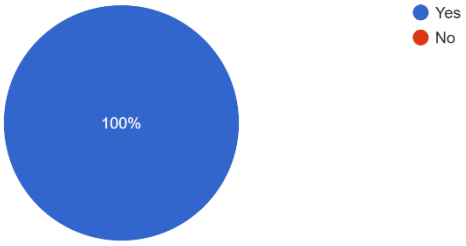
Are all features in the app that were promised in the concept included in the prototype?

11 responses



Would you use this group chat app?

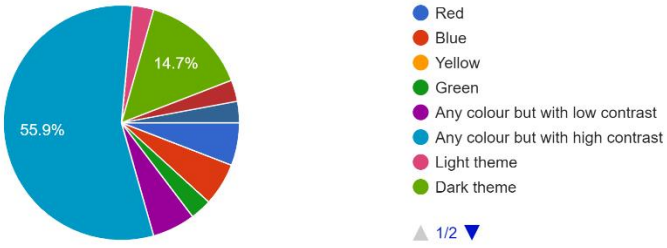
11 responses



Colour Scheme Survey

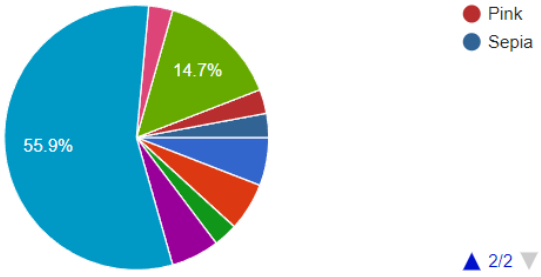
What colour scheme would you like our application to have?

34 responses

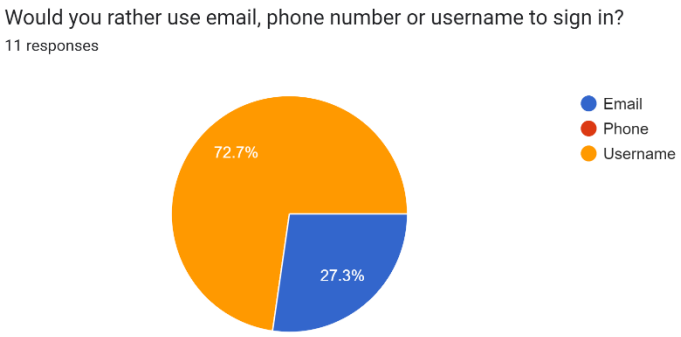


What colour scheme would you like our application to have?

34 responses

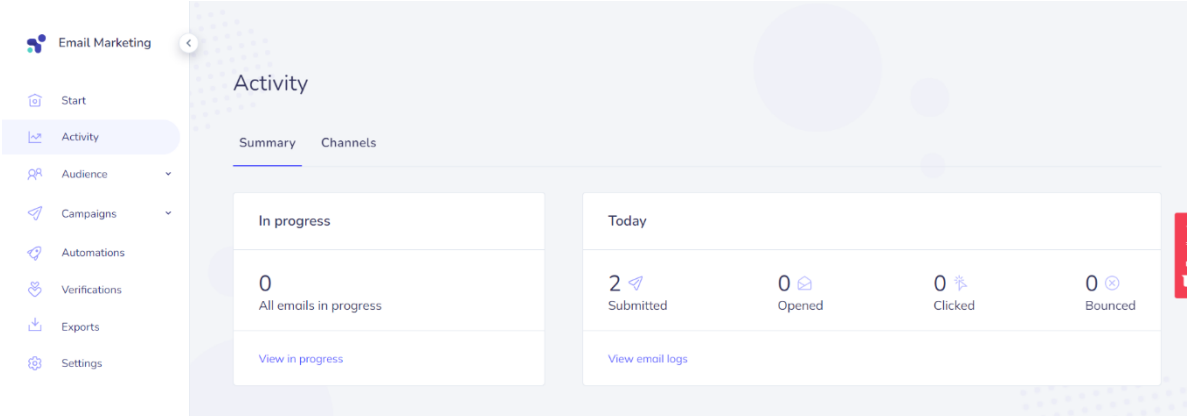


Login Survey

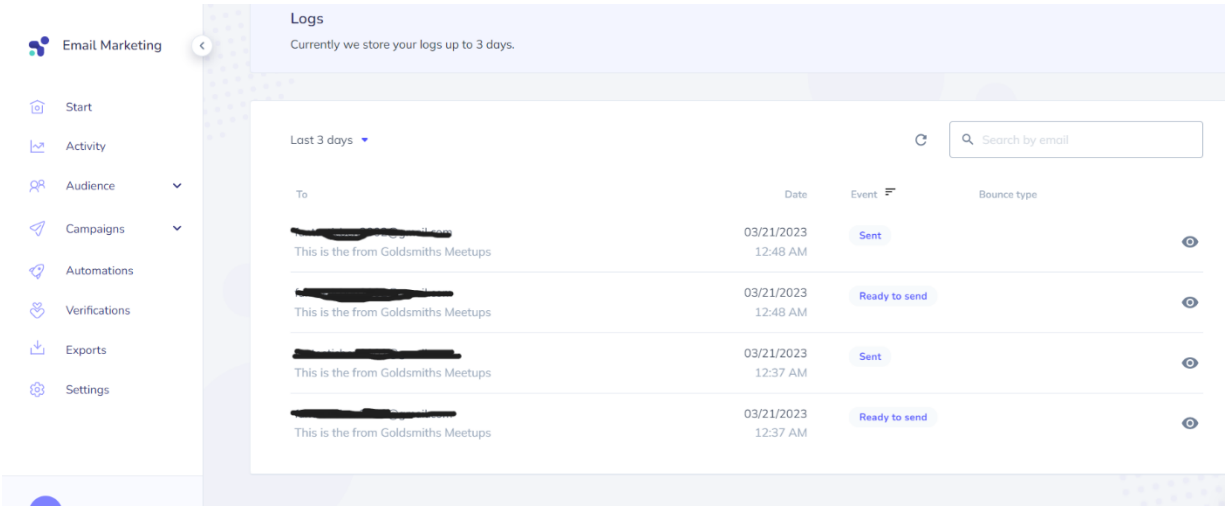


Elastic Email Interface

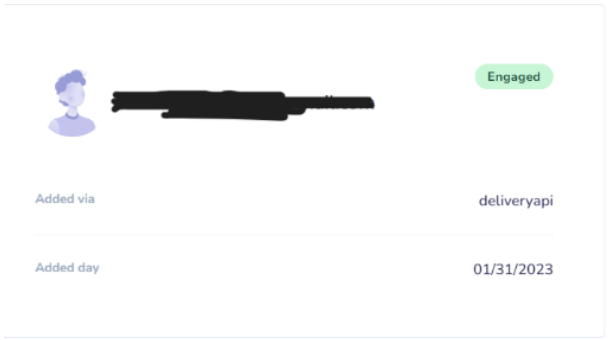
Shows activity of how many emails submitted:



Shows who the email has been sent to:

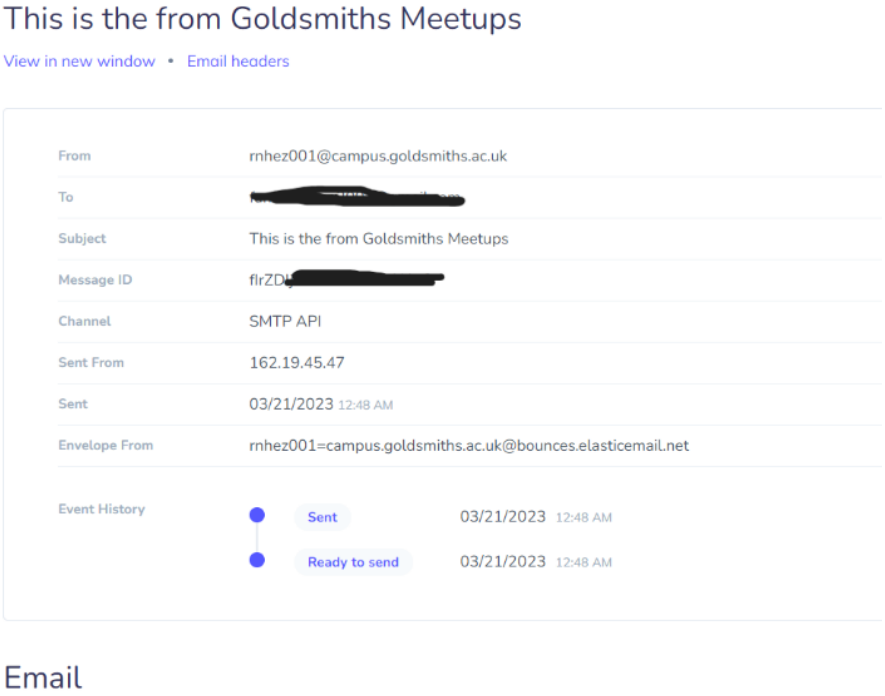


Elastic Email taking a record of when email was sent:



Checks whether user has engaged with the message, how many email were sent, opened and some basic information about the user:

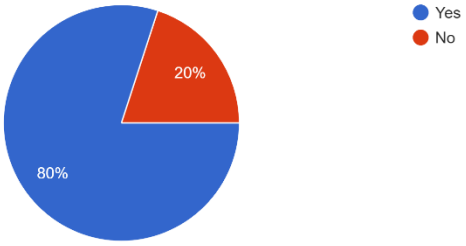
Keeps a history of each email:



Profile Page Survey

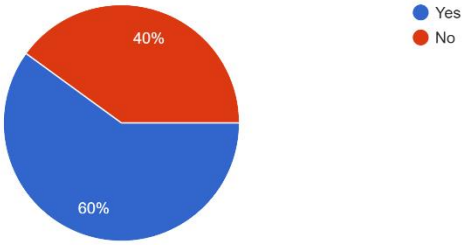
Do you like the layout of the profile page?

10 responses



Do you think the profile page needs anything added?

10 responses



Do you like the colour scheme of the profile page?

10 responses