Development of a Social media application using novel features to enhance trust between users.

Newcastle University
School of Computing
Name: Tharan Nareshbhai Patel
Student Number: 180643329
Supervisor: Dr. Ellis Solaiman
Word count: 14689 words

Abstract

With the ubiquitous nature of social media, the prevalence of distrust between users poses itself as an ever-growing problem. Popular social media applications do not give sufficient tools to gauge nor vote on content in regard to their trustworthiness. In creating a social media application which is transparent of the trust of its users and posts alike, we hope that users can be more mindful of the content they choose to consume on the platform.

Declaration

This dissertation is submitted to Newcastle University in accordance with the requirements of the degree of Master of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institute of learning. Except where specially acknowledged, it is the work of the author.

Acknowledgement

I would like to thank my project supervisor, Dr. Ellis Solaiman, for his guidance on this project. I would also like to express my gratitude to my parents, Deepika Patel and Naresh Patel, for their support and advice throughout the years which has been invaluable and made all my studies possible.

Nomenclature

FR – Functional Requirement
JSON – JavaScript Object Notation
JWT – JSON Web Token
NFR – Non-functional Requirement
SCSS – Syntactically Awesome Cascading Stylesheets
SDLC – Software Development Lifecycle
SQL – Structured Query Language
UI – User Interface
UX – User Experience

Table of Contents

1. Introduction *6*

1.1. About this project **6**

1.2. Project Objectives **6**

1.3. Project Methodology **7**

1.4. Outline **7**

2. Background Research and Technical Material *9*

2.1. Literature review **9**

2.1.1. Dangers of misinformation 9

2.1.2. Trust buttons 10

2.2. Similar systems **10**

2.2.1. Facebook 10

2.2.2. Twitter 12

2.2.3. Reddit 14

2.2.4. Similar systems overview 15

2.3. Technologies used **16**

3. Design and Implementation *17*

3.1. Requirements analysis **17**

3.1.1. Functional requirements 17

3.1.2. Non-functional requirements 18

3.2. Planning **18**

3.3. Design **19**

3.3.1. Website Design 19

3.4. Database **20**

3.4.1. Database tables 22

3.4.1.1. Users Table 22

3.4.1.2. Posts Table 23

3.4.1.3. Comments Table 23

3.4.1.4. Likes Table 24

3.4.1.5. Positive Trusts Table 24

3.4.1.6. Negative Trusts Table 25

3.4.1.7. Relationships Table 25

3.5. Application components and pages. **26**

3.5.1. Login and Registration 26

3.5.2. Home page 27

3.5.3. Share function 28

3.5.4. Posts 29

3.5.5. Profile page 32

3.5.5.1. Profile page (current user) 33

3.5.5.2. Profile page (other users) 34

3.5.6. Friend trust rating 35

3.5.7. Post trust Rating 35

3.5.8. Mutual friends 36

3.5.9. Navigation bar 37

3.5.10. Dark mode 38

3.5.11. Mobile Optimisation 39

3.6. Testing **40**

3.7. Summary **41**

4. Results and Evaluation. 42
4.1. Results 42
4.1.1. Evaluation of functional requirements 42
4.1.2. Evaluation of non-functional requirements 44
4.2. User surveys 45
4.2.1. Survey methodology 45
4.2.2. Survey results 47
4.2.2.1. User interface survey results 47
4.2.2.2. Trust features survey results 48
4.2.2.3. General survey results 50
4.2.2.4. Survey summary 50
4.3. Technology review 50
4.3.1. Frontend technologies 50
4.3.2. Backend technologies 51
4.4. Methodology review 51
5. Conclusion 52
5.1. Meeting objectives 52
5.2. Future work 53
6. References 54
7. Appendices 55
7.1. Figma Designs 55
7.1.1. Login and Registration page designs 55
7.1.2. Navigation bar designs 56
7.1.3. Homepage designs 57
7.2. Test List 58
7.3. Survey results 61

1. Introduction

1.1. About this project

Whilst thoroughly researched, mainstream social media networks still pose a major problem, which is the prevalence of misinformation and distrust between users. Trust is often tackled using a binary approach, such as the use of verification badges, with the user otherwise having to discern the trustworthiness of a user/post by taking the follower count or post engagement into account, which is a less than sufficient means of doing so.

As such, the main project aim is to resolve the aforementioned problem by producing a social media application that is much more transparent regarding its userbase and provides users of the platform with more information to gauge how much they should trust a user or post. In addition to this we aim to give users a multitude of tools to vote on the trustworthiness of content and users on the platform itself.

Trust escapes simple measurements as it is inherently too subjective to allow itself to be measured by universally reliable metrics [1]. In addition to this, the application of trust is a mental process and as such is unavailable to statistical measuring instruments. Knowing this, we aim to also develop a trust-based algorithm which is to be implemented as a means of measuring a trust level for users/posts within the application, which in turn will be used in tandem with other 'standard' features of the application.

1.2. Project Objectives

The objectives of the project are comprised of both technical and personal objectives which can be seen below.

- Conduct background research regarding the concerns of social media in terms of trust between users, as well as to identify common trends and techniques used in successful social media applications, particularly in terms of the user interface/user experience.
- Structure and plan the project by adopting an agile methodology.
- Plan and design the website using *Figma,* a website design tool.
- Learn the skills required to construct an interactive social media application involving the React.js library for front-end development, Node.js runtime environment and MySQL database for back-end development. Syntactically Awesome Cascading Style Sheets (SCSS) will be used for website styling.
- Design an algorithm that uses a variety of factors to calculate a fair and accurate trust rating for users, which will be used in many of the applications features.
- Design the website for use on both desktop and mobile devices.
- Gain proficiency using GitHub to provide version control for the development of the application.
- Evaluate the project using the set of requirements (FRs and NFRs) that are to be outlined prior to development, in addition to using the feedback provided by user surveys.

1.3. Project Methodology

An Agile methodology has been adopted to assist the planning and structure of this project primarily because it prioritises transparency and feedback during the development of the project. This means that the customer (in this case, the supervisor) is able to see what is happening often and easily, which allows for useful feedback throughout project development which can be implemented as needed. Agile succeeds at this due to its flexibility as opposed to the strict regimen of the Waterfall model.

Comparison	Agile	Waterfall
Lifecycle	Multiple, shorter SDLC's.	Single, long SDLC.
Planning	Planning is done for a very short term since the product is delivered in 2-4 weeks.	Planning takes a long time.
Requirements	All requirements are not initially clearly defined, instead they develop over the lifetime of the project.	All requirements are clearly defined before development begins.
Progress	Progress is reviewed once per day.	Progress is reviewed once a week.
Team	Teams are self-governing, more independent teams.	Teams governed by single authority.
Emphasis	Focuses on delivery of business value to consumer (the supervisor).	Focuses on implementing the requirements.
End Goal	Aims to deliver product features in a sprint to the consumer.	Aim is to deliver a complete product only once it has gone through all the phases.

Table 1. Main differences of Agile and Waterfall developmental processes.

1.4. Outline

This dissertation is comprised of 5 main chapters which cover the background research, development, evaluation and summarisation of the project.

1. Chapter one identifies the problem to be solved as part of this project, why it was worth solving and the main objectives that will be tackled to achieve this. The development methodology and scope of the work is also explained in this section.
2. Chapter two covers the background research undertaken which contextualises the work undertaken in this project. A comprehensive review of mainstream social media applications is done here in order to identify common features of a successful application, as well as areas which are perceived to be lacking, which the project will attempt to improve upon. Chapter 2 will also provide a brief description of the technologies used in the project.
3. Chapter three outlines the planning, design and implementation of the social media website as well as testing strategies using a high-level explanation.
4. Chapter four describes the results of the project as well as a detailed evaluation. This will include a post-development review of the requirements outlined prior to the development of the application as well as the results of the user surveys of the application. A review of the technologies and project methodology used will also feature here.

5. Chapter five is comprised of a summarisation of the project, what has been learned and what could be done in follow-up work. The references used throughout the project can also be found here.

2. Background Research and Technical Material

2.1. Literature review

2.1.1. Dangers of misinformation

As a result of an advancing society, there grows a stronger reliance on technology, and how it is used to interact with those around us. Unfortunately, due to the anonymous nature of the internet and more specifically, social media, this often gives way to the propagation of unreliable news and a dangerous spread of misinformation.

A significant example of the effects of the spread of misinformation is the case of the Myanmar security forces, which spread biased and untrue propaganda, using Facebook as a platform to do so, which led to violent action being taken against Rohingya Muslims in Myanmar, forcing more than 700,000 Rohingya Muslims fled the northern state of Rakhine. Many believe Facebook should be held accountable for not sufficiently controlling the spread of harmful anti-Rohingya Muslim content. The algorithms which Facebook employ to increase the visibility of certain posts had worked against the Rohingya Muslims, in this case an anti-hate initiative supported by Facebook, in which users can post anti-hate stickers (containing messages such as 'Think before you share') on posts that advocate violence, meant that the amount of people that saw these posts actually increased as the algorithm saw this use of stickers as an indication users enjoyed the post, thus increasing the visibility of the harmful post [2].

Figure 1. Artwork commissioned for publication of research, looking at the Rohingya right to remedy from Facebook [2].

As demonstrated, it is insufficient to rely on the social media provider to provide administration to posts and delete them prior to their propagation throughout the website. We believe that the dangerous spread of this misinformation could have been significantly inhibited by giving users of the platform more tools to not only accurately gauge the trustworthiness of the content they consume but also by giving them tools to rate the trustworthiness of content/users directly. As such we theorise that this would allow for a decentralised verification system, instead of one that relies on a relatively small number of individuals compared to the userbase.

2.1.2. Trust buttons

Recent studies have theorised that giving users the option to react to posts with not only 'likes' and 'dislikes' but also with trust and distrust buttons may have beneficial effects on the content found on social media platforms in regard to trust [3]. This implementation not only prompts users to actively think about the trustworthiness of a post, leaving them less perceptible to consuming disingenuous content, but also allows them to participate in the moderation of content, helping other users on the platform to understand the trustworthiness of posts. We believe that this model could use this trust data and incorporate it into an algorithm, potentially filtering out content that is below a certain level of trust automatically, before it is able to propagate throughout the network and influence individuals.

2.2. Similar systems

As part of the research for this project, the decision was made to analyse the most popular social networks used as of the date of writing. The intention of this analysis is to not only recognise common themes that help create a successful website in terms of design, but also how the platform attempts to incorporates aspects of trust, thus the analysis will be separated accordingly. As one of the main aims of the application in this project was content sharing, we opted to focus our efforts on the analysis of social media websites where the sharing of content was the primary function, as opposed to, for example, messaging (such as WhatsApp). As a result, the 3 main websites studied can be seen below.

1. Facebook (3 billion users)
2. Twitter (450 million users)
3. Reddit (430 million users)

2.2.1. Facebook

With Facebook being the largest social media platform in the world, with 59% of *all* social media users owning a Facebook account, it was logical to analyse this social media application first.

Design

Facebook is a feature rich social media that has content sharing capability, messaging, marketplace, short form videos (reels) and much more, however the smart UX and UI design make it such that a user is not overloaded with information when navigating the website [4]. The website allows users to comment, like and share content that features on the main feed to others, allowing for a highly interactive social media.

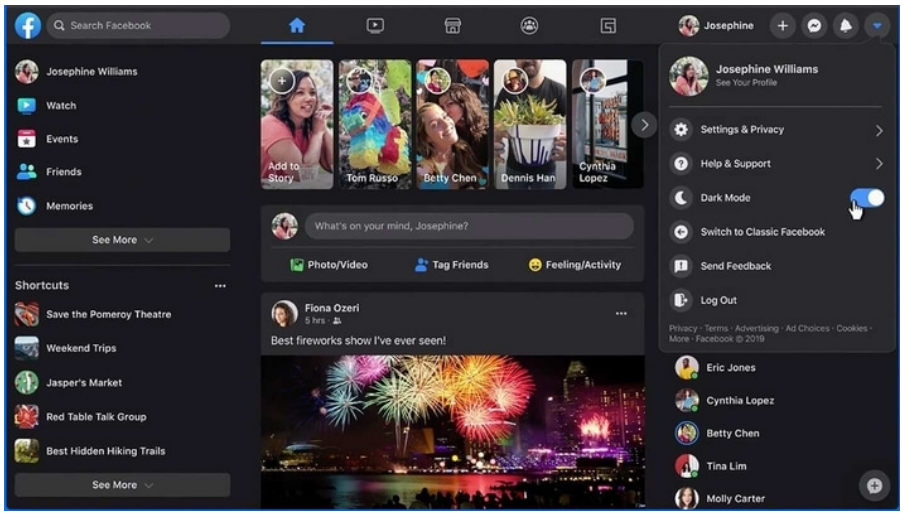


Figure 2. Facebook Homepage.

Facebook appears to use an agenda-driven design philosophy, wherein buttons which lead to sections of the app they want to be used to most, are placed in the most prominent positions. They are able to do this as they have sufficient user retention through their immense market foothold. This differs from user driven design where the priority is to make user experience the easiest.

An interesting thing to note is that the desktop UI is similar to that of a zoomed in mobile viewport, featuring the main feed and a navigation bar at the bottom (to improve the reachability of commonly used buttons). This can likely be attributed to the high proportion of users that visit the website on their mobile device, who would appreciate a similar experience due to the reduction in mental strain that would otherwise be required to learn a different interface.

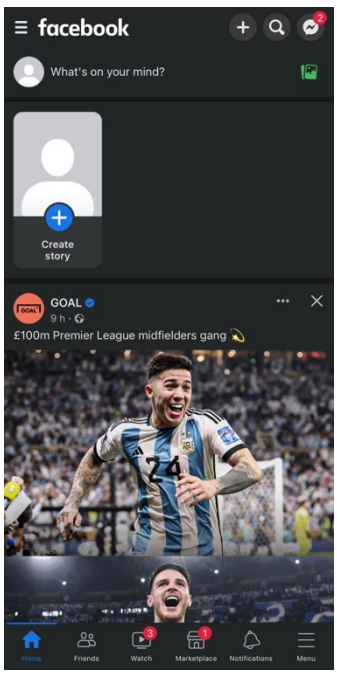


Figure 3. Facebook Homepage (mobile).

There is a calculated use of whitespace, differentiating the sections into a three-column arrangement that again relieves the mental strain on the user by preventing *content bombardment* [5].

Trust

When looking at posts it can be said that there are two main metrics which can be used to gauge the trustworthiness of the content within, this being the presence of a verification checkmark on the posting user's name and the amount of engagement with the post (likes, comments and shares).

The process for obtaining a verification checkmark is relatively long and Facebook states that users must be well known enough that it is in the public's interest to verify them. This means that this method of gauging trust simply cannot apply to the masses of users that share content on Facebook and so instead users must rely more so on the amount of engagement with a post to measure its trustworthiness. However, as the main, most accessible way to quickly engage with a post is to like the post (without an option to dislike readily available), we question whether users can properly use the like function to measure a posts authenticity.

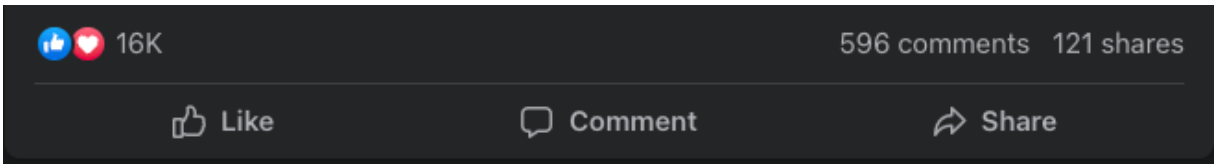


Figure 4. The engagement metrics and buttons available to the user.

2.2.2. Twitter

Twitter is a social networking site whose premise is to create highly skimmable content through 'tweets', which is done by the imposed 140-character limit per tweet [6]. This concept stimulates consumers of the content without boring them with overtly long posts. Users of the network are also able to directly message other users.

Design

Similar to Facebook, Twitter uses a 3-column arrangement to organise its homepage for all the same reasons provided before. However, it uses the left column to act as the primary navigation section, perhaps to provide increased clarity to the user of where each button will take them, due to the combination of an icon and accompanying icon description.

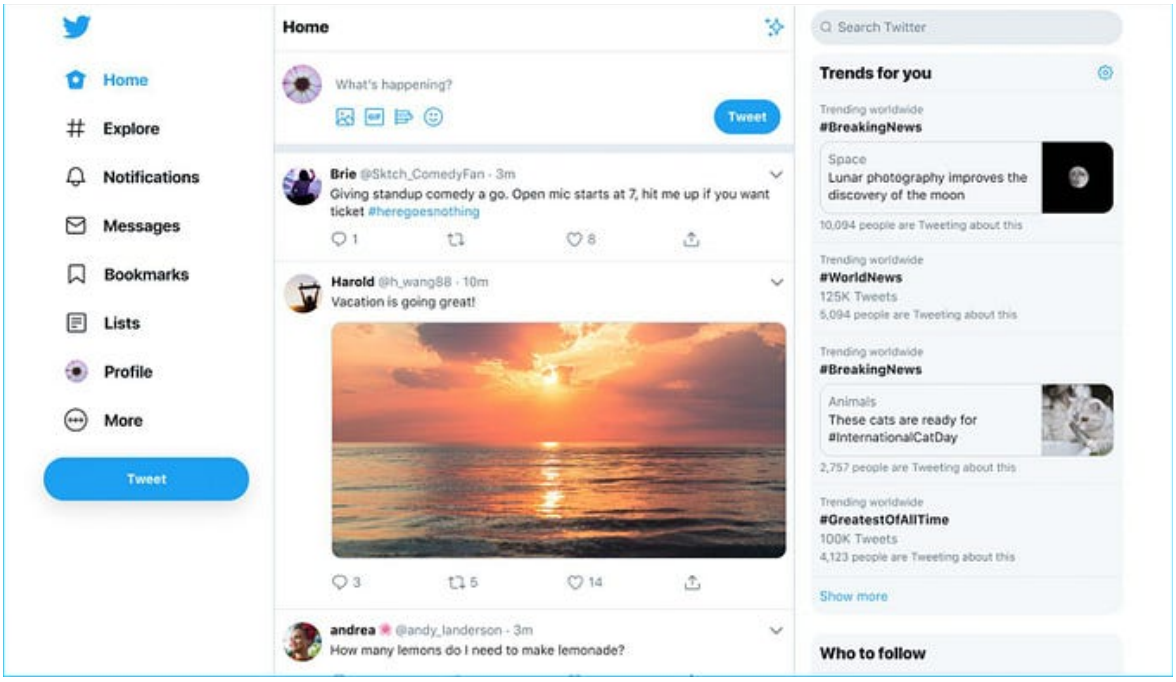


Figure 5. Twitter Homepage.

The primary accent colour, blue, is used sparingly, so that the importance of coloured buttons is accentuated, these uses can be seen on the 'tweet' button, hashtags (non-spaced words/phrases that can be clicked on to see other posts with the same hashtag) and for the currently visited page (Home, in the case of *figure 5*). This sparing use of colour is done so that the user can quickly identify elements of the website in which they would typically use the most, thus reducing mental fatigue as the user does not have to expend much time or effort looking for said element.

The main feed is again, similar to the viewport of that of the mobile interface, likely for the reasons provided in the *Facebook* section.

Trust

Again, Twitter suffers from the same lack of clearly defining trust metrics as Facebook, with users again only being able to gauge a posts authenticity using the amount of engagement of a tweet and the presence of a verification checkmark for the poster (which can now be simply bought using the platforms *Twitter Blue* subscription service, with significantly less requirements compared to Facebook). When Twitter is so widely used as source of factual information, can we consider this as good enough?

Figure 6. Engagement buttons for a Tweet, from left to right; comment, retweet, like, save, share.

In a publication regarding information credibility on Twitter, which analysed the credibility of tweets over a 2-month span using various computational techniques, it was stated that 'Users online lack the clues that they have in real life to assess the credibility of the information to which they are exposed, which is more evident in the case of inexperienced users, who can be

easily mislead by unreliable information. It is critical to provide tools to validate the credibility of online information' [7]. This conclusion reinforces our argument that Twitter provides an insufficient level of tools for users to gauge trust for posts and users alike.

2.2.3. Reddit

Reddit is a network of communities, or forums, where people are able to share news and content as well as comment on other people's posts [8, 9]. However, compared to the previous examples wherein users use their real name and a picture of themselves, users on Reddit generally do not relate their profile to any real-world identifiers. Again, users are capable of directly messaging each other.

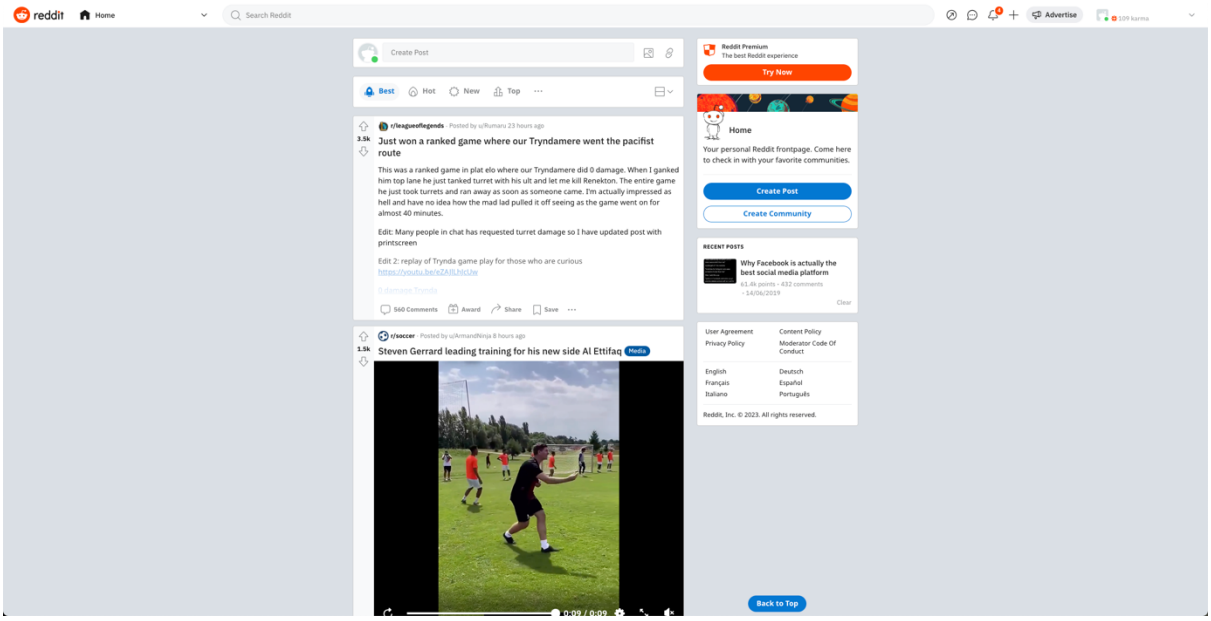


Figure 7. Reddit homepage.

Design

Surprisingly, reddit strays away from the common three-column arrangement that competitors use, instead opting for a two-column layout, with the secondary column being minimal and unintrusive. We believe the intention of doing so is to stress importance of the main feed to visiting users, as the platform is much more focused on having users engage with posts instead of other aspects such as a marketplace or direct messaging.

The navigation bar features a large search-bar, as that is typically the most used function that features on the navigation bar for this application.

As with the other examples, Reddit uses a significant amount of white space, to emphasise a priority on the main feed. However, the feed width is slightly larger this time, which is likely to encourage users to spend more time interacting with singular posts rather than quickly consuming the content and moving on.

Trust

In terms of engagement with a post, a user is able to leave upvotes/downvotes (denoted as the 'karma' system) and comments. Whilst still better than previous examples in terms of implementing a rating system which incorporates negative ratings, as opposed to solely positive ones, Reddit still suffers from the problem of not being able to explicitly rate a post or user based on trust.

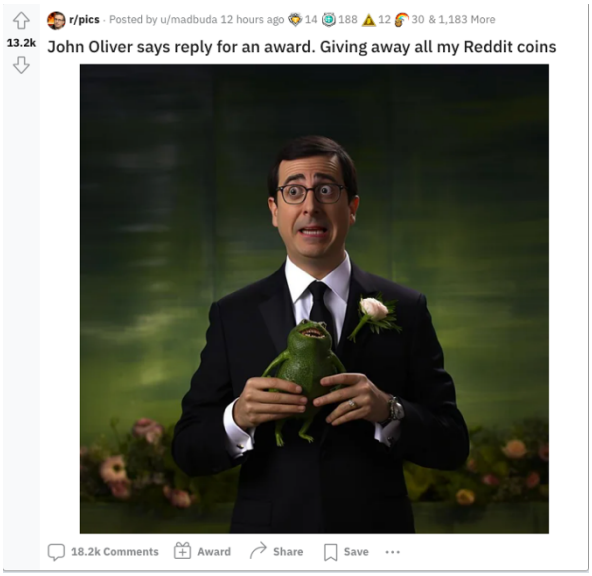


Figure 8. A reddit post which implements the upvote system (top left).

2.2.4. Similar systems overview

Design

All websites study generally stick to one colour, usually the colour of the company logo, with all other colours being grayscale. This design principle is intentional, as it allows the user to feel a sense of harmony when interacting with the website, which can be defined as a pleasing arrangement of parts. In contrast, when something is not harmonious it is either boring (lacks colour), where the user rejects engaging with the content as it is under stimulating or is chaotic (excessive use of colour and lacks arrangement), in which the viewer feels discomfort in viewing. The human brain simply rejects that which it is unable to organise and cannot understand, inevitably leading to less user engagement. This is likely why all websites analysed follow the same design principle of creating simplistic engaging interfaces [10].

Trust

This is an area in which all websites significantly lacked. When looking at the main 'posts' of each website, there were no tools a user could see to directly measure the trustworthiness, apart from a 'report post' option, which serves as quite a binary way to 'rate' trust of a post. Such a feature is also somewhat ineffective, considering the sheer speed at which dangerous content can propagate within a network before it is deleted.

Platform	Trust features	UI Features
Facebook	Low number of features - A verification checkmark is used for authenticating high-profile users, otherwise lacking in features here.	• 3-column layout • Blue-white colour scheme • Bright colours used sparingly. • Top navigation bar • Desktop layout is similar to mobile layout.
Twitter	Low number of features - Again, a verification checkmark is used for authenticating high-profile users, otherwise lacking in features here.	• 3-column layout • Blue-white colour scheme • Bright colours used sparingly. • Left navigation bar
Reddit	Slightly more features but still lacking -Upvote/downvote 'karma' system, verification checkmark for high-profile users.	• 2-column layout • Orange-white colour scheme • Bright colours used sparingly. • Top navigation bar

Table 2. Summary of system comparison.

2.3. Technologies used

For the initial design of the website, an online tool, *Figma,* was used in order to quickly create designs that could be reviewed by the project supervisor. A useful feature of Figma was that it provided CSS styling properties for elements within the created designs, deeming it very useful when it came to actually implemented the designs using SCSS.

Due to the relatively short development lifecycle of this project, we opted to use technologies that we were familiar with, those being React.js and MySQL, whilst also incorporating unfamiliar technologies, this primarily being Node.js. This approach gives a balance of being able to accurately predict development times whilst also providing valuable practical experience with an unfamiliar language.

React.js was also chosen due to its rich ecosystem of libraries and tools, which the developer made extensive use of throughout this project.

During the learning phase of the project, many tutorials were used to gain a sufficient understanding of the technologies used in the project. As a result, the decision was made to use some of the components created in these tutorials within the project itself. The main tutorial which the developer took inspiration from was called *'social-media-ui'* by *Lamadev* [11].

3. Design and Implementation

The purpose of the social media application is to provide a proof of concept of a social media that successfully incorporates features that enhance the measurement of trust between users. This section aims to demonstrate the design and implementation process that took place in order to achieve this.

3.1. Requirements analysis

A requirements analysis is composed of two primary items: functional and non-functional requirements (FRs and NFRs respectively). The use of a requirements analysis allows us to define user needs early on in the development process, assisting the planning of the project, especially when using an Agile methodology [12]. The priority of both FRs and NFRs will be defined using a selection of 4 options: Required (R), High (H), Medium (M) and Low (L).

3.1.1. Functional requirements

Functional requirements exist to provide a description of the service that the software *must* offer. It is critical that FRs are met to ensure the expected basic functionality of the software and help us to capture the intended behaviour of the system.

Requirement description	**Priority: R, H, M, L**
1. The application should require users to register an account.	R
2. The user should be able to update their profile information.	R
3. The application should allow users to be able to share content, including images.	R
4. The application should allow users to 'follow' each other, which causes the followed users posts to appear to the following users home feed.	R
5. The application should allow users to 'like' posts.	R
6. The application should allow users to comment on posts.	R
7. The user must be able to rate posts in terms of its trust/reliability.	R
8. Users should be able to delete their own posts.	R
9. The application should allow users to directly message each other.	M
10. The user should be able to search for other users using their name.	R
11. The user must be able to directly rate other users based on trust.	R
12. The application must integrate a trust-ranking feature that calculates the level of 'trust' one should have of other users using an algorithm that is based on a variety of parameters.	R
13. The application suggests friends based on trust rating.	R
14. The user should be able to delete their account.	H
15. The user must be able to log out of the application.	R

Table 3. Functional requirements.

3.1.2. Non-functional requirements

Non-functional requirements specify the quality attribute of the software, describing how the system should perform rather than simply what it should do. Meeting these requirements ensures that users are not left unsatisfied and generally provides a more pleasant experience, even if the core functionality of the software is not changed.

Requirement description	Priority: R, H, M, L
1. The application should feature a dark mode.	H
2. The app should be optimised for mobile devices using a responsive design that is able to accommodate a wide range of screen sizes.	R
3. The application will be secure and sensitive user information (such as passwords) be salted and hashed when stored in the database.	R
4. The application complies with GDPR regulations.	M
5. The application is available on most commonly used browsers (Chrome, IE, Firefox, Safari).	R
6. The application should feature a colour theme that assists navigation of the application.	R
7. The application should make use of animations to improve the user experience.	H
8. The application should make use of established design principles to provide a clear and intuitive user experience.	R
9. The application is relatively performant to provide an optimal user experience.	R
10. Development will involve the use of version control systems to ensure there is a recent backup of a working version of the application.	H
11. The application features a usage guide.	L

Table 4. Non-functional requirements

3.2. Planning

Using the requirements analysis outlined in the previous chapter, we were able to easily implement each requirement into a SCRUM management software, known as *Flying Donut* [13]. In doing so, progress could be clearly tracked as well as allowing us to identify remaining tasks and their expected completion time, which strongly facilitated the time management aspect of the project. This also helped in tracking any bugs that arise during development.

A Gantt chart was used to plan and schedule the project as a whole, including report writing, presentation planning and more, as opposed to solely the technical/programming aspect of the project (which was covered by the use of Flying Donut).

In addition to this, a GitHub repository was created to store versions of the application remotely, to ensure an up-to-date version of the application can be found online, should anything happen to the device the application code is stored on locally [14]. The use of GitHub also will allow the developer to rollback to previous versions of the application, if required.

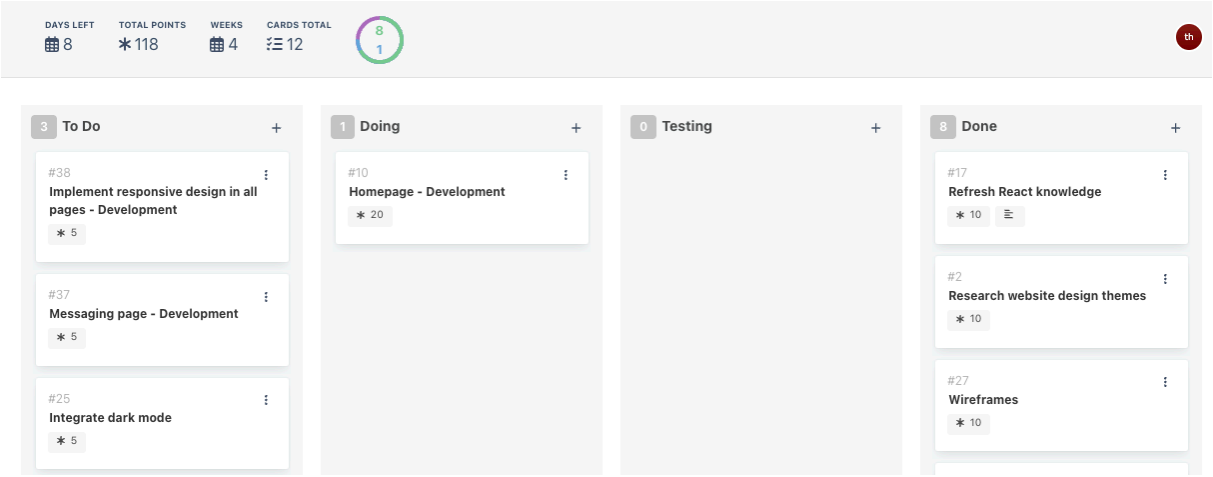


Figure 9. The development progress board for Frontend development. Note that each 'point' (indicated by an asterisk) corresponds to an hour of time.

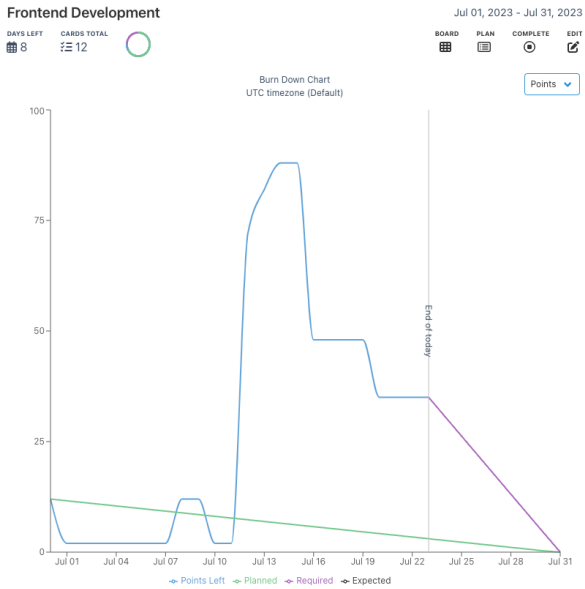


Figure 10. The progress graph for frontend development. A rise indicates the addition of tasks whereas a drop indicates the completion of tasks.

3.3. Design

It is important to provide a design of the software prior to implementation so that the developer can comprehend and thus better satisfy the objectives laid out in the project. This increases efficiency as the developer can solely focus on implementing the design given without having to think too much about if/how the design works.

3.3.1. Website Design

Prior to implementation, the decision was made to construct prototype website designs and website components using *Figma*, in order to have a visual example of exactly how the desired website should look, without having to implement prototype designs using SCSS,

which would otherwise significantly slow development time [15]. Figma also benefits the developer by quickly supplying the user with the hex colour codes used in the design and specific CSS properties used for the elements within the design (such as border-radius, font-size and more).

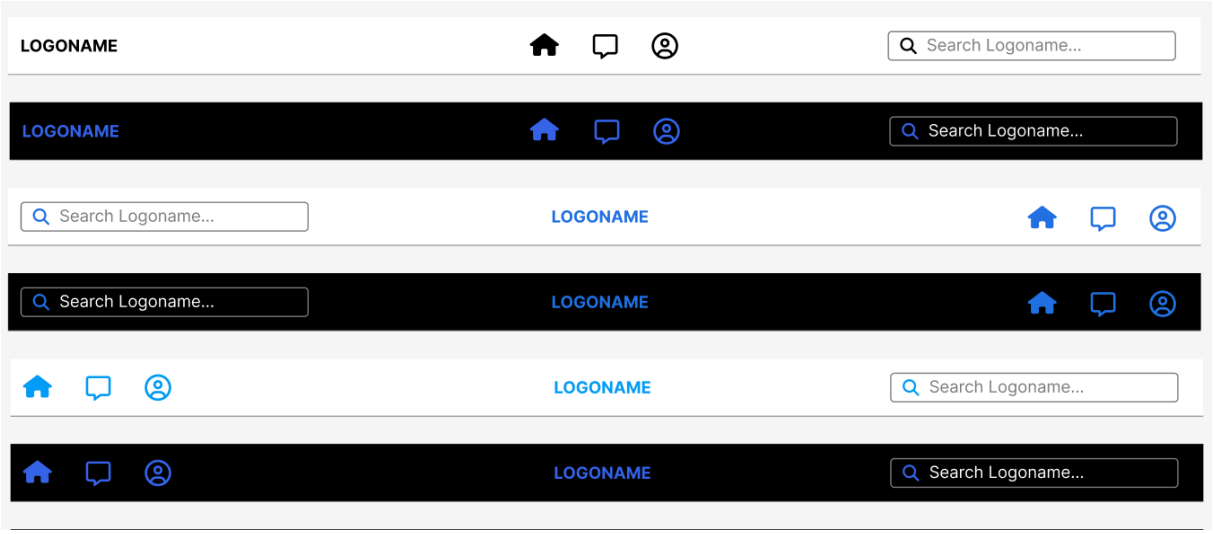


Figure 11. Navigation bar design ideas in Figma (dark and light mode).

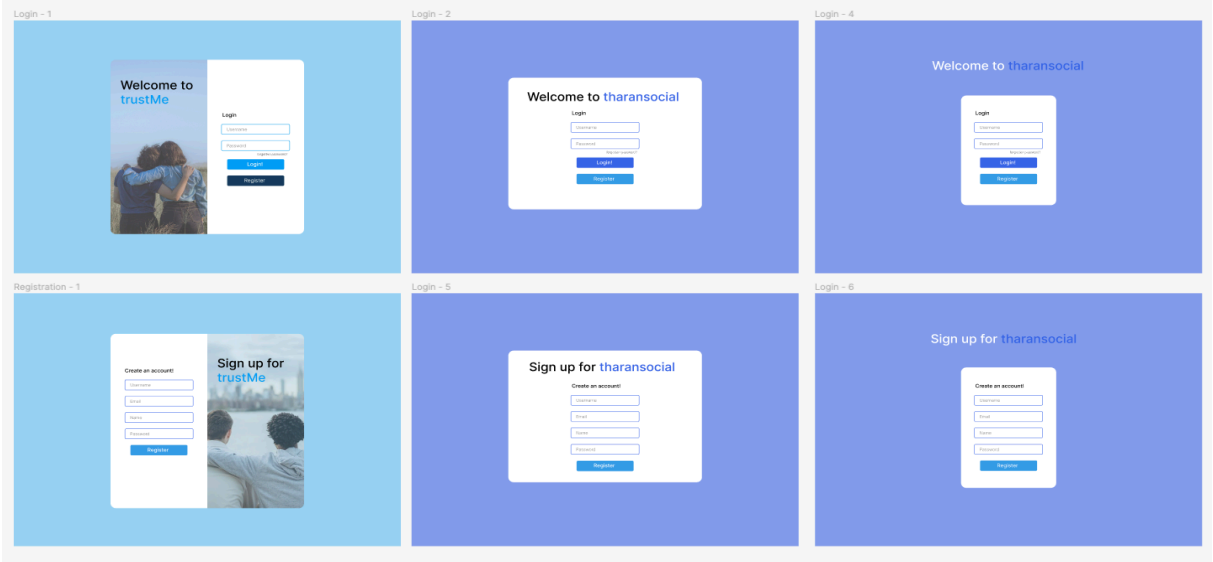


Figure 12. Login and Registration page design ideas.

3.4. Database

The database management system (DBMS) chosen for this project was MySQL, due to the developer's previous experience with the technology. It is also a suitable choice for a social media platform due to being a relational database, which allows for efficient modelling and querying of data with complex relationships, which our data will have.

An Entity-Relationship diagram has been used for the initial planning and structure of the database which can be seen below. This provides a visual representation of the data model, allowing for an easier understanding of database's structure, entities, attributes, and relationships.

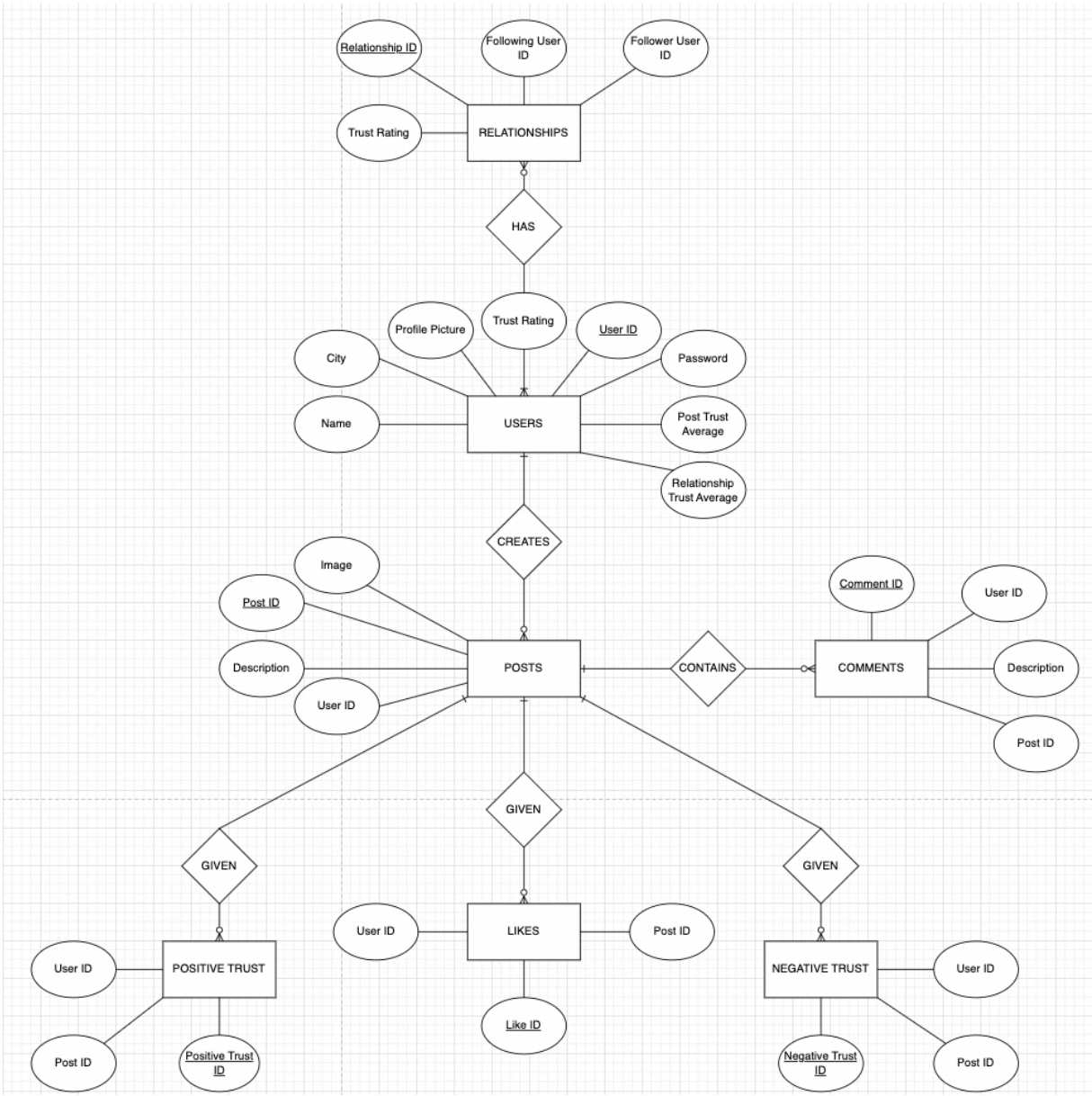


Figure 13. Entity-Relationship diagram for the database, created using Draw.io [16].

3.4.1. Database tables

This section aims to outline the purpose of each table (and its corresponding fields) used in the application database.

3.4.1.1. Users Table

The users table stores direct profile information regarding all users of the application, including login credentials, display information and average trust data about the users posts and relationships.

id	username	name	password	profilePic	city	postTrustSum	relationshipTrustS...
1	test1	test1	$2a$10$JuwmeNGQ4EpQp0IFP9I7d.F/UN2ftp4...	1691701084032IMG_6142.jpeg		50	65
2	test2	test2	$2a$10$dkP00Rfr2h04arvIZaMapuIU1GjqXYIs1...	defaultProfilePic.avif	NULL	100	63
3	test3	test3	$2a$10$TmU.jBN9tKb0fjJfAMJFCusUXaMwxN...	defaultProfilePic.avif	Newcastle	0	60
4	test4	test4	$2a$10$vDV8M//l9ZurduYTXDSP6eefYborfUY...	defaultProfilePic.avif	Manchester	0	38
5	test5	test5	$2a$10$ZUBQDjCL9ViRQz5ObPMxme2IW9H....	16917010514788b1dc73b-2832-444f-9c1b-25e...	NULL	0	70
6	test6	test6	$2a$10$wmLL.H84JpcySZVL3Eo/ieH/4BlK2/9U...	defaultProfilePic.avif	NULL	0	65
8	test8	test8	$2a$10$SKeRMEozAc2d/5CkN9ICkuWpkk8iFA...	defaultProfilePic.avif	NULL	0	43
9	test9	test9	$2a$10$tQSQ7/CrW2GnCk2gV0IYNOk8D9ZX...	defaultProfilePic.avif	NULL	0	0
10	test10	test10	$2a$10$uns3rCJ/Ro4i1QCIfKrswOqiCCUiRhef5...	defaultProfilePic.avif	NULL	0	0
11	test11	test11	$2a$10$Kk5mLw9HLID6kFRO2nCyCuVHXzfdv...	defaultProfilePic.avif	NULL	0	30

Figure 14. Users table.

Field	Required?	Other notes
id	Yes	The id of the user. Auto-increments to guarantee uniqueness of the automatically generated value.
username	Yes	The username which is used to log into the application. This value is unique to prevent the chance of users logging in to other users accounts which have the same username.
name	Yes	Acts as the display name, which will be seen by other users.
password	Yes	The user's password which has been hashed and salted, so that the password is not stored in plain text in the database, thus helping secure the password information.
profilePic	No	Default value is 'defaultProfilePic.avif' which corresponds to a generic profile image within the application public folder, to be used for the user until they update their profile picture.
city	No	Represents the city/country the user is from. Default value is 'NULL', however if the user wishes to remove the city the field value will be an empty string in the database.
postTrustSum	No	The weighted average value of trust for all the users posts. Default value is '0'.
relationshipTrustSum	No	The average trust the user's followers have attributed to the user. Default value is '0'.

Table 5. Fields used in the users table.

3.4.1.2. Posts Table

The posts table stores information about all the posts made on the application.

id	desc	img	createdAt	userId	postTrust	trustVoteCo...
229	Look at this view!	1691700947489IMG_5229.jpeg	2023-08-10 21:55:47	5	100	1
231	Hello this is test two's post!		2023-08-10 22:03:34	2	100	1
235	Wow look at this!	1691772713006b8886d18-1f39-4732-a3a0-5b7...	2023-08-11 17:51:53	1	0	1
237	How is everyone feeling today!		2023-08-11 17:53:16	1	100	1

Figure 15. Posts table.

Field	Required?	Other notes
id	Yes	The id of the post. Auto-increments to guarantee uniqueness of the automatically generated value.
desc	No	The textual description of the post.
img	No	The image of the post. Stores a string URL which corresponds to the URL of an image stored in the application public folder.
createdAt	No	The date and time at which the post was created, stored using the DATETIME format.
userId	Yes	The id of the user which created the post.
postTrust	No	The calculated trust for the post, default value is 'NULL'.
trustVoteCount	No	The amount of trust votes for the post, default value is 'NULL'.

Table 6. Fields used in the posts table.

3.4.1.3. Comments Table

The comments table stores the information about all comments made on the application.

id	desc	userId	postId
67	This is a comment.	1	231
68	This is another comment.	5	231

Figure 16. Comments table.

Field	Required?	Other notes
id	Yes	The id of the comment. Auto-increments to guarantee uniqueness of the automatically generated value.
desc	Yes	The description of the comment.
userId	Yes	The id of the user which created the comment.
postId	Yes	The id of the post which the comment belongs to.

Table 7. Fields used in the comments table.

3.4.1.4. Likes Table

id	userId	postId
299	3	238
300	3	231
301	1	231

Figure 17. Likes table.

The likes table stores all information regarding likes that have been given to posts on the application. The table has been designed such that each individual like is attributed its own record within the likes table. A consideration was made to have a like counter be a part of the posts table, however this was quickly rejected, as tracking which users have liked which posts was considerably harder, in such an approach. Important calculations such as the summation of a posts total likes is also easy in the approach taken, by simply finding the total amount of records which have the same post id.

Field	Required?	Other notes
id	Yes	The id of the like. Auto-increments to guarantee uniqueness of the automatically generated value.
userId	Yes	The id of the user which liked the post.
postId	Yes	The id of the post which has been liked.

Table 8. Fields used in the likes table.

3.4.1.5. Positive Trusts Table

The positive trusts table utilises the same implementation that was used for the likes table.

id	userId	postId
879	1	229
880	2	237
881	2	235

Figure 18. Positive trusts table.

Field	Required?	Other notes
id	Yes	The id of the ‘positive trust’. Auto-increments to guarantee uniqueness of the automatically generated value.
userId	Yes	The id of the user which voted the post as ‘trustworthy’.
postId	Yes	The id of the post which has been voted on.

Table 9. Fields used in the positive trusts table.

3.4.1.6. Negative Trusts Table

The negative trusts table utilises the same implementation that was used for the likes and positive trusts table.

id	postId	userId
488	237	1
491	239	3
492	231	3

Figure 19. Negative trusts table.

Field	Required?	Other notes
id	Yes	The id of the 'negative trust'. Auto-increments to guarantee uniqueness of the automatically generated value.
userId	Yes	The id of the user which voted the post as 'untrustworthy'.
postId	Yes	The id of the post which has been voted on.

Table 10. Fields used in the negative trusts table.

3.4.1.7. Relationships Table

A relationships table is used to store/delete a relationship whenever a user follows or unfollows another user. The trust rating the user attributes to the user they are following is stored here as well.

id	followerUserId	followedUser...	trustRating
174	2	1	50
176	3	2	60
177	2	3	60
178	4	2	65

Figure 20. Relationships table.

Field	Required?	Other notes
id	Yes	The id of the relationship. Auto-increments to guarantee uniqueness of the automatically generated value.
followerUserId	Yes	The id of the user which has selected to follow the other user.
followedUserId	Yes	The id of the user that has been followed.
trustRating	Yes	The trust rating the following user has given to the user they have followed.

Table 11. Fields used in the relationships table.

3.5. Application components and pages.

This section aims to outline all the significant components and pages that the application is comprised of, using high-level explanations to do so.

3.5.1. Login and Registration

The first page a new user would visit would be the *login page*, which has been simplified as much as possible to make logging in as easy and seamless as possible. The page features a main 'card' composed of two horizontally separated sections. This is done with the *registration* page in mind; by swapping the positions of the input sections with respect to the previous page upon user navigation to the other corresponding page, visual feedback is immediately provided to the user, indicating they have changed pages. This is done in addition to an entry animation of the cards, giving the user a first impression that the application is polished and enjoyable to use.

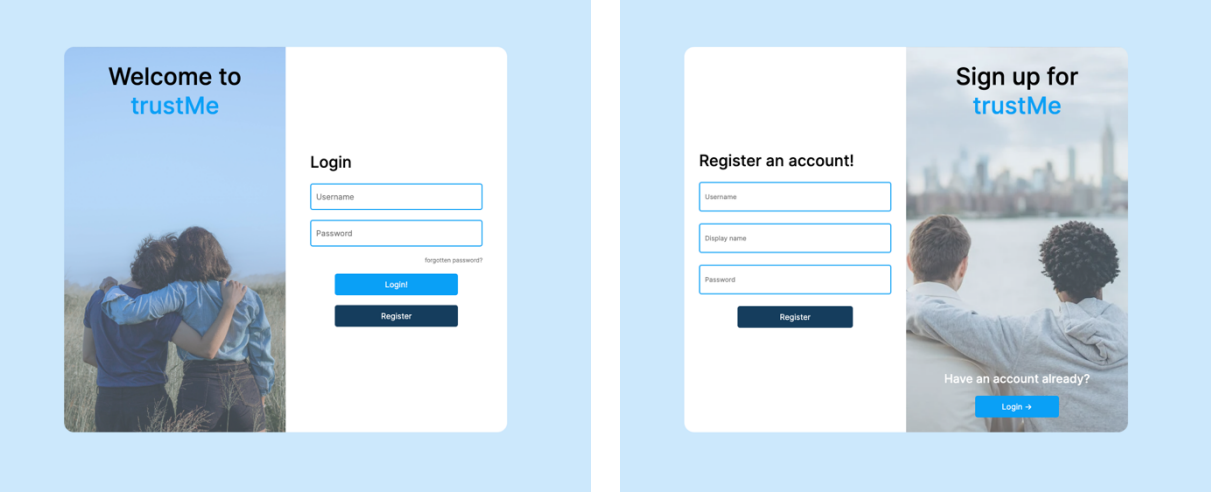


Figure 21. Login and Registration pages.

Various hues of blue have been utilised within the pages to set a theme to the user indicating 'important' colours, which will be used throughout the application.

The use of tooltips has been implemented in the registration section, for users unfamiliar with the website. The tooltips advise users as to why both a username and display name are required, however these only appear upon the hovering of the corresponding input to make the interface as unintrusive as possible.

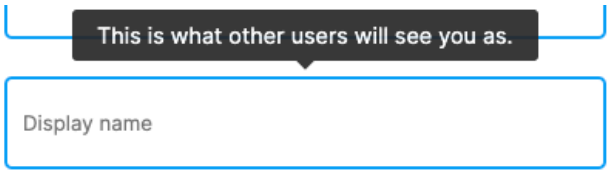


Figure 22. Tooltip for the display name input.

By using React.js, input is automatically sanitized, thus we do not have to worry about security vulnerabilities posed to the database through user input. Password input is also automatically hidden due to the use of the password input element.

The register component handles the registration of users, given the correct input, whilst also handling incorrect input by returning the appropriate error message, which is used to then inform the user about the invalid details. Prior to insertion into the database, the password is both salted and hashed to increase the security of the user's account. Upon insertion to the database, the user is given a default profile picture.

```
const salt = bcrypt.genSaltSync(10); // generating a salt to salt password
const hashedPassword = bcrypt.hashSync(req.body.password, salt);
const q = "INSERT INTO users(`username`,`name`,`password`) VALUE (?)"; // SQL query
```

Figure 23. Salting and hashing of user password using 'Bcrypt' from the node package manager (npm) registry.

On failure the user is notified, informing them that their username has already been taken by someone else. Upon success, the user is notified of their successful registration and is automatically navigated to the login page.

The login component handles the authentication of the user, which is done by checking if the given username and corresponding password exist in the database. Upon failure the user is informed they have supplied the incorrect credentials, without indicating which field is incorrect such as to deter hackers. Upon success, a JSON web token (JWT) is created (which is used for authorisation) and the user is automatically redirected to the homepage. The decision to use JWT for authentication is due to its 'stateless' nature, meaning the token can be stored in the user's local storage as a cookie, negating the need to track the user's session in the server. By using a JWT we are able to constantly authorise the user in order to conditionally permit and deny actions throughout the website as needed (such as the ability to delete posts, update profiles etc.).

An AuthContext Context API is used in conjunction with the JWT, to store the logged in user's data in their local browser storage, allowing other components of the application to access the information and render information depending on the user data stored.

3.5.2. Home page

The home page is the primary page of the application and utilises a 3-column arrangement, which persists throughout the application. The content of the left and right column both display the same data irrespective of the page the user is currently on, given that they are logged in. The left column provides a dark-mode switch and a logout button, whilst the right column provides a mutual friends tab which the user can use to cycle through various mutual friends they may have, these columns will be explained in further detail later on. The main column (middle) consists of 2 main features, the sharing function, which resides at the top of the column, and the posts feed below, which displays all the posts of the user and the posts made by whom they follow.

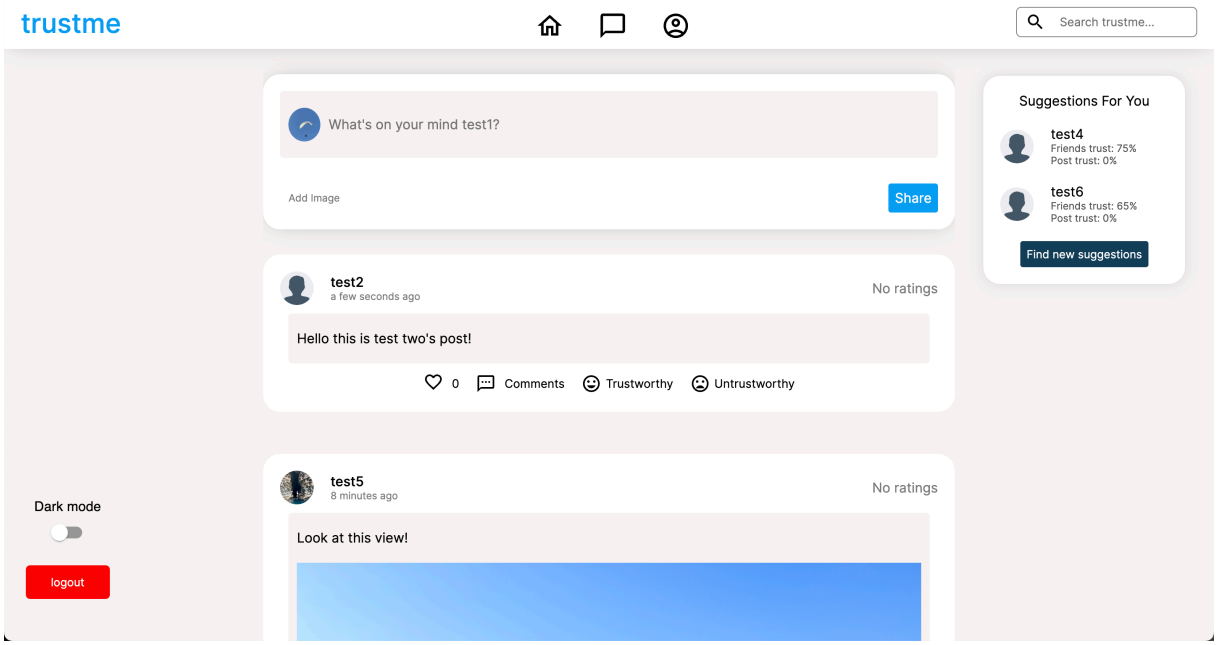


Figure 24. The homepage.

3.5.3. Share function

The share 'tab' gives the user the ability to share content in both a text and image format. It has been placed in a prominent position in order to grab the user's attention straight away, increasing the likelihood that they would share content upon seeing the tab. By using the JWT tokens explained previously, the application is able to display the users profile picture, accompanied by a personalised greeting which acts as placeholder text. This was done to give the application a more friendly, customised experience.

Figure 25. The share tab.

Upon attachment of an image, the user is given a small 'preview' in the share tab of the image reassuring the user that they have attached the correct file prior to sharing the post.

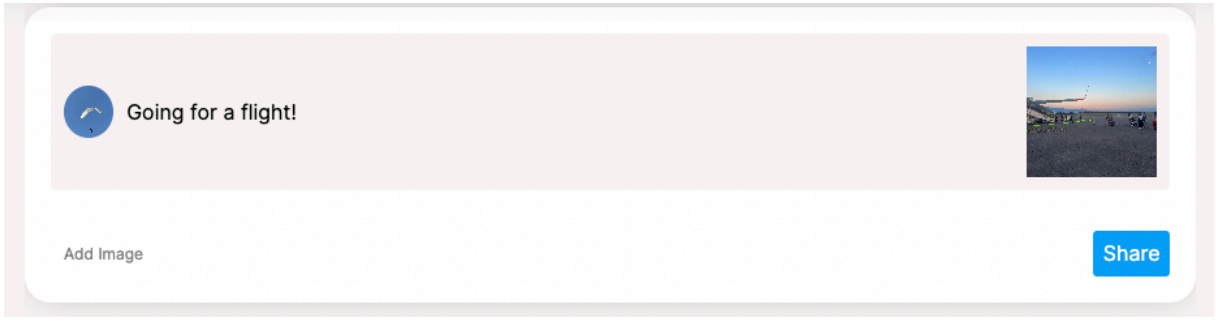


Figure 26. Attached image preview.

Again, a light blue has been used on an important element, the *share* button, which allows the user to quickly recognise and gauge the importance of the button. When this button is clicked, the post is added to the posts table in the MySQL database and can be seen in all related feeds.

3.5.4. Posts

The posts component makes up a significant portion of the project, combining many of the other significant trust and non-trust-related components designed in the project, which gives users the ability to interact with and understand posts much better in the context of trust, when compared to similar applications.

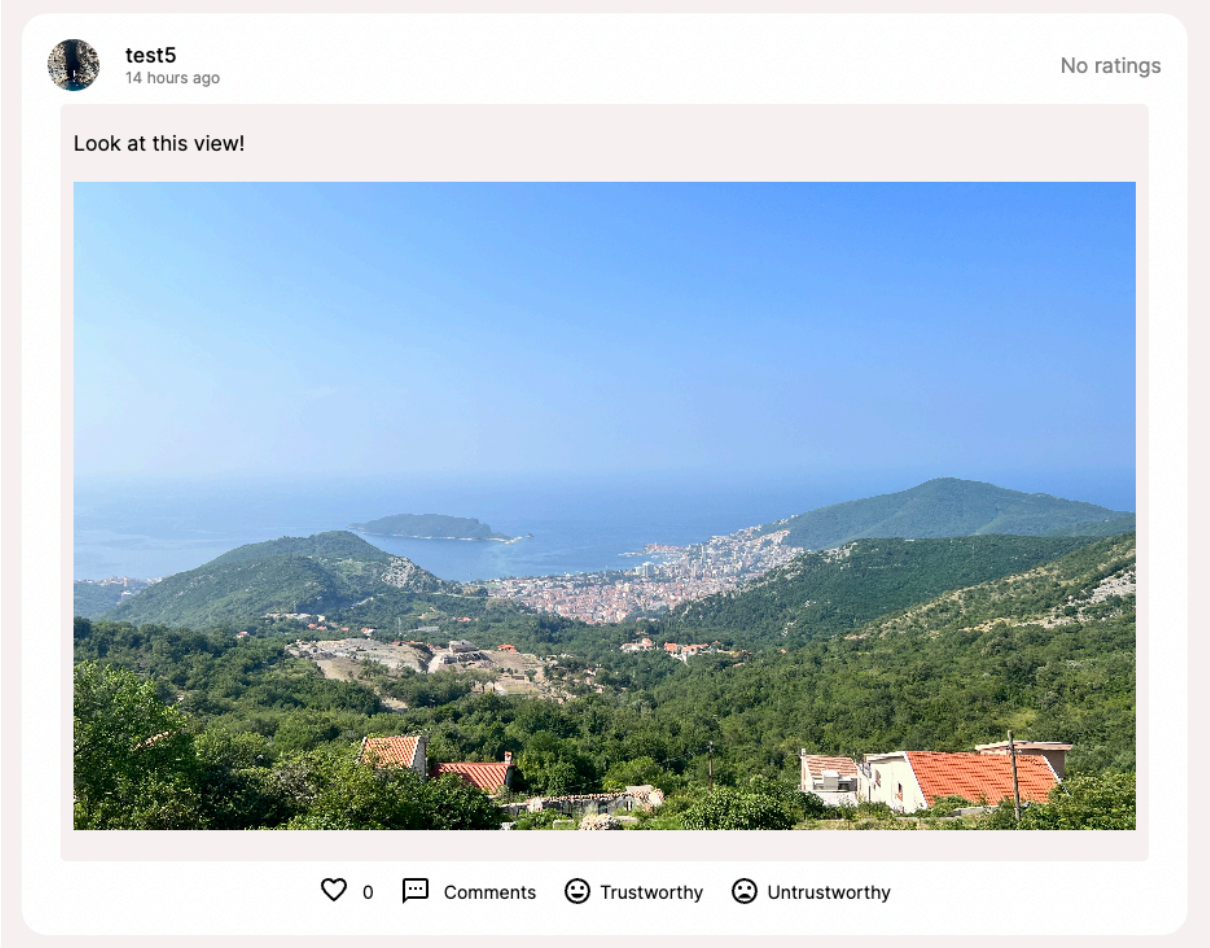


Figure 27. A post from a user.

The top section of every post is comprised of the posters identifying information (their name and profile picture), which is typical for almost every popular social media application, as it allows readers to quickly identify the initial source of the information. Clicking on the name of the poster navigates the user to their profile.

The length of time that has passed since the post has been shared is also given, which has been implemented using the moment package from the npm software registry, which allows the application to store the exact date and time at which the post was uploaded to the database [17]. This time is then subtracted from the last time the application has been rendered (which

would be close enough to the present time) to give the amount of time since the post was uploaded. This is critical in allowing users to better recognise if the contents of the post are outdated.

On the top-right hand section of the post is the *trust rating* for that individual post, which serves as the primary indicator of how trustworthy other users perceive that post to be. This rating is given as a percentage. The calculation of the trust rating is given by the following formula.

$$Trust\ rating\ (\%) = \frac{Positive\ trust\ votes}{(Positive\ trust\ votes + Negative\ trust\ votes)} * 100$$

A number is given next to this value indicating how many users have voted on the trust for the given post. This is necessary as it gives users the ability to recognise how reliable a posts trust rating is, as the amount the rating can change when given a vote varies significantly depending on the number of total votes. If a post has no ratings, this will be indicated, as opposed to a rating of 0% trust, to avoid misleading readers of the post.

Figure 28. From left to right, (a) trust of 50% with 2 votes, (b) trust of 100% with 1 vote, (c) no votes.

The main body of each post is occupied by the shared contents, which is the text of the post and, if present, the accompanying image. This is given the most space as it is the primary content of the post which users will pay attention to.

The bottom section of every post feature 4 buttons which are used for; liking the post, rating a post as trustworthy or untrustworthy, and a button to open the comment section for the given post. These buttons serve as the primary ways for users to *engage* with the post.

Figure 29. The engagement buttons. From left to right, (a) like button, (b) expand comment section button, (c) trustworthy, (d) untrustworthy buttons.

To like a post, the user must simply click on the heart icon, with the number beside the icon indicating the amount of 'likes' the post has. When this is done, a record is added to the likes table within the database, which contains the ID of the user and the ID of the post that has been liked, ensuring a user can only give each post a singular like. In doing so, the presence of the like can be checked by checking the presence of the record in the database, which is used to provide visual feedback to the user, informing them whether or not they have liked the post, as well as allowing the application to sum the total likes for each individual post through basic calculation using the total number of records within the database that contain the post ID. The SQL queries for getting total likes, adding and removing likes are given below.

```
SELECT userId FROM likes WHERE postId = ?
INSERT INTO likes (`userId`,`postId`) VALUES (?)
DELETE FROM likes WHERE `userId` = ? AND `postId` = ?
```

Upon liking a post, the heart icon animates and changes colour to red, indicating to the user that the post has been liked by them.

Figure 30. Like icon (unliked and liked).

The trustworthy and untrustworthy buttons are implemented using the same logic as the like functionality, having separate tables in the database to store their relevant data. The trust totals for both positive and negative trust are then each summed up and used to calculate the trust rating of the post as demonstrated in the *post trust rating* formula. The user is also prevented from rating a post as both trustworthy and untrustworthy simultaneously, by removing the record from the database when the opposing button is clicked (if present in the database at all).

Figure 31. The icon colour when the user votes a post as trustworthy and untrustworthy respectively.

Lastly, the comment icon expands the post, revealing any comments that have been left, as well as an input section that allows the user to leave a comment. The opening and closing of the comment section have been animated using the framer-motion package. The logic required to add a post is extremely similar to that of when a like is added to the database, ensuring that other posts do not feature comments that do not belong there, with the help of using the unique comment ID as a key for the comment.

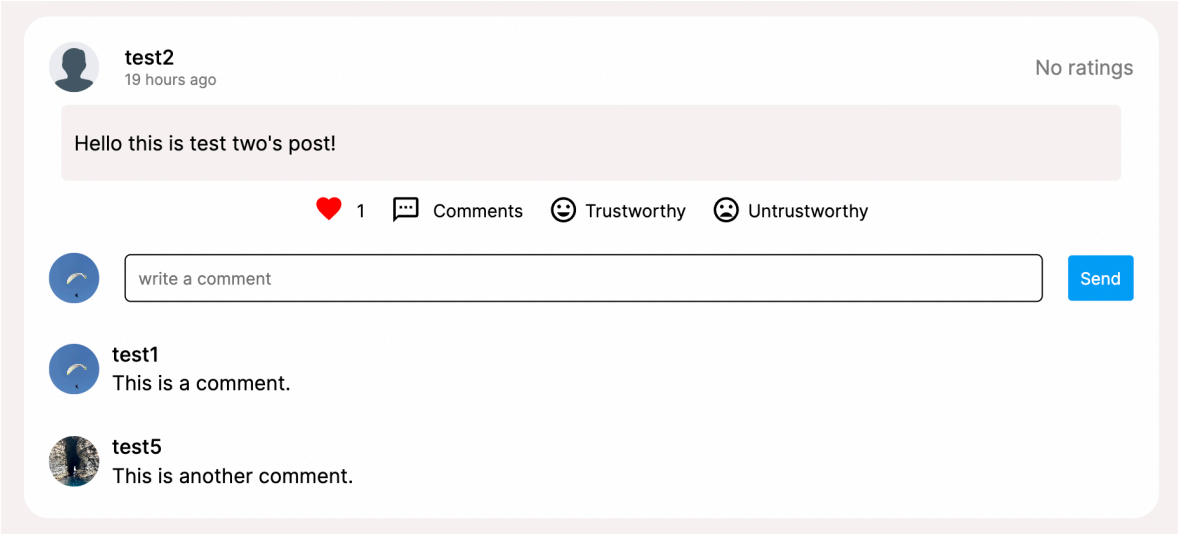


Figure 32. A post with the comment section opened.

The application allows users to delete their *own* posts, by comparing the user ID of the post with the ID of the current user which is given by the AuthContext API (part of the react package). This delete message is conditionally rendered, preventing users from being given the option to delete other people's posts.

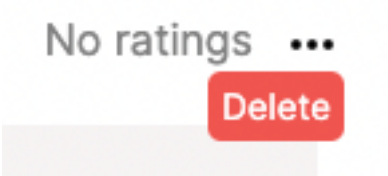


Figure 33. Delete button for a post. Appears after clicking the ellipsis button above.

3.5.5. Profile page

The profile page is comprised of 2 major sections, with the first being the users profile information tab, which resides at the top of the page, and the second being their posts which are situated directly under this tab. The functionality of the first section changes depending on if the user is visiting their own profile compared to visiting that of another user.

For all profile pages a trust level section can be found in the top right section of the profile information tab, this serves as a means for page visitors to quickly assess the trustworthiness of the user profile they are visiting. The trust levels are split up into two parts, a friend rating, and a post rating. This has been done so that users are able to give a much more direct trust rating in the form of the friend rating, whilst the post rating will develop naturally as the user garners engagement with their posts. As such, the visitor is given more information to be able to decide whether they can trust the user and if they would like to follow them or not. The calculations and rationale behind each measurement will be explained further later in the report.

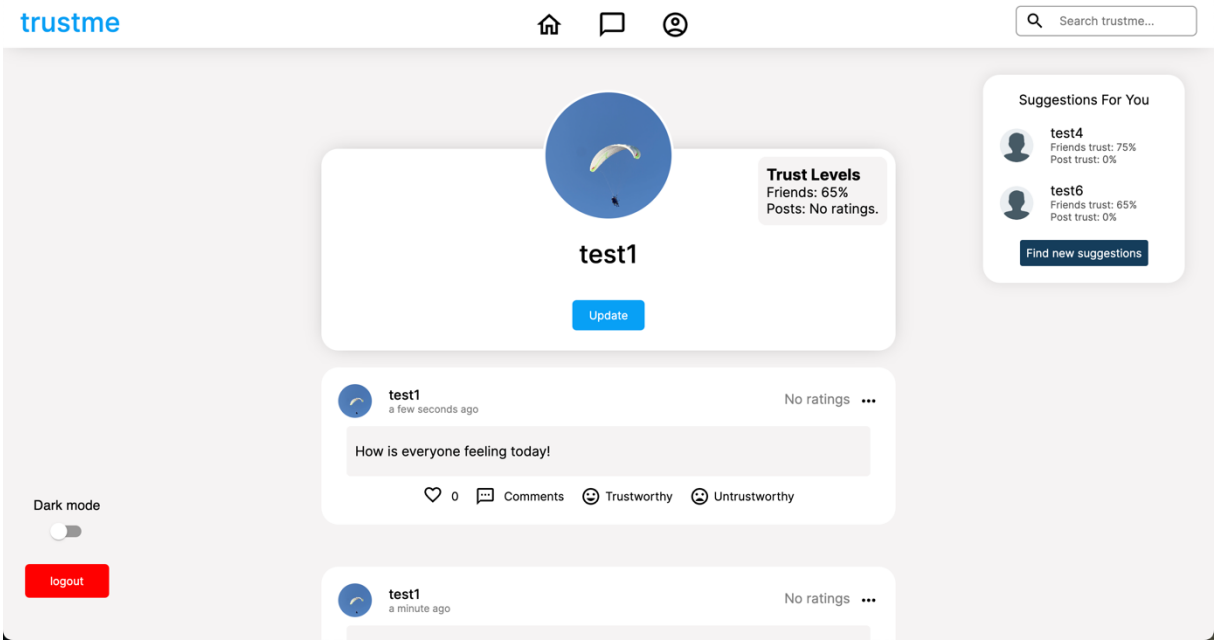


Figure 34. The profile page.

3.5.5.1. Profile page (current user)

When a user visits their own profile page, they see the page that can be seen in *figure 34*. Upon clicking the update button, the user is prompted with a modal which can be seen below.

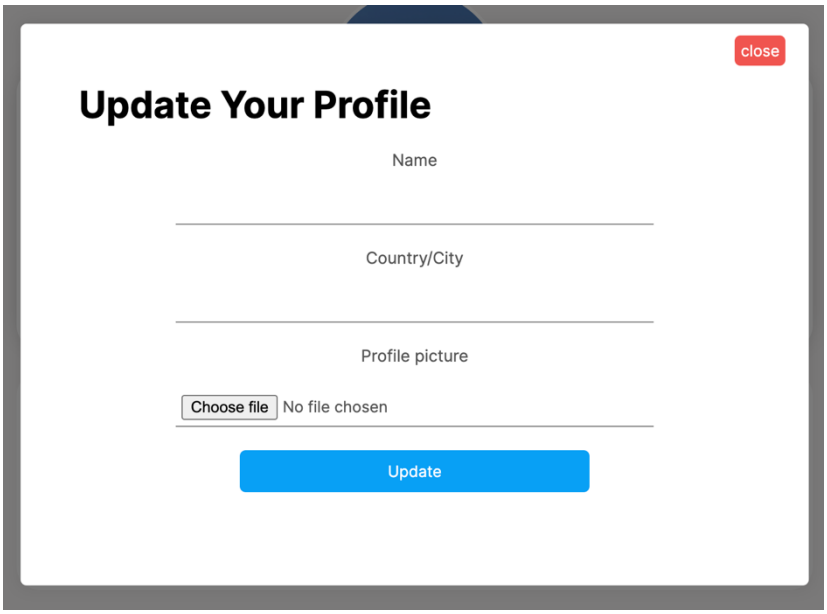


Figure 35. The modal allowing users to change various aspects of their profile.

As seen in *figure 35*, the user is able to change their display name, city/country, and profile picture. If the user decides to add their country of residence, an emoji of the country is automatically added to the end of the country name on the profile information tab, to provide clarity to other visiting users of where the user is from. This was done with the assistance of the country-locale-map package from the npm registry [18]. Displaying the city is purely optional, such as to allow users to maintain a degree of anonymity should they wish to.

When a new profile picture is uploaded a 'fake' URL is created, which is added to the users record in the database. This URL corresponds to the URL of the image stored locally in the public folder of the application. This was done so that the user is able to use files from their own device as opposed to having to upload the file to an external image hosting service and using the URL from there, which is not only tedious but also creates unneccesary privacy concerns.

When the update button is clicked all relevant user information is updated within the database and the page is forced to re-render, to show the newly changed profile information. The user is also able to simply close the modal using the 'close' button, should they decide that they do not want to make any changes to their profile. The SQL query for updating the user information can be found below.

```
UPDATE users SET `name`=?,`city`=?,`profilePic`=? WHERE id=?
```

3.5.5.2. Profile page (other users)

When a user visits another user's profile page, they are greeting with a similar arrangement as in *figure 34,* however the update button is instead a 'follow' or 'following' button, depending on whether the visiting user is following the user who's profile they are on. When a user clicks on the follow/following button they are prompted with a different modal which can be seen below.

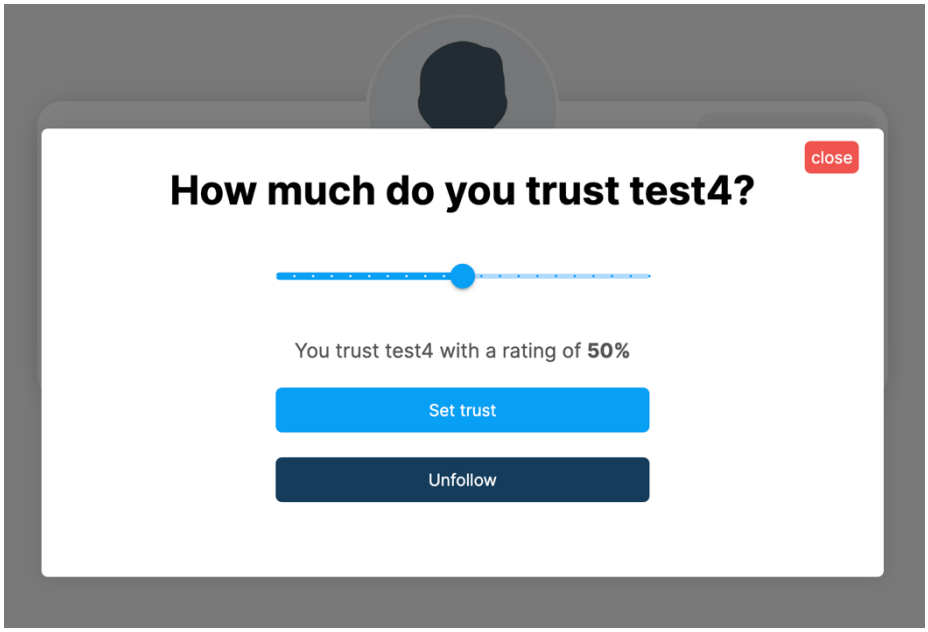


Figure 36. The modal prompting users to rate how much they trust the followed user.

By setting the default slider value to 50, it is visually implied to the user that the scale that they must rate the trust is from 0 – 100. The slider increases in intervals of 5, as we believe this provides the best balance of giving the user precise control of the value, whilst simultaneously being able to quickly set the trust to an appropriate value without excessive precision needed. An input that allows the user to enter a numeric value was also considered, but as that requires extra effort from the user to use the keyboard every time they follow a user, we believe our choice of the slider to be the better, more user-friendly approach.

The user is also able to unfollow users by clicking the unfollow button, which removes the relationship from the database, as well as any effects their given trust rating can have on the unfollowed users relationship trust ratings. This is done by using the on delete - cascade function within MySQL. The user is also able to simply close the modal should they not want to take any further actions after opening it. The SQL queries for updating the users trust and unfollowing users can be seen below.

```
UPDATE relationships SET `trustRating`=? WHERE id=?
DELETE FROM relationships WHERE `followerUserId` = ? AND
`followedUserId` = ?
```

3.5.6. Friend trust rating

The friend trust rating (or relationship trust rating) is calculated by finding the sum of every trust rating each users' followers have given them, in this case, *Individual Friend Trust Rating* and dividing this by the number of followers.

$$Average\ Friend\ Trust = \frac{\sum Individual\ Friend\ Trust\ Rating}{Number\ of\ Followers} * 100$$

This gives us an average on how much the user is trusted by their followers, which is then used to display on their profile information, as well as being displayed in the mutual friends tab. This approach allows each follower to have an equal influence on the users trust rating.

3.5.7. Post trust Rating

As opposed to the direct average approach used in the *Friend trust rating*, a weighted average approach is used to calculate the overall post trust rating. This has been done as we believe a post that has more engagement (e.g. 500 votes) should have more influence on the users average post trust rating than a post that has lesser engagement (e.g. 20 trust votes).

Every post has a *Total Vote Count* which signifies the number of votes for that individual post. We are able to calculate the total amount of votes across all the users posts by iterating over each post and adding the total vote count to a variable, *Overall Votes*, until every post has been iterated through.

$$Overall\ Votes = \sum Total\ Vote\ Count$$

The *Overall Votes* value is then used to calculate the amount of influence an individual post should have on the trust rating, using the following formula, where *Post Trust* is the direct post trust for an individual post, and *Weighted Post Trust* is the recalculated post trust for the individual post when compared to all the other posts of the user.

$$Weighted\ Post\ Trust = \frac{Total\ Vote\ Count}{Overall\ Votes} * Post\ Trust$$

The *Weighted Post Trust* for every post is then summed together to give the users average post trust.

$$Average\ Post\ Trust = \sum Weighted\ Post\ Trust$$

3.5.8. Mutual friends

A ‘Suggestions for you’ tab features in the right-hand column throughout all pages in the application when the user is logged in, which recommends the user potential friends who already share a relationship with the user’s current friends.

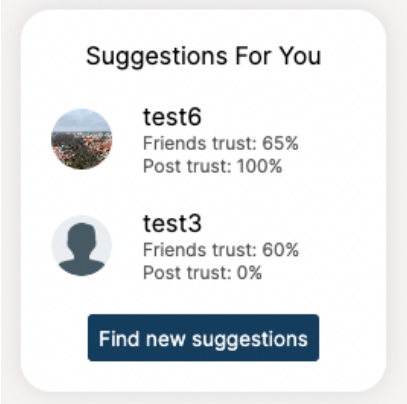


Figure 37. The mutual friends tab.

The decision was made to only show 2 mutual friends at a time, such as to not clutter the application with a large list of mutual friends which the user may not be interested in. Instead, the user is able to cycle through their mutual friends by two options at a time using the ‘Find new suggestions’ button. Once the user has cycled through all their possible mutual friends they are informed and presented with a ‘reset’ button which resets the tab back to the first set of mutual friends.

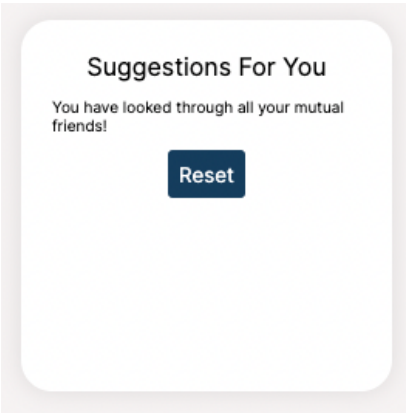


Figure 38. The mutual friends tab when the user has cycled through all available mutual friends.

The mutual friends are displayed in the order of their corresponding ‘Friends trust’, from highest to lowest. This was done as we believe that, when choosing friends, people are more likely to value the opinions of their friends compared to other factors. However, the post trust of the mutual friend is also displayed to provide extra information for the user, to assist their decision as to whether they would like to follow them or not. Clicking on the mutual friend’s name redirects the user to the mutual friend’s profile page, where the user is able to follow them or look at the posts they have made.

The SQL queries for finding a user’s mutual friends can be seen below.

```
    SELECT r1.followedUserId
    FROM relationships AS r1
    LEFT JOIN relationships AS r2 ON r1.followedUserId =
r2.followerUserId AND r2.followedUserId = r1.followerUserId
    LEFT JOIN relationships AS r3 ON r1.followedUserId =
r3.followedUserId AND r3.followerUserId = ?
    WHERE r1.followerUserId = ?
      AND r2.followerUserId IS NOT NULL
      AND r3.followerUserId IS NULL
      AND r1.followedUserId != ?
```

3.5.9. Navigation bar

The navigation bar provides the primary method for the user to navigate across different pages within the application. It is present on every page once the user is logged in, to allow for quick access to the most frequently used pages on the application.

Figure 39. The navigation bar.

On the left side the application name can be found, this is a clickable element which redirects the user to the homepage, as is the norm with most website. It can be assumed to user is aware of this already due to their previous experience using other websites.

In the middle a button group can be found, which consists of home, conversation, and profile icons. The home icon redirects the user to the homepage, similar to the application name. The direct message icon does not serve a functional purpose as of writing; however, it is intended to redirect the user to a page showing all their current conversations/messages with other users, once implemented. Lastly, the profile icon redirects the user to their own profile page. Each button in the button group has been styled with visual feedback in mind, by changing their appearance upon mouse hover.

Figure 40. The home icon colour and background colour changes upon mouse hover.

On the right-hand side, a search box is present, which the user can use to find other friends whom they know the name of. The user is able to click on the name to be redirected to their corresponding profile. The decision was made to show the users profile picture alongside their name in order to allow the user doing the search to easily differentiate between users that have the same name without having to click on each profile.

Figure 41. The search bar in use.

3.5.10. Dark mode

A ‘dark mode’ was implemented to provide users with an alternative option to the default traditional light mode colour scheme of the application. This involves changing all content to be displayed using a darker colour scheme which has a range of benefits for the user. The most important factor in the choice to implement dark mode was that it would reduce eye strain and fatigue for the user, allowing them to use the application for prolonged periods of time, particularly during night-time usage.

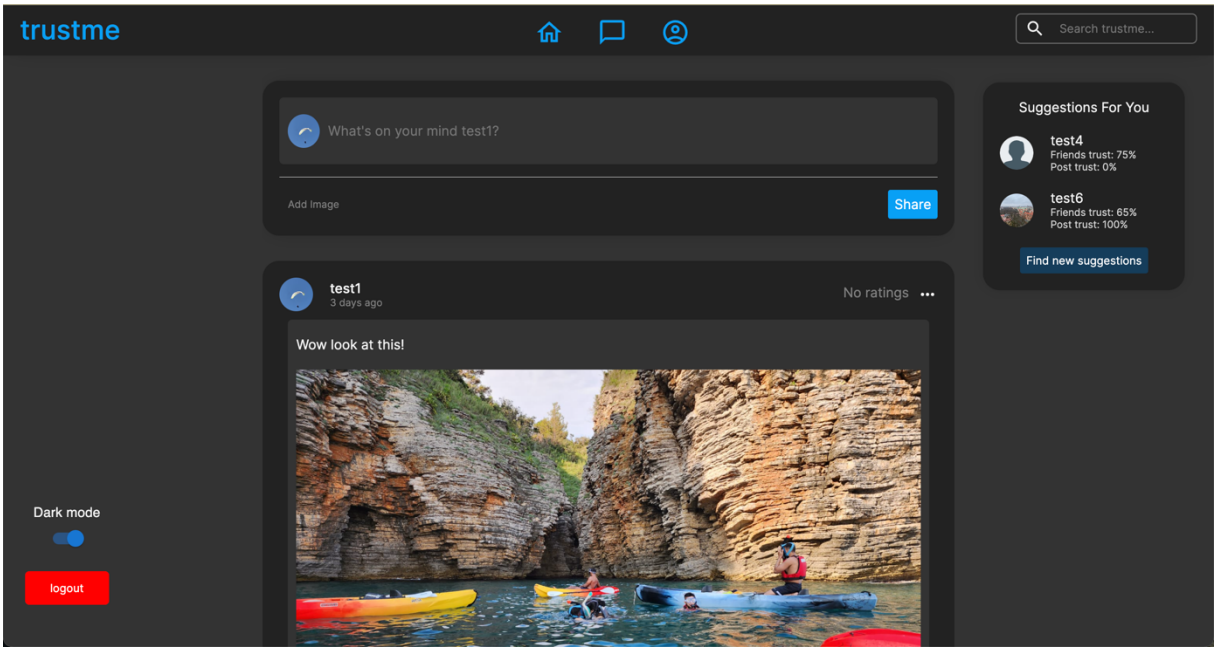


Figure 42. The homepage whilst dark mode is enabled.

The dark mode was implemented with the assistance of *mixins* which is a feature within SCSS. Mixins allows the developer to group a set of specific styling instructions that can be re-used throughout the application, making the implementation of a dark mode relatively fast. Using this in conjunction with the context feature in react allows the dark mode to persist throughout the application (similar to how the authentication feature works), as the dark mode ‘switch’ stores a Boolean value (light or dark) in the user’s local storage, which each components corresponding stylesheet will adhere to depending on the value. The SCSS code for part of the dark mode implementation can be seen below to give a better idea of how the Boolean value is used.

```
$themes: (
  light: (
    textColor: #000,
    bg: white,
    logo: darkblue,
    bgSoft: #f6f3f3,
    textColorSoft: #555,
    border: grey,
    buttonColor: black,
  ),
  dark: (
    textColor: whitesmoke,
    bg: #222,
    logo: white,
    bgSoft: #333,
    textColorSoft: lightgray,
    border: #444,
    buttonColor: #08A0F5,
  ),
);
```

3.5.11. Mobile Optimisation

The application has been optimised for mobile viewpoints by removing the left and right columns, which is typical for most mobile versions of social media applications. In order to retain some functionality, the logout button was relocated to the navigation bar. The mobile optimisation implementations were done using mixins, which made the development of this feature swift to implement.

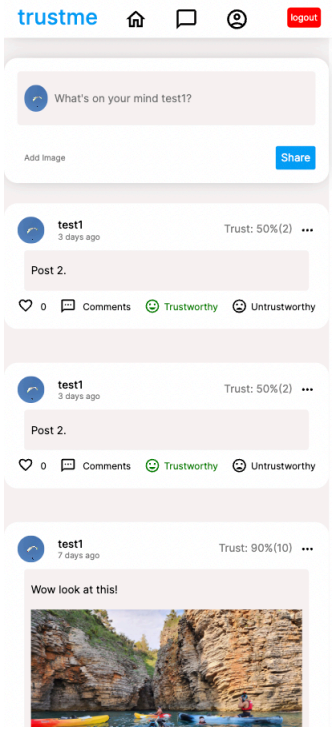


Figure 43. Homepage view on mobile display.

3.6. Testing

As the developer has utilised an agile approach to the project, there was not a dedicated testing period. Instead, testing was undertaken as each component of the application was developed, an approach known as *shift-left testing*. The main benefit of this approach is that it ensures there is a lower likelihood that the product overshoots estimated timelines, as major bugs are recognised early in development [19].

Although testing was generally done every time a component was developed, it was deemed more efficient at times to artificially enter certain details directly into the database to ensure the functionality of components prior to their actual development. A good example of this is creating relationships directly in MySQL prior to developing the follower/following UI and logic, ensuring time was not wasted in developing the UI for a component that did not work. Testing was also redone if any components of the application were created that affected previously tested components.

As there was a significant number of tests conducted, we have decided to only show the testing of trust calculations below. Tests regarding the functionality of the app as a whole can be found in the appendix, thus it can be assumed important testing such as the prevention of duplicate accounts, handling of incorrect login credentials etc. has been handled appropriately.

Test Description	Expected Outcome	Actual Outcome	Result
Post trust: A post that has 2 votes, with one vote being trustworthy and the other untrustworthy.	Using the formula given in *section 3.5.4*, the post trust for the post should be 50%. The number of votes should be 2.	The post trust displays 50% (2), indicating a post trust of 50% with a total of 2 votes.	**Pass**
User friend rating: A user has 2 followers. Follower 1 gives the user a trust rating of 80%. Follower 2 gives the user a trust rating of 40%.	Using the formula given in *section 3.5.6*, the average friend rating for the user should be 60%.	The friend rating shown on the user's profile is 60%.	**Pass**
User overall post rating: A user has 2 posts. Post 1 has a trust rating of 90% with 10 votes. Post 2 has a trust rating of 50% with 2 votes	Using the formula given in *section 3.5.7*, the average post rating for the user should be 83.33%.	The overall post rating shown on the user's profile is 83%.	**Pass**

Table 12. Tests for trust calculations.

3.7. Summary

In summary, this chapter has outlined all planning, design, technical steps and technologies used to achieve the goal of creating a trust-based social media application. The process of development was arguably quite smooth, with development being finished well within expected timeframes, due to the clear and defined planning done by the developer with the assistance of the supervisor.

4. Results and Evaluation.

This chapter aims to discuss the results of the project and evaluate how well the results align with what was set out to be achieved. We will also look to evaluate the technologies and methodologies used.

4.1. Results

An ideal way to contextualise the results of the project is to compare the results versus the functional and non-functional requirements set out prior to development of the application, which were outlined in *section 3.1*.

4.1.1. Evaluation of functional requirements

The project was generally successful in fulfilling the functional requirements outlined, with the only unfulfilled requirements being users not being able to delete their account and the user's inability to directly message other users.

As a whole we believe that the implementation of the core social media application features were executed well, with FRs 1,2,3,4,5,6,8,10 and 15 being implemented with complete functionality. It was important to ensure these requirements were met before the implementation of the trust-based features, as the rationale behind the trust-based features was to supplement the experience of using a social media application that is otherwise functionally similar to competitors.

The FRs and their status can be found below.

Requirement description	**Priority: R, H, M, L**	**Achieved?**
1. The application should require users to register an account.	R	Yes
2. The user should be able to update their profile information.	R	Yes
3. The application should allow users to be able to share content, including images.	R	Yes
4. The application should allow users to 'follow' each other, which causes the followed users posts to appear to the following users home feed.	R	Yes
5. The application should allow users to 'like' posts.	R	Yes
6. The application should allow users to comment on posts.	R	Yes
7. The user must be able to rate posts in terms of its trust/reliability.	R	Yes
8. Users should be able to delete their own posts.	R	Yes
9. The application should allow users to directly message each other.	M	No
10. The user should be able to search for other users using their name.	R	Yes

11. The user must be able to directly rate other users based on trust.	R	Yes
12. The application must integrate a trust-ranking feature that calculates the level of 'trust' one should have of other users using an algorithm that is based on a variety of parameters.	R	Yes
13. The application suggests friends based on trust rating.	R	Yes
14. The user should be able to delete their account.	H	No
15. The user must be able to log out of the application.	R	Yes

Table 13. Completion status of functional requirements.

In implementing FRs 1,2,3,4,5,6,8,10 and 15 the user is able to:

- Register an account with a password that is stored securely using hashing and salting techniques.
- Search for other users on the application using the search box which is present in the navigation bar.
- Follow and unfollow other users on the application.
- Share posts on the application, including images, which appear on the user's homepage feed, the homepage for followers of the user and the user's profile.
- Engage with posts on the application, which can be done via likes, trustworthy and untrustworthy ratings and by writing comments.
- Users are able to update their display name, city/country and profile picture.
- See mutual friends that have been recommended by the application.
- Log out of the application using the logout button.

The remaining functional requirements involve the features that enhance trust between users of the platform. We believe these were implemented well, in a manner that is user friendly and fits organically with the rest of the application. By using multiple methods to measure the trust of both posts and users of the platform, we have achieved the goal of giving the user a variety of tools to help both assess and vote on content and other users of the application.

In implementing FRs 7,11,12 and 13 the user is able to:

- Vote on the credibility of posts using the trust buttons present on the post, which in doing so updates the trust rating on the post, which all users can see. By using the trust rating of all posts, the application calculates a weighted average for the overall trustworthiness of the user's post, which considers the level of engagement for each post.
- Rate other user's on how much they trust them upon following them, which can be updated whenever needed during their relationship. The application then calculates an average value for how much the user is trusted by their followers.
- See each trust rating on the user's profile as within the mutual friends tab.
- See mutual friends which are sorted by their corresponding friends' trust.

Although the implementation of trust features was deemed largely a success, we do believe that improvements could be made. There was significant consideration given to the thought of allowing the trustworthiness of a user affect the strength of the vote they give to posts which

would have been an interesting concept. Unfortunately, due to both time restrictions and the way that post trust is calculated, which initially uses binary values (votes) instead of numeric values (e.g., votes multiplied by user's trust, thus requiring a complete overhaul of both the database tables and program code), meant that this was not plausible for this project.

4.1.2. Evaluation of non-functional requirements

The project was again generally successful in achieving the non-functional requirements outlined.

The NFRs and their status can be found below.

Requirement description	**Priority: R, H, M, L**	**Achieved?**
1. The application should feature a dark mode.	H	Yes
2. The app should be optimised for mobile devices using a responsive design that is able to accommodate a wide range of screen sizes.	R	Yes
3. The application will be secure and sensitive user information (such as passwords) be salted and hashed when stored in the database.	R	Yes
4. The application complies with GDPR regulations.	M	No
5. The application is available on most commonly used browsers (Chrome, IE, Firefox, Safari).	R	Yes
6. The application should feature a colour theme that assists navigation of the application.	R	Yes
7. The application should make use of animations to improve the user experience.	H	Yes
8. The application should make use of established design principles to provide a clear and intuitive user experience.	R	Yes
9. The application is relatively performant to provide an optimal user experience.	R	Yes
10. Development will involve the use of version control systems to ensure there is a recent backup of a working version of the application.	H	Yes
11. The application features a usage guide.	L	No

Table 14. Completion status of non-functional requirements.

Here, there are only two unfulfilled requirements. The first being the applications non-compliance with GDPR regulations (NFR 4). While it would have been ideal to have the app comply with GDPR regulations, its purpose is as a demonstration and thus it is not necessary to do so, as the application has not been deployed publicly. The other unfulfilled requirement is the lack of a user guide (NFR 11); however, we believe this requirement has been made redundant, due to the self-explanatory nature of the application which is explained in further detail below.

NFR 8 has been subtly implemented by adding obvious, visible and understandable feedback for every clickable element on the application, alerting the user every time they have executed

an action. NFR 8 has also been achieved by the use of easily understandable icons and prompts (think of the placeholder texts for the many inputs in the application). These two design elements are arguably the most important design elements used in the application as it helps teach the user the effects of their actions, negating the need for a user guide, as the application becomes self-explanatory through these techniques.

By meeting NFR 3 in addition to the handling of incorrect user credentials by use of the authentication system developed, we believe that the application provides users with adequate security regarding their private details.

Overall, the implementation of the non-functional requirements has ensured that the core functionality of the application is supported by user-friendly and theory-backed design, which should lengthen user session times. The final product generally feels much more polished and intuitive to use compared to the feel of the application prior to the implementation of the NFRs.

4.2. User surveys

4.2.1. Survey methodology

Whilst comparing the results to the requirements analysis provides useful reflection to the success of the project, we felt it was necessary to allow people to test the application for themselves, after which they were provided a questionnaire to complete. As the application has not been deployed for users to access remotely, we were only able to ask people in the local vicinity that were willing to meet in person in order to test the application on our device. The device users were able to test the application with was an Apple MacBook air laptop with a 13-inch screen and M1 processor. Users were given a pre-made account with relationships and posts already existing, to simulate a realistic homepage feed without excessive work needing to be done by the participant. Although the participants doing the questionnaire were known personally, the survey was anonymous, and thus we would not know the answers until all participants had completed the survey. This was made aware to them such as to minimise any bias the survey may encounter.

The survey provider was *Qualtrics*, who provided a quick and simple way to set up the survey with a multitude of input options for each question [20]. The survey was tailored, keeping the amount of mental energy required for survey completion in mind, in order to get the highest quality feedback possible. As such the number of free-text answers the user must complete was kept to 3 answers, as Qualtrics suggests that past this number the completion rate begins to decline, which implies that the quality of the answer would also decrease. A combination of closed-end *Likert* style questions and open-ended questions were used to help address both quantitative and qualitative aspects of the user feedback. In using the Likert style questions, we were able to easily identify and quantify trends whilst still being able to realise the understanding of the user's choices, preferences, and opinions through open-ended questions.

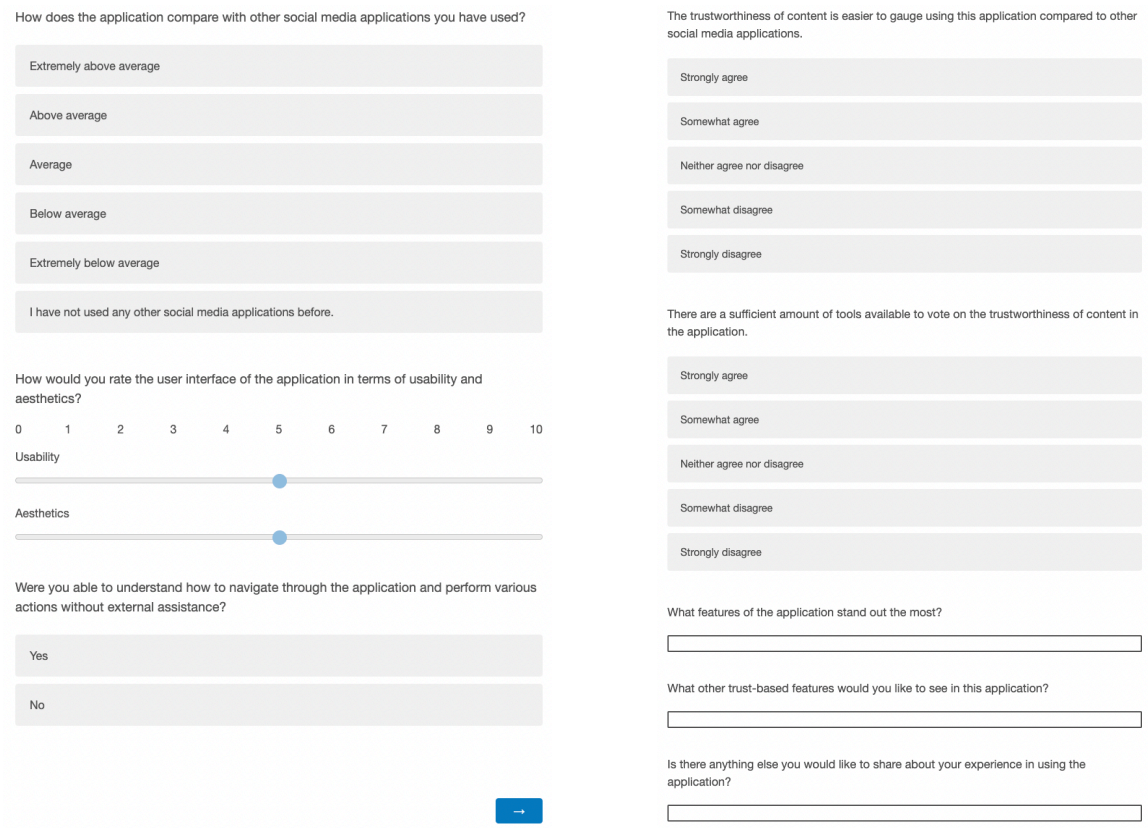


Figure 44. The appearance of the questionnaire.

The 8 questions asked to the participants can be seen below.

No.	Question	Answer style	Question area
1	How does the application compare with other social media applications you have used?	Likert	General
2	How would you rate the user interface of the application in terms of usability and aesthetics?	Slider scale (0 – 10), two separate sliders for usability and aesthetics.	User Interface / User Experience
3	Were you able to understand how to navigate through the application and perform various actions without external assistance?	Yes/No	User Interface / User Experience
4	The trustworthiness of content is easier to gauge using this application compared to other social media applications.	Likert	Trust features
5	There are a sufficient amount of tools available to vote on the trustworthiness of content in the application.	Likert	Trust features

6	What features of the application stand out the most?	Textbox	General
7	What other trust-based features would you like to see in this application?	Textbox	Trust features
8	Is there anything else you would like to share about your experience in using the application?	Textbox	General

Table 15. Questions asked to participants of the survey.

4.2.2. Survey results

This section will be split up according to the 3 question areas of the survey: User interface, Trust features and General thoughts. The survey was given to a total of 11 participants.

4.2.2.1. User interface survey results

The feedback regarding the user interface/user experience was overwhelmingly positive, with all users answering yes to Question 3, evidencing the intuitive and user-friendly design of the application.

Question 2 was presented to the participant using two sliders, on a scale from 0 – 10, which would ask the user to rate the application in terms of its usability and aesthetics respectively. Interestingly, all participants rated the two parameters with a maximum of 1 point deviation from each other.

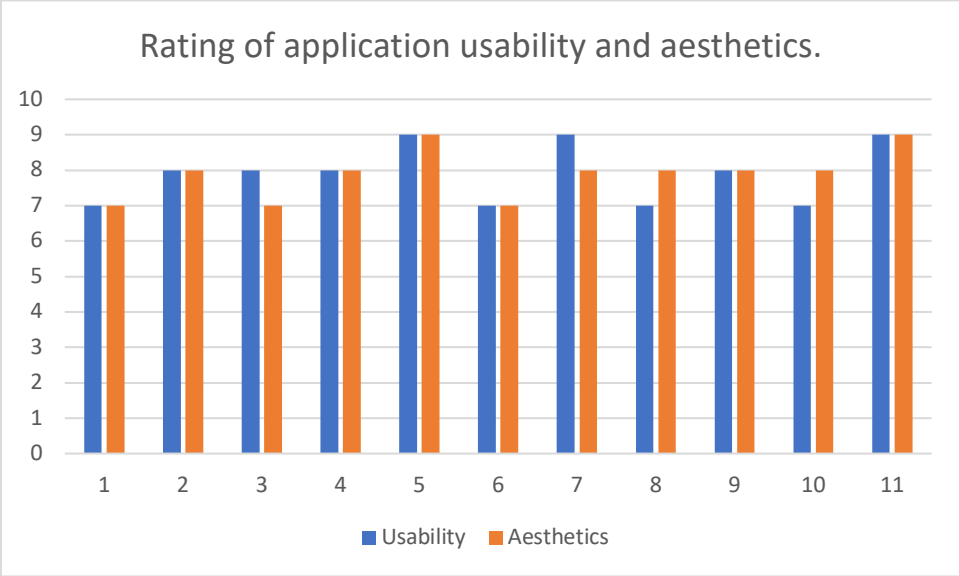


Figure 45. A bar chart showing survey participant ratings of application usability and aesthetics.

Looking at the chart above, and with this 1-point deviation in mind, we can see that there is a correlation between the usability of the application and the user's perception of the application aesthetics, perhaps implying that an app that is visually appealing and organised inherently improves its usability. On average users rated both the usability and aesthetics with a rating of 7.9 out of 10, which we are pleased with.

The results also tend to agree with the additional comments made by some participants, which specified that *'the application was very fluid and smooth to use'* and that *'The application as a whole felt refined and to the quality I would expect of social media applications'.* The dark mode feature was also well received, with two participants highlighting its usefulness.

Overall, in terms of the application UI and UX, the project was highly successful, which was pleasant to see as a significant amount of time, research and theory was allotted to implementing successful design principles. However, as the laptop uses high end hardware, performance was expected to be good, it would be interesting to compare the feedback given with feedback from participants which would conduct the test using a laptop with significantly worse hardware.

4.2.2.2. Trust features survey results

The emphasis on the trust-based features within the application did not go unnoticed, with all questions using the Likert scale being answered positively. However, some additional comments did specify important criticisms of the application.

For Question 4 the feedback was generally very positive, with most participants seeing a real benefit of using this application compared to competitors in the realm of trust-based features. There were 0 participants that saw the application have a negative effect on their abilities to gauge the trustworthiness of content, with 91% of participants seeing slight or significant benefits.

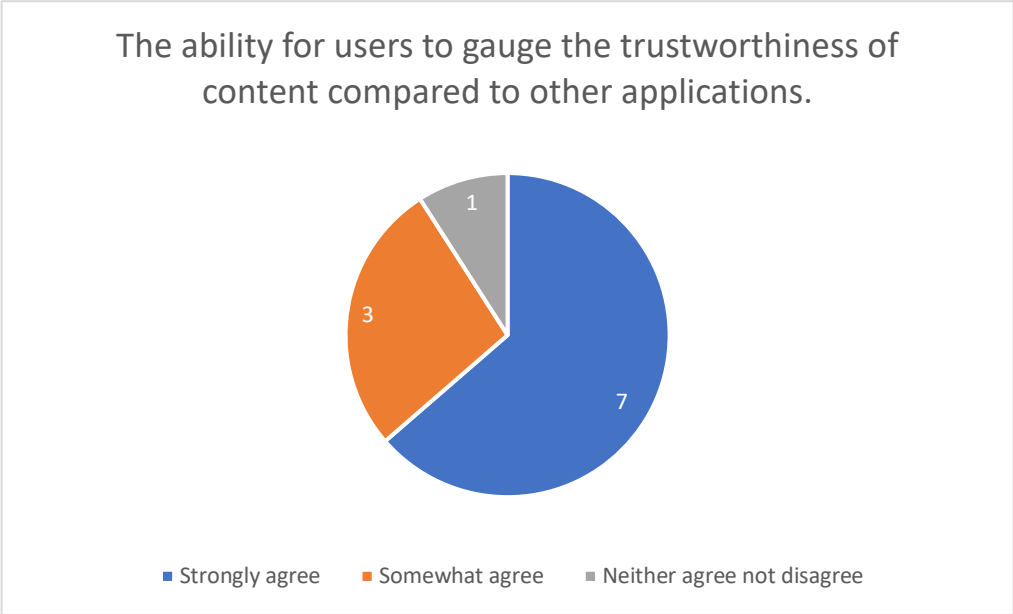


Figure 46. A pie chart showing the distribution of answers to question 4.

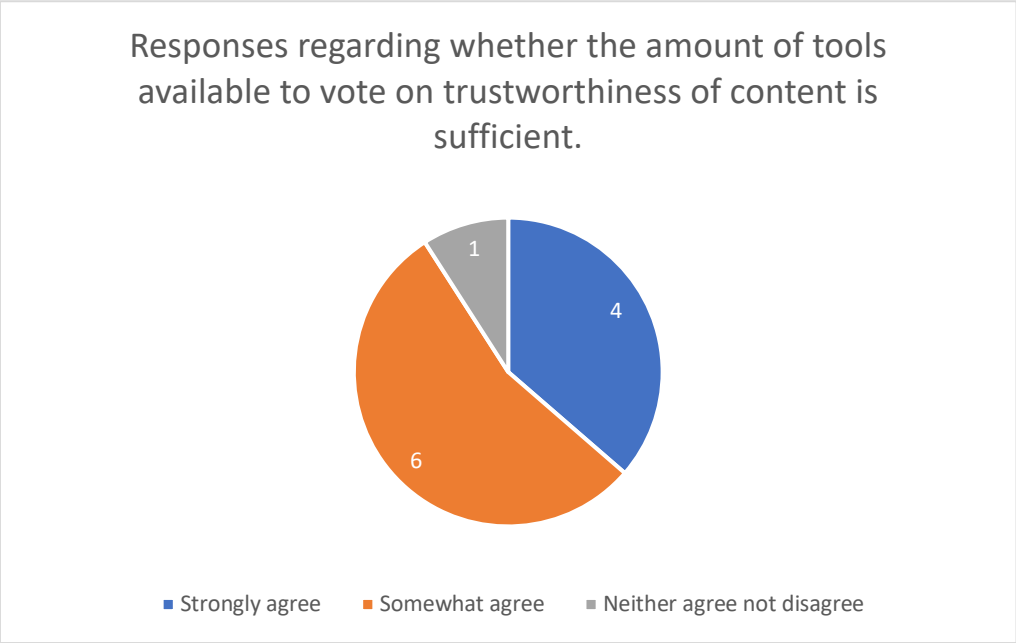


Figure 47. A pie chart showing the distribution of answers to question 5.

Question 5 follows a trend where users were generally satisfied with the number of tools available to vote on the trustworthiness of content, however there is a noticeable drop in the number of participants 'strongly agreeing' here compared to Question 4. When looking at the individual participants, many would strongly agree in Question 4 but then 'only' somewhat agree in Question 5. This trend among participants suggests that whilst the application identified and tackled the lack of trust between users on social media to a degree of success, there was still more that could be done.

This finding tends to agree one of the additional comments made by the participants, which stated *'It would be useful to see the trust rating for the people that have written comments'*. This feedback was rather significant, as, in hindsight, it can be seen as quite a large oversight of the project. The comments feature has been largely ignored whilst most social media applications do tend to feature ways to engage with comments in the form of likes, upvotes etc., which is important as comments can also be a potential source of misinformation. This feature would be easy to implement, considering the voting logic would be identical to that of the *post* component.

Another interesting comment made was the suggestion to *'implement an algorithm that automatically moderates posts and removes them or places a warning once they fall below a certain trust threshold'*. This could be a good approach as it would mean if the application had a large userbase, posts could be moderated with little required in the way of human moderation. However, this would also mean that posts could be targeted and unfairly removed by such an automated moderation system. The feature could also be negatively received as it could be seen as a form of censorship to unpopular opinions, thus a system must be put in place that can discern an unpopular opinion to one which poses genuine harm, which may be difficult to implement.

Lastly, multiple comments were made asking *'How is the trust value calculated?',* which was asked by users which did appreciate the trust-based features according to their previous answers. This suggests that while they felt that the trust-value was useful, it would have been

nice to have some transparency as to how this value is actually calculated. Although the calculation is simple, it can be assumed that it would give clarity to the users and give them further information to allow them to process the trustworthiness of content on the application even better.

4.2.2.3. General survey results

Overall, participants of the survey generally viewed the application to be of a high standard. Using the results from question 1 we saw that all participants deemed the application to be average or better when compared to competition, with 73% of those viewing the application as 'above average'.

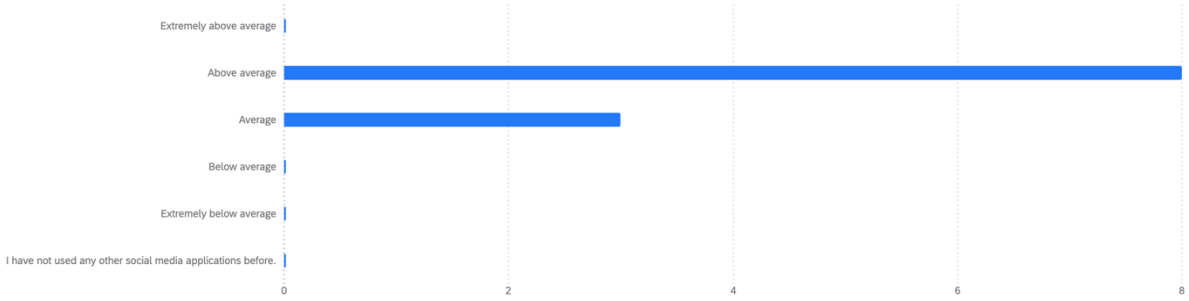


Figure 48. A chart showing the distribution of answers to question 1.

An expected takeaway from multiple respondents to question 8 was that merely placing an emphasis on trust-based features makes the user think more critically about the trustworthiness of the content they have consumed, even if they did not interact with the trust-based features otherwise.

4.2.2.4. Survey summary

Although only 11 participants were able to take part in the survey, the feedback received has been invaluable. Many of the suggestions given for improvements to the project are simple to implement which we would look to do in future work.

The results show that as a whole the project has succeeded in both of the major aspects it had set out to achieve, in producing a high-quality application which integrates features that enhance the trust between users. In future work, we would look to refer back to the feedback given in this survey due to the high quality of responses given.

All survey answers can be found in *section 7.2* within the appendices.

4.3. Technology review

This sub-section will aim to review each major technology used within the project.

4.3.1. Frontend technologies

The technology used for frontend development was primarily React.js, which was chosen due to the developer's prior knowledge of the language. However, the level of knowledge required to undertake the project was underestimated, thus a larger amount of time was attributed to

learning React.js to a sufficient level than was preferred. Regardless, we still believe this was the correct choice, as the large amount of documentation that comes with React.js, as well as the equally large number of tutorials online helped with the extra learning that was required. Once a sufficient level of knowledge was learnt, development was quite easy and the component-based architecture of React.js proved to be the perfect match for the application, as it frequently reused components (which React.js excels at). The developer also leveraged many packages within React.js, accelerating development significantly, which would have not been possible with other libraries.

In regard to the styling of the website, we are again satisfied with our choice. By using SCSS, the program styling code used was much more readable than standard CSS in our opinion, due to its nested nature, which helped accelerate development and identify bugs much faster. The mixins feature was also found to be critical in the implementation of the application's dark mode and mobile optimisation.

4.3.2. Backend technologies

For the backend development of the application, two primary technologies were used, this being MySQL and Node.js. In terms of the knowledge required, the developer already possessed a sufficient enough level of expertise using MySQL and was able to implement this technology easily where needed. On the other hand, the learning curve required for Node.js proved to be quite difficult, as a result, this was where the most time was spent during development, in which the developer had to grasp a very strong fundamental knowledge of the technology to have even very basic things work.

In hindsight, we believe that a longer time should have been attributed to learning Node.js, instead of trying to learn the technology 'on the go' during development. However, we are still please with the technology choice, as the developer has learnt a significant amount about the technologies used, specifically Node.js, which was a primary objective of the project.

4.4. Methodology review

As explained previously, an Agile methodology was utilised throughout the project. We believe this was the correct choice as we were able to swiftly implement feedback given from the supervisor into the application after meetings and email conversations, due to the flexible nature of the methodology. As development of the application only lasted approximately 5 weeks it was necessary to be able to quickly adjust the application when feedback was given.

The Agile methodology complimented the use of *Flying Donut* (The SCRUM planning software used) extremely well, allowing the developer to keep easily keep track of the progress of the project, which was crucial in ensuring no features were forgotten and could be accounted for when estimating the completion time of development. As a result, development was completed on time, thus allowing for sufficient time to complete the report and all other aspects of the project to a level that we are satisfied with.

5. Conclusion

5.1. Meeting objectives

Looking at the objectives set out at the start of the project, we can see that most have been met. Most importantly, we believe the main aim of the project has been satisfied, which was to create a fully functional social media application which possesses features that enhance the trust between users, which has been evidenced through the survey feedback.

Significant research had been undertaken for the project, using a variety of sources. We opted to split research into two primary areas. The first research area involved the identification of real-world problems regarding the lack of trust-based features in social media. The latter involved an analysis of successful real world social media systems. Using the survey feedback received, we believe in implementing the system designed in the project (with the addition of some improvements that were identified in the survey), the problems identified in the background research could be mitigated significantly. We also attribute a significant part of the success of the user interface and application intuitiveness to the system analysis, as this was frequently referred to during development of the UI.

In terms of the structure and planning of the project, we believe a great effort was made, which paid dividends over the lifecycle of the project. By using multiple tools (such as the Flying Donut SCRUM software and Gantt chart) to plan the project we were able to maintain a satisfactory level of progression regarding the development of the application whilst still allocating sufficient time for report writing.

The website designs made using Figma had a positive effect on the implementation of the designs when it came to styling the application, as actual implementation was both easier and faster when being able to reference the designs during development.

As for learning and personal development, this was arguably one of the most significant objectives to meet. The developer had learned a great amount about using both the React.js library and the Node.js runtime environment, which is important as they are some of the most popular technologies at the time of writing, which will serve well in future work. As this was the first large-scale solo project the developer has created, a great amount was learnt in the way of the entire software development lifecycle, which will be of use when the time comes to work alongside others, as a better overall understanding of projects will be had. In addition to this, a valuable lesson was learnt in that it is wise to ensure one is well versed in the language they are to use within the project, as the time taken to learn the fundamentals of a language can easily be underestimated and cut into the time given for actual development.

We believe the algorithms used to calculate a fair and accurate trust rating were implemented well, using both a combination of both weighted average and mean value calculations. The algorithms were implemented succinctly and effectively in various features of the application, such as within the posts of users, users profile page and in their mutual friends tab. However, we hope to design algorithms of higher complexity that take into account more factors, to give trust ratings that are even more accurate, in future work.

GitHub was used throughout the project, who's use proved to be invaluable at points when excessive changes had been made and a previous version of the application was needed, which saved the developer a lot of time. The developer become comfortable in using GitHub

via the terminal within VSCode, which will be a skill that will inevitably be used in future programming projects.

Finally, the feedback received from the survey that was carried out was generally very positive, however we did also get a significant number of suggestions and criticisms of the application as well, which was deemed useful as it has given a perspective of the application from someone not involved in the development. We believe the feedback received was of high quality and will be considered when carrying out future work. The project also had met most requirements stated within the requirements analysis. Due to the quality of feedback and requirements being mostly achieved we can then deem this objective as being successfully met.

5.2. Future work

Whilst we believe this project has achieved its goals, we still believe there is a significant amount of work to be done, as such some potential goals have been listed below.

- Implementation of a moderation algorithm for content which uses the trust ratings designed.
- Implementation of chat messaging functionality between users.
- Addition of trust buttons to comments.
- Visualisation of algorithms used through animations, flowcharts etc. which can be found on the application itself.
- Deployment of the application to make it accessible online.

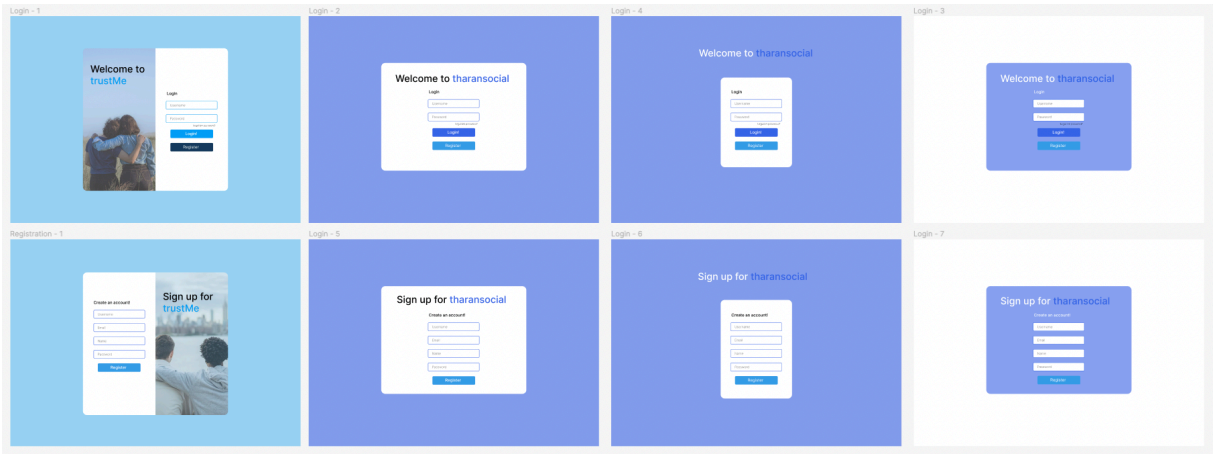
6. References

1. Wikipedia. (2023) *Trust metrics*. Available at: https://en.wikipedia.org/wiki/Trust_metric
2. Amnesty International. (2022) *The Social Atrocity. Meta and the right to remedy for the Rohingya.* Available at: https://www.amnesty.org/en/documents/ASA16/5933/2022/en/
3. C. Lane. (2023) *Social Media 'trust'/'distrust' buttons could reduce spread of misinformation*. Available at: https://www.ucl.ac.uk/news/2023/jun/social-media-trustdistrust-buttons-could-reduce-spread-misinformation
4. Facebook. (2023) *Facebook.* Available at: https://www.facebook.com
5. C. Hu, F. Xu. (2019) *A review of White space Research.* Available at: https://www.scirp.org/pdf/jss_2019032015210256.pdf
6. Twitter. (2023) *Twitter.* Available at: https://www.twitter.com
7. C. Castillo, M. Mendoza, B. Poblete. (2011) *Information Credibility on Twitter.* Available at: https://www.researchgate.net/publication/221023878_Information_credibility_on_Twitter#:~:text=Previous%20research%20has%20shown%20that,and%20false%20rumors%2C%20often%20unintentionally.
8. Reddit. (2023) *Reddit*. Available at: https://www.reddit.com
9. J. Widman. (2022) *What is Reddit?* Available at: https://www.digitaltrends.com/computing/what-is-reddit/
10. Colormatters. (2022) *Basic Color Theory*. Available at: https://www.colormatters.com/color-and-design/basic-color-theory
11. Lamadev. (2023) *react-social-ui*. Available at: https://github.com/safak/youtube2022/tree/react-social-ui
12. GeeksforGeeks. (2022) *Functional vs Non Functional Requirements.* Available at: https://www.geeksforgeeks.org/functional-vs-non-functional-requirements/
13. FlyingDonut (2023) *FlyingDonut*. Available at: https://www.flyingdonut.io
14. GitHub (2023) *GitHub*. Available at: https://www.github.com
15. Figma (2023) *Figma.* Available at: https://www.figma.com
16. Draw.io (2023) *Draw.io*. Available at: https://www.draw.io
17. npmjs. (2023) *Moment.js.* Available at: https://www.npmjs.com/package/moment
18. npmjs. (2023) *Country Locale Map.* Available at: https://www.npmjs.com/package/country-locale-map
19. Testim. (2021) *What is shift left testing? A guide to improving your QA*. Available at: https://www.testim.io/blog/shift-left-testing-guide/
20. Qualtrics. (2023) *QualtricsXM.* Available at: https://www.qualtrics.com/uk/

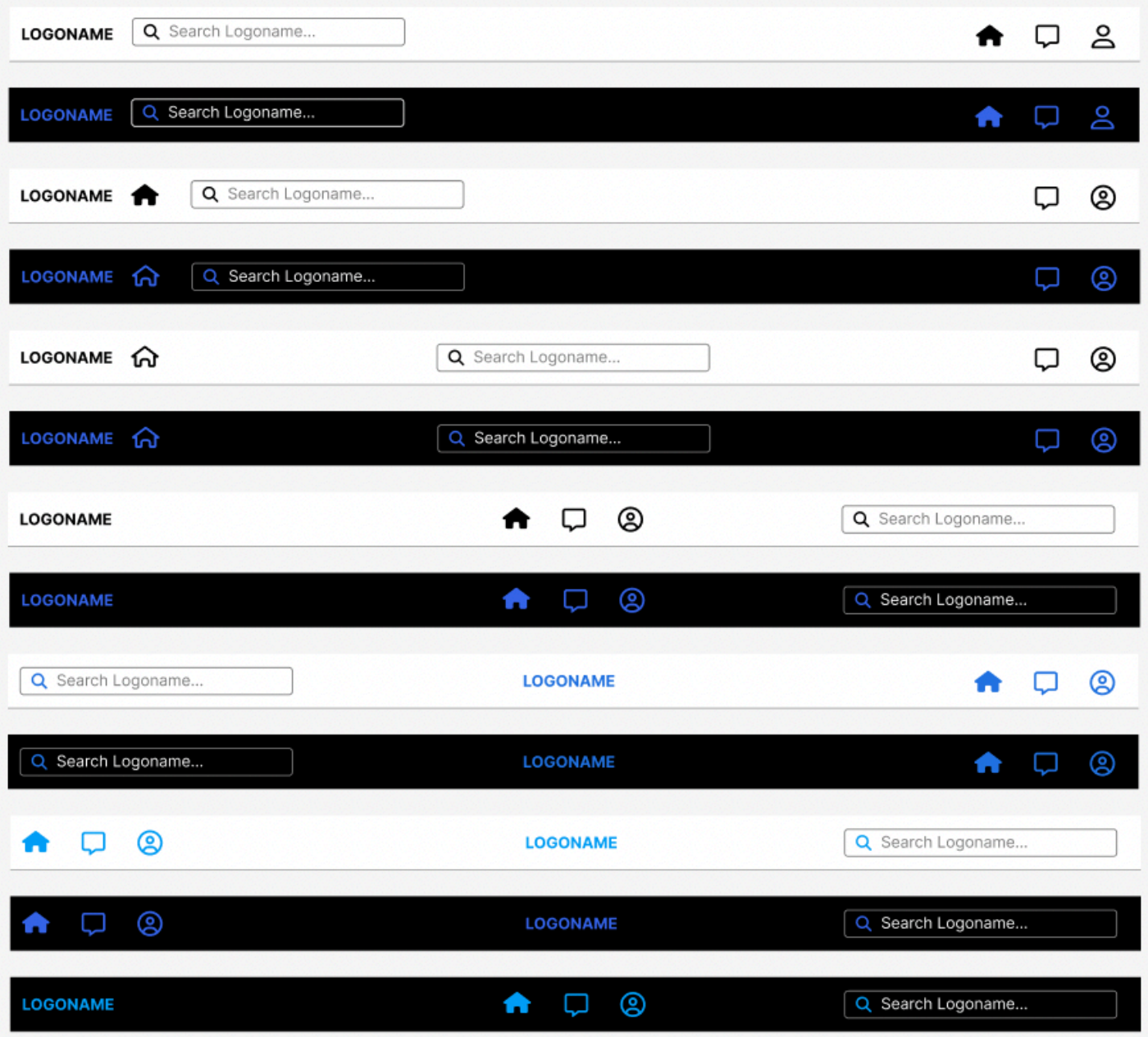
7. Appendices

7.1. Figma Designs

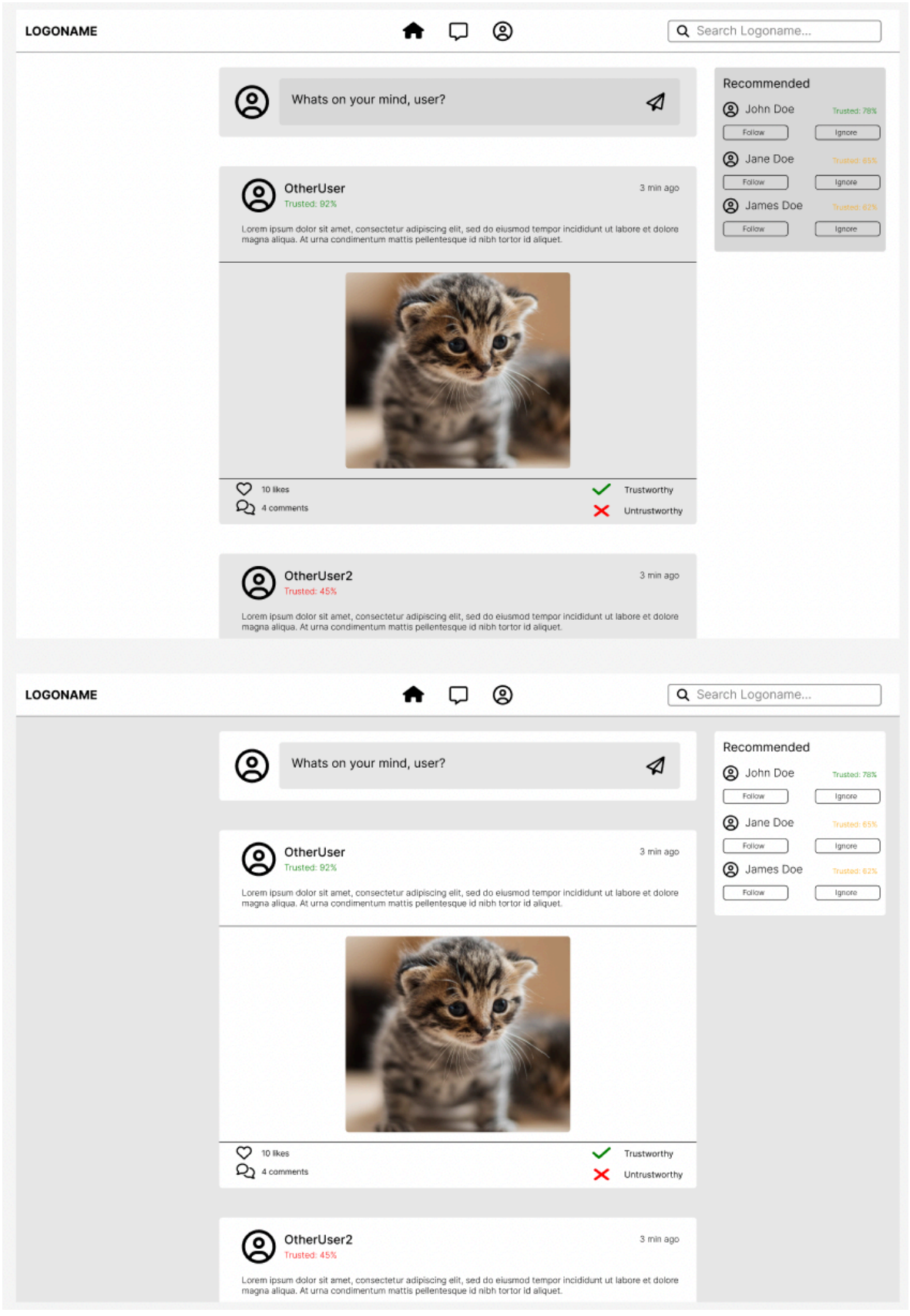
7.1.1. Login and Registration page designs



7.1.2. Navigation bar designs



7.1.3. Homepage designs



7.2. Test List

Test Description	Expected Outcome	Actual Outcome	Result
Invalid registration: Attempt to register a user with a username that already exists in the application.	The registration fails and the user is notified of this.	The registration failed and the user was displayed a message 'User already exists!'.	Pass
Valid registration: Register a user with a username that does not exist in the application.	The registration is successful, and the user is notified and redirected to the login page.	The registration was successful, and the user was displayed a message 'User registered, redirecting to login!'. The user was then automatically redirected to the login page.	Pass
Invalid username: Entering a username that does not belong on the application.	The user is unable to log in and prevented access to the logged-in features of the application.	The user was denied access and a message was displayed making the user aware that the username does not exist on the database.	Pass
Incorrect password: Entering the incorrect password for an existing user.	The user is unable to log in and prevented access to the logged-in features of the application.	The user was denied access and a message was displayed making the user aware that they have entered the incorrect login credentials.	Pass
Valid login: Entering the correct login credentials for an existing user.	The user is logged in and is automatically redirected to their homepage.	The user was logged in and redirected to the application homepage.	Pass
Logout: Clicking the logout button.	The user is logged out, their cookie is removed from local storage, and they are redirected to the login page.	The user was logged out, upon checking the local storage the cookie is changed to NULL and the user is redirected to the login page.	Pass
Navigation bar buttons: Clicking	'trustme' logo – user is redirected to the homepage.	The use was redirected to the	Pass

the buttons on the navigation bar.	Home icon - user is redirected to the homepage. Profile icon – User is redirected to their own profile page.	correct pages for each button.	
Search bar 1: Searching the name of existing users on the application.	The users name appears in the search results with their profile picture when their name is typed in the search bar.	Only the names (and corresponding profile pictures) of the searched users appeared when their name was typed in the search box.	Pass
Search bar 2: Clicking on the name in the search results.	The user is redirected to the profile page of the user who's name they clicked.	The user was redirected to the corresponding profile page of the user who's name they clicked.	Pass
Sharing a post 1: Sharing a post with either textual content, an attached image and both at the same time.	The posts are visible on the user's feed. Posts can be sent as solely text, as an image, or as a pair accompanying each other.	All posts could be found on the post feed in each format.	Pass
Delete post: Deleting a post that the user has made.	The post should be removed from all feeds it previously appeared on.	The post is not visible on any feeds, nor is it present in the database.	Pass
Follow test: Clicking follow on a user's profile.	When test1 follows test2 a relationship should be created in the database.	When test1 follows test2 a relationship is created in the database, where the followerId is that of test1's and the followedId is that of test2's.	Pass
Friends post: Following another user, test2, makes test2's posts appear on the user's own homepage.	test1 follows test2. test1 is able to see test2's posts on their own homepage.	test2's posts appear on test1's homepage.	Pass
Commenting: Writing a comment in the comment section of a post.	The comment is made visible to anyone reading the comment section.	All users that see that post were able to view the comment left.	Pass

Liking posts: Clicking like on a post.	The like icon animates and turns red for the user, the number beside the icon increases by 1.	The like icon animated and turned red for the user. For other users who did not like the post the icon coloured remained white. The like number incremented correctly.	Pass
Positive trust: Clicking positive trust on a post.	The positive trust icon turns green and the trust ratings changes.	The positive trust icon turned green, and the corresponding post trust rating increased.	Pass
Negative trust: Clicking positive trust on a post.	The positive trust icon turns red and the trust ratings changes.	The negative trust icon turned red, and the corresponding post trust rating decreased.	Pass.
Opposite trust: Clicking the opposing trust button when a trust button has already been used.	When clicking positive trust, when the user has already voted the post as untrustworthy, the negative trust vote should be removed and the positive trust vote should be added.	Negative trust vote was removed and the positive trust vote was added, icons changed colours accordingly, post trust vote rating changed correctly.	Pass.
Mutual friend test: Test to see if mutual friend tab displays the correct users.	test1 follows test 2. test2 follows test3, however, test1 does not follow test3. Therefore, test3 should be recommended to the user in the mutual friend tab.	The user test3 was recommended to test1 in the mutual friend tab.	Pass
Update user: Changing user details in update modal.	Any details that are changed in the update modal should change the corresponding fields in the users table in the database.	Multiple fields can be changed at once. Upon update the homepage, posts and other components use the new updated user data.	Pass

Dark mode: Clicking dark mode switch.	The colours of the application should change to that of a dark theme.	The colours change, however, the dark-mode switch should have a smooth animation, which doesn't happen, likely due to a re render preventing the animation from finishing.	Partial pass

7.3. Survey results

Question 1. *How does the application compare with other social media applications you have used?*	
Participant	**Answer**
1	Average
2	Above average
3	Average
4	Above average
5	Above average
6	Above average
7	Above average
8	Above average
9	Above average
10	Average
11	Above average

Question 2. *How would you rate the user interface of the application in terms of usability and aesthetics?*	
Participant	**Answer**
1	Usability – 7, Aesthetics - 7
2	Usability – 8, Aesthetics - 8
3	Usability – 8, Aesthetics - 7
4	Usability – 8, Aesthetics - 8
5	Usability – 9, Aesthetics - 9
6	Usability – 7, Aesthetics - 7
7	Usability – 9, Aesthetics - 8
8	Usability – 7, Aesthetics - 8
9	Usability – 8, Aesthetics - 8
10	Usability – 7, Aesthetics - 8
11	Usability – 9, Aesthetics - 9

Question 3. *Were you able to understand how to navigate through the application and perform various actions without external assistance?*	
Participant	**Answer**
1	Yes
2	Yes

3	Yes
4	Yes
5	Yes
6	Yes
7	Yes
8	Yes
9	Yes
10	Yes
11	Yes

Question 4. *The trustworthiness of content is easier to gauge using this application compared to other social media applications.*	
Participant	**Answer**
1	Neither agree nor disagree
2	Strongly agree
3	Somewhat agree
4	Strongly agree
5	Somewhat agree
6	Strongly agree
7	Strongly agree
8	Somewhat agree
9	Strongly agree
10	Strongly agree
11	Strongly agree

Question 5. *There are a sufficient amount of tools available to vote on the trustworthiness of content in the application.*	
Participant	**Answer**
1	Neither agree nor disagree
2	Strongly agree
3	Somewhat agree
4	Strongly agree
5	Somewhat agree
6	Strongly agree
7	Somewhat agree
8	Somewhat agree
9	Somewhat agree
10	Somewhat agree
11	Strongly agree

Question 6. *What features of the application stand out the most?*	
Participant	**Answer**
1	The application as a whole felt refined and to the quality I would expect of social media applications.
2	The window to rate friends is quick and easy to use due to the slider, the dark mode and overall UI is quite impressive.
3	The profile page for users lets me quickly recognise if it is a fake account or someone that cant be trusted.

4	The trust rating of each post is a nice touch which makes you think more about the quality of the post
5	The application was very fluid and smooth to use.
6	The emphasis on trust makes me think critically about how much to trust a post more so than I would usually.
7	The animations were a nice touch.
8	The trustworthy buttons are something i haven't seen on other apps, it makes me think more about the trustworthiness of content on the app.
9	The trust section on the profile page is something i would like to see on other social medias.
10	The dark mode is nice and the trust levels section on the user profile is useful.
11	The post trust rating allows me to quickly understand whether I should trust a post or not.

Question 7. *What other trust-based features would you like to see in this application?*	
Participant	**Answer**
1	An explanation of the way the trust values are calculated would be useful, it would also be good to let the user have some agency over who is able to follow them, as anyone can follow them and rate their trust, which could be abused
2	An algorithm that can automatically moderate posts and remove them or place a warning once they fall below a certain trust threshold would be useful
3	A way to filter posts based on trust.
4	Possibly combine friends trust and post trust to give an overall trust rating?
5	I think the application did a good job of giving me a better idea of which posts and users could be trusted, however I also think it would have been good to implement a checkmark system like the ones used in facebook, instagram etc.
6	If comments had a trust or upvote system.
7	A way to sort posts based on their trust rating.
8	It would be nice to see how the value is calculated instead of just seeing the value
9	Colour indicators for trust ratings would be a good addition, such as red for low ratings, orange for medium ratings, and green for high ratings
10	It would be useful to see the trust rating for the people that have written comments.
11	N/A

Question 8. *Is there anything else you would like to share about your experience in using the application?*	
Participant	**Answer**
1	I feel like the trustworthy buttons may be slightly gimmicky, as there is also a like button, users might get confused and take a liked post to be one that is trustworthy?
2	N/A
3	N/A
4	The application felt quick and responsive to use, it was easy to understand what the trust ratings were representing.

5	How would one go about targeted attacks where many users vote a user as untrustworthy for no good reason
6	The application dark mode was interesting and something I would use often.
7	The image preview when sharing posts is a nice touch and the window for when you rate users was quick and easy to use
8	The application felt nice to use although i couldn't use emojis.
9	Perhaps more customisability in the user profile section, such as the ability to add a description?
10	The application felt very smooth to use!
11	It would be nice to see a complete list of all mutual friends instead of having to cycle through a list.