School of Electronic Engineering
and Computer Science

Final Report

Programme of study:
Computer Science BSc

Project Title:
A Central Hub for Charities

Supervisor:
Dr Mustafa Bozkurt

Student Name:
Sahir Ahmed

Final Year
Undergraduate Project 2021/22

Queen Mary
University of London

Date: 29th July 2022

Abstract

Give to Receive is an application that has been created to create equal opportunities for all charities worldwide as they will have a level playing field to create publicity for themselves. The application is consumer-based and puts the user-experience at the forefront of matters and creates a hassle-free environment when wanting to contribute compared to many different competitors on the market. The purpose of this project is to amalgamate all of the best possible features of internet donations under one platform to provide a stress-free experience to the user. The success of this project will hopefully reflect itself on the amount of overall donations given to charities and educate those consumers who are living in a bubble about the issues that surround them.

Acknowledgements

A special thank you to Dr Mustafa Bozkurt for holding weekly meetings to check up on my progress and making sure that I am happy with my project as well as lending support where and whenever I required it. Thank you to my family and friends for helping me through this stressful period in my life and always being there when I needed them.

Contents

Chapter 1: **Introduction**

This chapter is dedicated to inform the reader of what I hope to achieve from this project, the overall background of where I came up with this project idea as well as paths I must follow in order to have the best possible system at the end.

1.1 Background

It is needless to say, that charities have been part of human culture dating back several millenniums whether it be in the form of taxes or personal contributions. In recent times, donations towards charitable organisations have increased to the point where it is outpacing the rate of inflation and figures show that over 95% of households in the US donated to charity in the year of 2013 (Lauth, 2015).

Charities may present themselves via advertisement, on the internet, or even physical shops/stalls. Ever since the arrival of the internet, dedicated websites have become one of the many reasons as to why charities have been able to boost their presence globally and raise awareness of issues that befall different communities and countries worldwide. From fighting infectious diseases (i.e. Malaria, Diarrhoea) to helping against lifechanging pandemic's and outbreaks (i.e. the Bubonic Plague, Covid-19), charities ensure the welfare of the less fortunate, and support human rights. By donating to a charity, it encourages people to be better human beings along with improving their social skills and reaches out to their softer side when it comes to certain situations that may arise.

1.2 Problem Statement

A problem that is often highlighted is that bigger charity organisation tend to have extra funds which they can use to advertise themselves. This causes them to overshadow smaller non-profit organisations leading to very little to no publicity. This creates rifts between those bigger charities and the smaller charities as the smaller ones have to resort to different forms of marketing (SEO and social media) which has a smaller effect (Rubens, 2020). As charities are there to help and protect the welfare of others, they should **all** have the ability to grow at the **same** rate regardless of their creation date and funds.

Web Applications are an important source of income for many charitable organisations, and in order to maximise potential contributions, it is vital that the user interface is flawless and easy to use as well as interactive. Majority of people experience the "feel-good factor" when they donate but rarely do they see the extent of how much they have helped, nor can they show others what they have partake in. Donations require people to be selfless and is something that needs 100% of the effort to come from the user's themselves and in order to reciprocate the generosity, the user experience should be effortless and motivational as well as mirror the thankfulness.

1.3 Aim

The aim of the project is to create a Web Application, which acts as a **central hub** for charities so visitors could use it to donate to charities they wish to without having to juggle between different websites. It would allow direct payments and would also keep

track of major milestones whilst constantly giving updates and allow the donor to have a portfolio of their achievements.

The key in mind for this project is improving the user experience compared to current systems by improving usability. One way this will happen is by using a minimalistic design and getting rid of unnecessary information. Gamification aspects will also be incorporated to motivate the user to donate to these causes. The simplicity of navigating this application as well as the transparency of funds, should be one of the key factor's as to why donors consistently commit and pledge to make contributions.

The overall aim of this project is to create an application which can meet every user's expectation when wanting to donate and make it simple and straightforward on the user-end. They should be able to find everything they want and carry out whatever task they deem necessary, effortlessly.

To summarise, I wish to create a website which allows charities to sign up to it and have a landing page of which they can accept donations from. Similarly, individual users should also be able to host their own landing page in the name of fundraisers of which they will also be able to accept donations. These donations will be tracked and be presented in a gamified manner, whether it be in the form of a leaderboard or an award, in the hope that it promotes competitiveness against each other and boost donations. Secondly, this website should act as a fair playing field for whichever charity registers to it and gives causes exposure with the hope that they also receive a share of the pie. The main aim of the creation of this website is to boost donations to charitable causes by making it easier and fun on the users.

1.4 Objectives

- Researching Existing platforms and seeing what they excel at and identifying their limitations (e.g. what features have they implemented)
- Reading up on psychological articles to see what motivates the average human to do certain things
- Exploring different gamification tools and checking how this can be included within the software to boost donations
- Finding out key marketing techniques to boost success of the application
- Creating a minimalist and easy-to-use interface promoting trouble-free navigation
- Making sure application is secure and keeping the data secure if entrusted with it by following laws and regulations
- Retrieving Feedback from user's after carrying out intensive testing to see what is good and what can be improved
- Preparing for possible risks and obstacles which I may encounter in advance to decrease system downtimes
- Developing an application which is as easy for the consumers as it is towards the charities to hold fundraisers themselves

- Securing payments and being able to manage funds so unauthorized personnel cannot access it

1.5 Research Questions

These questions will be kept into consideration throughout the development of the project:

1. How can gamification be used to award people and boost overall donations?
2. To what extent can I use a minimalistic design to present my interface in a clutter-free manner?
3. How does portability affect frequent uses of an application?
4. What technologies will be used to develop this project?

Chapter 2: **Literature Review**

Since 1991, the birthdate of the first web page, web sites have been used widely to represent companies and organisations in order to connect different individuals. Along with this, several studies show that there has been major advances in the field of cognitive science relating to how applications on the internet and their elements interact with human nature.

2.1 Literature Review

2.1.1 Use of Gamification for applications

Gamification is the process of incorporating game-like experiences in a non-game environment to bring virtual value to the user. It should be seen as an experience rather than a method (Krig, 2019). The key takeaway from this is the main aim of a gamified environment is to warrant that the user experience is of utmost priority.

Juho Hamari, a researcher from Tampere University, conducted a 2-year experiment to see whether a gamified element, such as badges, would improve the web activity on their particular site, Sharetribe, which is a local community marketplace (Hamari, 2015). The outcome of the experiments showed that people were more prone to actively use the service during the time badges was implemented (Krig, 2019). Not only does this suggest that it makes people want to use the service, but it also tells us that gamified elements make users more willing to consistently use the service which is great for our scenario where maximum donations are necessary to have the biggest effect on worldwide issues.

It is one idea implementing gamification elements, but it is a completely different idea when overdoing the service with these gamification elements, especially when certain mini games are disliked, and some people prefer getting straight to the point. Gamification elements should be used wisely and in the right places, but it is not required for every organisation to have the same effect (Krig, 2019). Unless the gamified elements provided a service towards the user's, it is commonly seen as pointless and meaningless.

Gamification has been one of many emerging factors which is useful for user retention as well as user engagement (Krig, 2019). Even though donations can be seen as quite straightforward, by maintaining a user on the website, you can easily use certain design features to persuade consumers to give multiple contributions in the right context.

2.1.2 Effective Web Designs

It is often said that 'high-quality content, ease of use, speed and frequency of updates' are 4 of the key factors used to attract visitors to a website (Deborah E. Rosen, 2004). With a charity website, the aim is to make the most noise about the problems currently going on in a good way and the next step after that is to use marketing strategies to motivate them to go out of their way and press that 'donate' button. It doesn't end here however because effective web designs should also have the ability to convert normal web surfers to repeat visitors (Deborah E. Rosen, 2004) (Katerattanakul, 2002).

To maximise user retention, web designs should involve creating a special bond between the user and the organisation itself and evidence tells us that sensory stimuli is influential when it comes to looking at how much a user values a website (Deborah E. Rosen, 2004). From studies carried out, using minimalism as a general theme helps put greater

attention to those elements on the web page which you want the user to see (Deborah E. Rosen, 2004). The main issue with modern web designs is the information overload which puts visitors off from revisiting so it should be kept quite simplistic.

Roger Bennet from the London Metropolitan University tells us that the higher the frequency a certain individual donates, they are more prone to make an impulse decision when donating as they feel more relaxed the subsequent times they do it (Bennet, 2009). This informs us of the importance of user retention in the charity space as it could lead to exponential contributions which is what every charity desires.

2.2 Existing Systems

2.2.1 JustGiving

'JustGiving is the world's most trusted platform for online giving. They help people raise money for the charities and people they care about the most.' (JustGiving, 2021). The JustGiving website allows the visitor to fundraise for an event, yourself or for charity as well as donate to a plethora of different charities. It is said that over 95% of British postcodes have raised money or donated on JustGiving. (JustGiving, 2021)

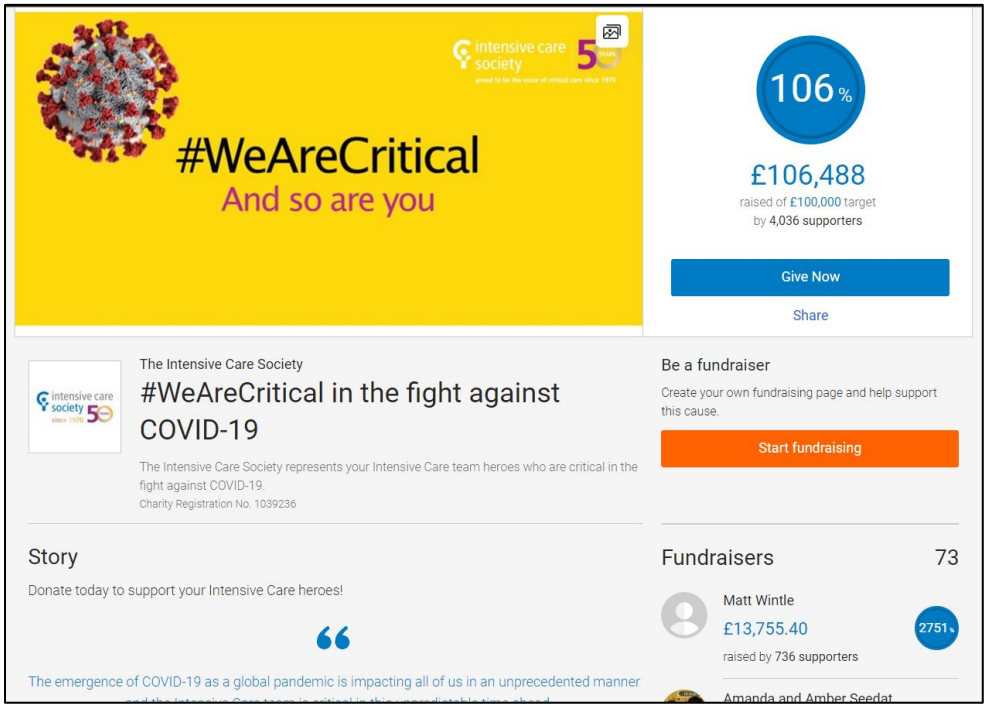
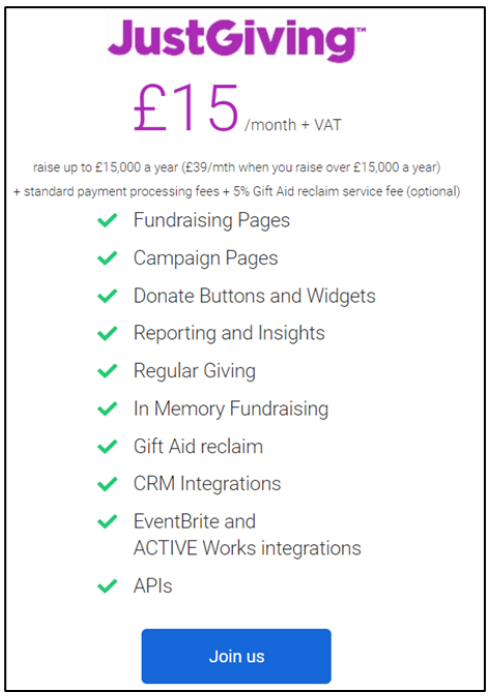


Figure 1. Screenshot of example JustGiving donation page (JustGiving, 2021)

JustGiving gives the visitor a variety of options to try and raise money. It is quite easy to create and manage pages and they provide the user widgets via API's which can be hosted on a personal page for example. They supply you with many different tools to try and raise funds such as extending the duration of a page for however long needed, supporting different currencies, permitting multiple methods of payment (PayPal and Debit) and customisability options. Another great feature which is implemented is that charities and events are sorted in categories which is a great way to filter and finally, they sometimes may dedicate a section for success stories (not all the time) (Children in Crossfire, 2021).



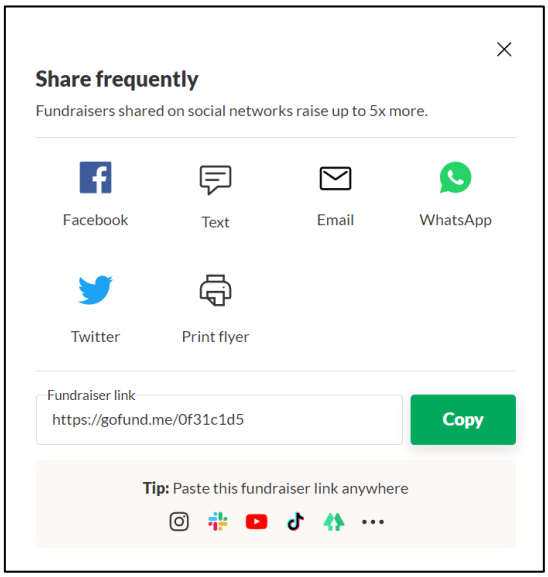
The downsides of JustGiving is that they say they support 25,788 charities but when checking the donation page, there are approximately 20 listed which I believe are used in rotation depending on the current world issues (JustGiving, 2021). Furthermore platforms such as JustGiving and Virgin Money Giving carry huge platform fee's if you want to use them which is morally wrong when the intention is to maximise donations. An example of this scenario in action is when Captain Tom Moore raised £32.8 million for NHS causes but fee's totalled up to £2 million. (Webb, 2020) The majority of these issues stem from the costs of staff and operations so if there was automation implemented, it would be favourable. One final negative is the idea that though the site does possess a search feature, it is not very extensive and quite populated.

Figure 2. Screenshot of JustGiving costs (JustGiving, 2021)

Whilst JustGiving provides an interactive and simple interface, it does not display all the charities unless you search for it yourself and it does not have a native mobile application port which is inefficient when the goal is to maximise contributions. Furthermore the fees needed to pay for the maintenance of the website defeats the purpose of donations to a small extent and can turn people away.

2.2.2 GoFundMe

Established in 2010, GoFundMe describes itself as the world's largest social fundraising platform. GoFundMe provides people and organisations the equipment they require to boost their causes from the generosity of others (GoFundMe, 2021).



The organisation authorizes the user to run campaigns for almost anything and does not restrict them to a specific cause. This makes it more personable and gives the host more power. GoFundMe gives the user the ability to connect third party applications and share fundraiser's easily which is a key reason as to how they make others aware of them so because it is integrated, it makes it easier for the campaign runner. One great feature GoFundMe has which no other social fundraiser's have is the GoFundMe funding guarantee meaning the donation is protected and if you find out that a donation has been misused, then you can be eligible for a refund by making a claim (GoFundMe, 2021). On top of that, GoFundMe gives the option to cover the missing amount of the contribution to go to the right recipient.

Figure 3. Screenshot of GoFundMe share campaign window

The issue GoFundMe has it that its charity fundraisers can only connect to one benefitting charity per each fundraiser, and you cannot attach multiple at once (GoFundMe, 2021). Another issue that stem from personal fundraiser's is that "charity fundraiser's have lower fee's than personal fundraiser's" (GoFundMe, 2021) which is bad when the main intention was for it to be forwarded to multiple charities. Moreover, there is no guarantee that any campaign hosted on this website will be successful as it requires more effort from the host rather than the website to generate attention towards it (Gaille, 2019). All GoFundMe does is hold the page on its site, but it makes no attempt to publicize it. The community needs to be built entirely by the runner of the campaign which overburden's them.

2.2.3 Free the Ocean

Free the Ocean is a charitable organisation whose main purpose is to build on the sustainability, education and action done on the world's oceans. Their main goal is to reduce plastic pollution (Free the Ocean, 2021).

Free the Ocean uses gamification in a great way to do a charitable act. They prompt the visitor with a trivia question and if they get it correct, it funds the removal of one piece of plastic from the ocean by the use of advertising revenue (Free the Ocean, 2021). The great thing about it is that the ad is not pushed to the user, rather to an unknown third party. In addition to this, they allow you to create an account where you can track how many pieces of plastic have been removed and give you option to buy a premium membership which doubles your plastic removal! They also have a shopping store where a portion of the profits go towards running the site and removal of plastic (Free the Ocean, 2021). This is a great example of a charity diversifying their income stream.

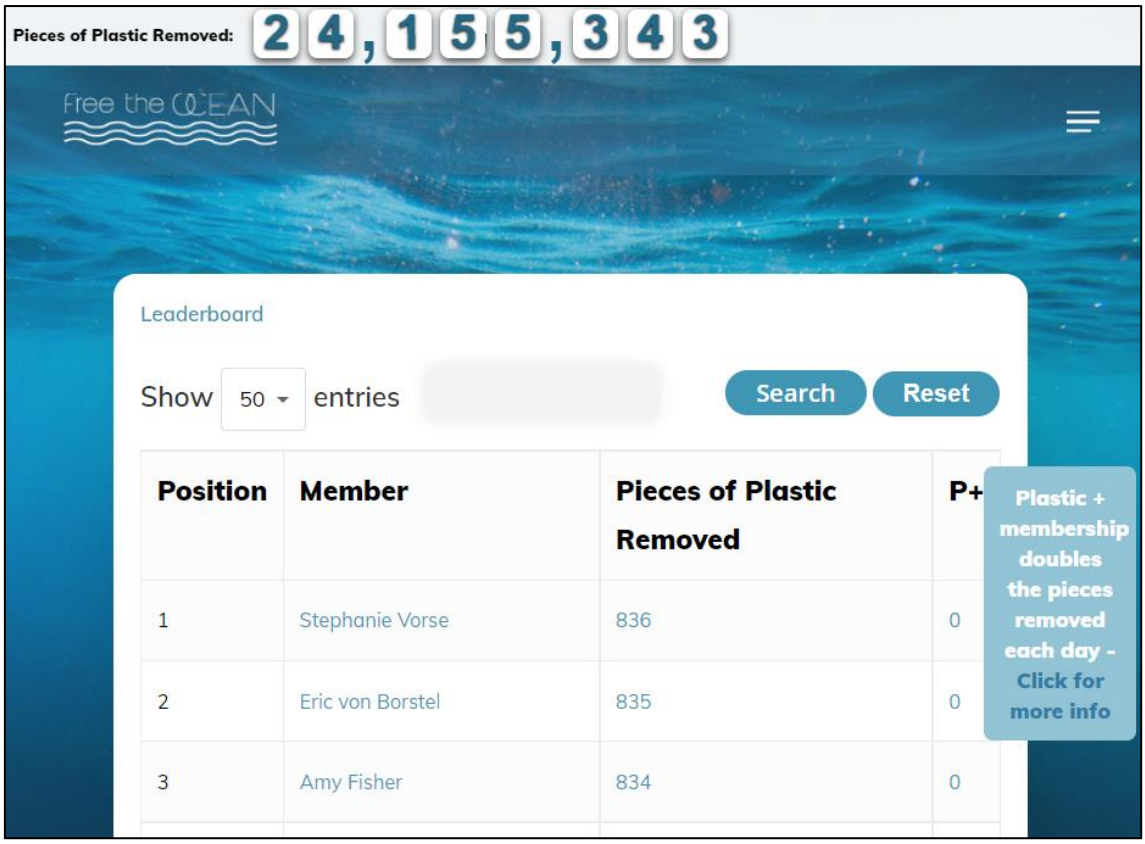


Figure 4. Screenshot of Plastic Removal Leaderboard

One major disadvantage of this application is that whilst scouring the site, it does not let you donate directly to it rather it restricts you to the trivia questions however there are alternative charities that do this.

For the functions it is intended to do, it carries them out really well however it could have some more added functionalities in order to maximise donations. Otherwise, the interface is really good and encapsulates itself directly with what it is trying to achieve.

Chapter 3: **Analysis**

3.1 Requirements Analysis

3.1.1 Functional

Number	Requirement	Expected Outcome	Result – If no, why not? (Ch.6)
1.	The system should allow users to create accounts: *Guest, Individual, Charities*	Guest users can create a charity account or an individual account	Success
2.	The system should allow *individuals* to delete accounts.	Once an account is deleted, they are removed from the database	Success
3.	The system should allow *users* to sign in/sign out.	Cookies should be applied/removed depending on the action and authorise them to permit certain actions	Success
4.	*All users* (except visitors) should be allowed to host fundraising pages for themselves.	Visible fundraising pages on the website	Success
5.	*Individuals* should be able to delete/close their own personal fundraising pages from donation.	Once they delete/close, either should be removed from database or hidden from view	Success – they cannot delete it but rather hide it from view
6.	System should delete fundraiser if *individual* account is deleted	Fundraiser should be removed from the database	Success

7.	*All users* (except visitors) should be able to edit their own personal fundraising pages. -Description -Milestones -Different Goals	Can go to edit page and apply changes	Success
8.	*All users* should be able to view dedicated charity pages and fundraisers	Can visit the landing page of the cause with the right information	Success
9.	The system should allow donations (*all users* except charities) -Should be able to give anonymous contributions as well as donate without needing to sign in -All users can give one-off payments.	Payments should go through	Success
10.	The system should be able to accept recurring donations (all users except charities)	Payments should go through every set duration	Fail – requires storing payment information which may not be secure and with PayPal, it is already quick to do and doesn't take long
11.	The system should have the ability to process *all* and different types of payments.	PayPal, Credit Cards, Debit Cards should all be accepted	Success – allows PayPal as well as credit/debit card
12.	The system should incorporate a leaderboard for *charities* on the system based on how much they have raised	Leaderboard of charities should be ranked on total money raised	Success
13.	The system should incorporate a leaderboard for *fundraisers* on the system based on how much they have raised	Leaderboard of fundraisers should be ranked on total money raised	Success
14.	The system should incorporate a leaderboard for *individuals* on the system	Leaderboard of individuals should be ranked	Success

	based on how much they have donated	on total money donated	
15.	The system should award badges when milestones have been met (*individuals and charities*)	Ranking/award should be received upon hitting milestone	Success – not exactly awards but rankings used which is displayed in their profile dashboard
16.	The system should allow *individuals* to partake in minigames	Only individuals should have access to the page whereas others are restricted	Success – minigame is a quiz
17.	The system should allow you to search for a charity/fundraiser (*all users*)	Search bar which narrows down results to fewer	Fail – not enough time. Would be great for scalability purposes.
18.	The system should prevent empty records from being created within the system	No empty records within database	Success – all database queries are secure and not empty

3.1.2 Non-Functional

Number	Requirement	Result – If no, why not? (Ch.6)
19.	Application must be responsive for both web development along with possible mobile development. -Should be compatible regardless of the platform	Success – initial plan was to possibly use react native, however multimedia queries proved to be sufficient and make it responsive
20.	Data which is intended to be private should be stored securely -By possibly implementing 2FA.	Success – only data which needs to stay secure is the password and this gets hashed
21.	Application should respond quickly when navigating through the different pages.	Success – use of asynchronous API calls ensure quick responses
22.	Password validation should be implemented for security purposes.	Success – authentication in place and confirms password
23.	A timeout system which automatically signs people out when they leave the	Fail – would have to set this in the cookie but since it is not

	session or are inactive should be added.	deployed, not a high priority at the moment
24.	Application should not hog system resources and should be compatible with all devices regardless of specification.	Success – very lightweight
25.	System downtime should be absolutely minimal. Could implement dynamic updates.	? – Downtime could not be tracked as it was not deployed but when it was being used locally, there was no downtime
26.	Localization should be implemented so people from all over the world can donate in a language and currency they understand.	Success – even though the API has it set to GBP, it automatically converts it to your currency
27.	Payments should be secure and quick and only be used with trusted third-party apps.	Success – PayPal makes sure that it is secure and snappy.
28.	Frontend should consistently be syncing with backend to ensure data integrity.	Success – separate API calls have been used each time to execute one function
29.	The system should minimise transaction/platform fees.	Success – literally no platform fees from the website. Only fees included are the default PayPal transaction fees.
30.	The system should not duplicate code if not necessary and maximise reusability	Success – React prevents due to the reusability of components
31.	The system should provide a simple and minimalistic interface which is easy to use and clutter-free.	Success – Interface is very clean and only has the relevant information

3.2 Risk Analysis

Description of Risk	Impact of Risk	Likelihood Rating	Impact Rating	Preventative Actions
Unable to make progress due to illnesses	Opportunity cost of making progress due to recovering. The service may be unfinished, and	Medium	High	Carry out tasks well in advance and do not leave till last minute so you are ready for

	schedule will have to be revised.			unforeseen circumstances. Set priorities on requirements.
Low morale and lack of motivation	Overall decrease of productivity and may inspire deferring of project development.	Medium	Medium	Focus on the smaller tasks and write code in a modular manner so you can work on different sections. Dedicate days to complete tasks whilst also organising days off to rest.
Poor time management	Less time available for other sections which could result in work of a lower quality or even unfinished work.	High	High	Use weekly meetings with supervisor and view them as soft deadlines to get certain features and functionalities complete. If need be, revise project schedule.
Incomplete requirements	Project will be unfinished and will not get the best marks. Could have a domino effect as some requirements depend on other requirements.	Medium	Medium	Review the functional and non-functional requirements regularly and give them a priority level. Set manageable objectives so I could hit all the requirements.
Application not complying with legislations	Will run into problems and could face prosecution. Possible sanction may also include fines.	Low	High	Before the deployment of the application, be sure to review each and every aspect. Constantly be up to date with any new policies and requirements.

Delays due to steep learning curve	Increased amount of time will be dedicated to learn new languages and frameworks. Less time available for other sections resulting in a low-quality final product.	High	High	Be ahead of the curve and practice in free time. Split time between learning and developing program.
Sub-optimal code	Increased resource usage which may result in poor performance and slow response times.	High	Medium	N/A. Can only try to the best of my ability. Finish the application ahead of time and spend time looking through code and optimising it.
Project gets corrupt	All project data lost and would have to restart from the beginning.	Low	High	Apply for extenuating circumstances. Constantly backup data on several devices and even on the cloud.
Bugs in code	Certain functionalities may cease to work whilst also affecting other features at the same time. Could lead to program breaking.	Medium	Medium	Implement automated test runs to check features work on each new addition and check performance of code after every new addition.
Differing Requirements /Clashes	Can have a huge impact on scheduling of project tasks and may lead to delays and thus leaving less amount of time for other sections.	Low	High	Use the agile methodology or an iterative methodology to get regular demands and be informed of changes in previous requests.

Chapter 4: **Design**

4.1 Use Cases

The charity application has many 3 main actors (guests, individual account, and charity account) and multiple use cases with many more to be added along the way. Examples of some use case include donate, host a fundraising page and search for charities and fundraisers. The full use case diagram can be seen in figure 5.

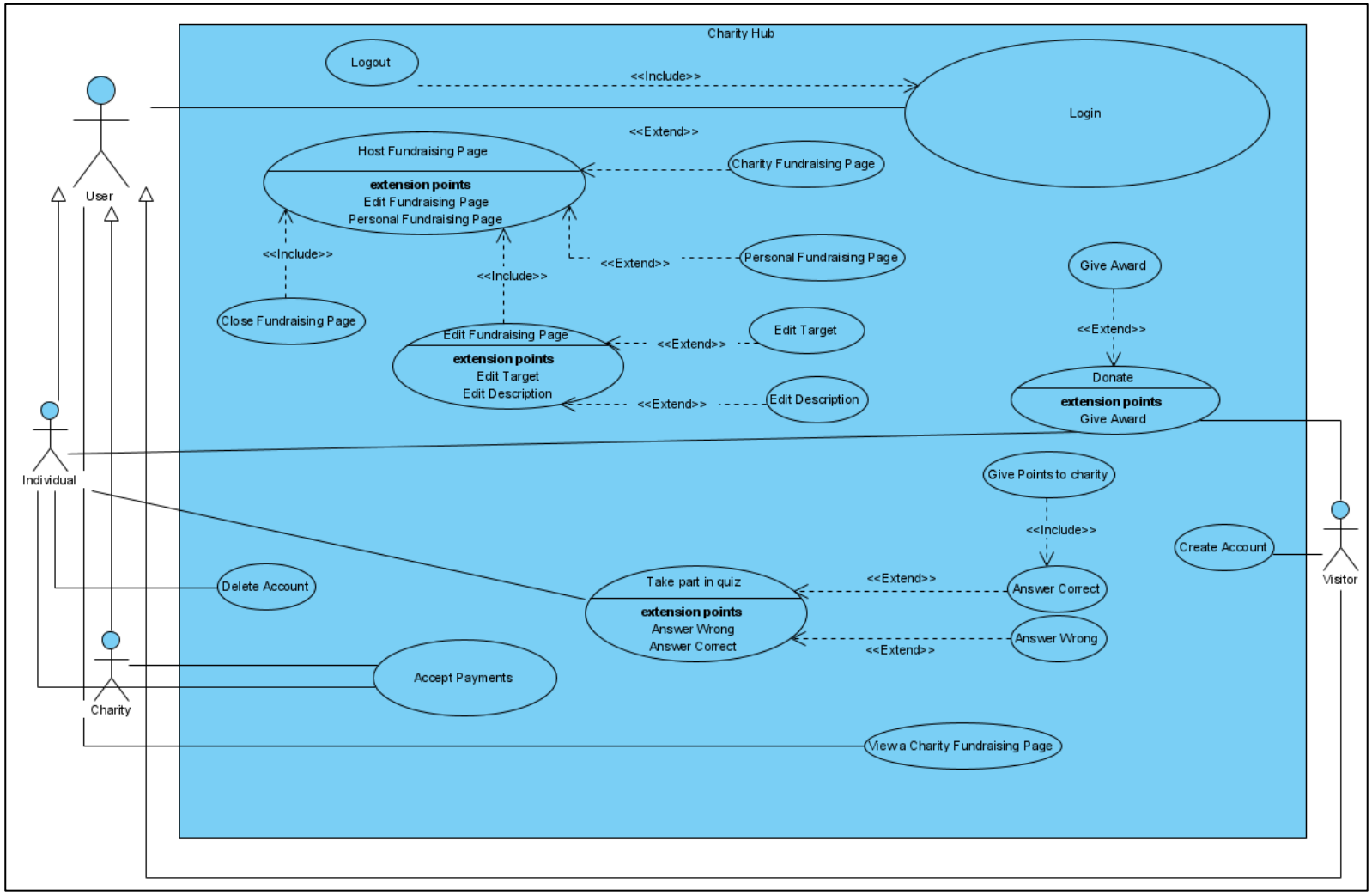


Figure 5. Charity Hub Use Case Diagram

4.2 Database Models

Below is a rough database model with no methods or fields entered just yet. The reason for why there is no additional information is due to it being unclear at this point in time of what information would be required and so will be adjusted as time goes on. It should however provide a basic overview of how the website should work.

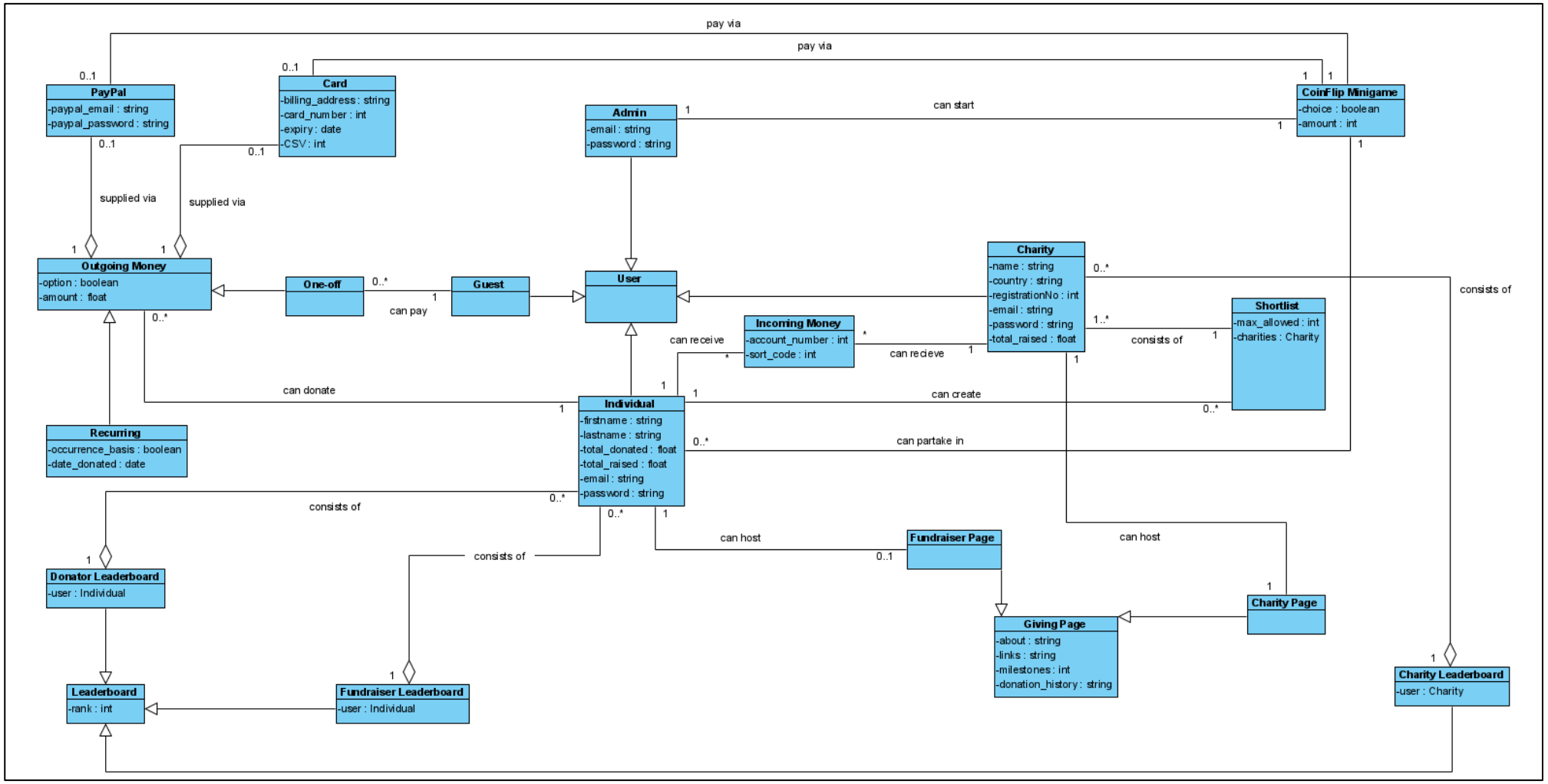


Figure 6. *Charity Hub Class Diagram*

4.3 Design Illustrations

4.3.1 Wireframes

The optimal way for me to develop a program is via illustrations since I am very much a visual learner. The first method of design I followed was by creating wireframes of designs I may aim to see within my charity website.

Some examples can be seen below whereas the rest can be seen within the appendix.

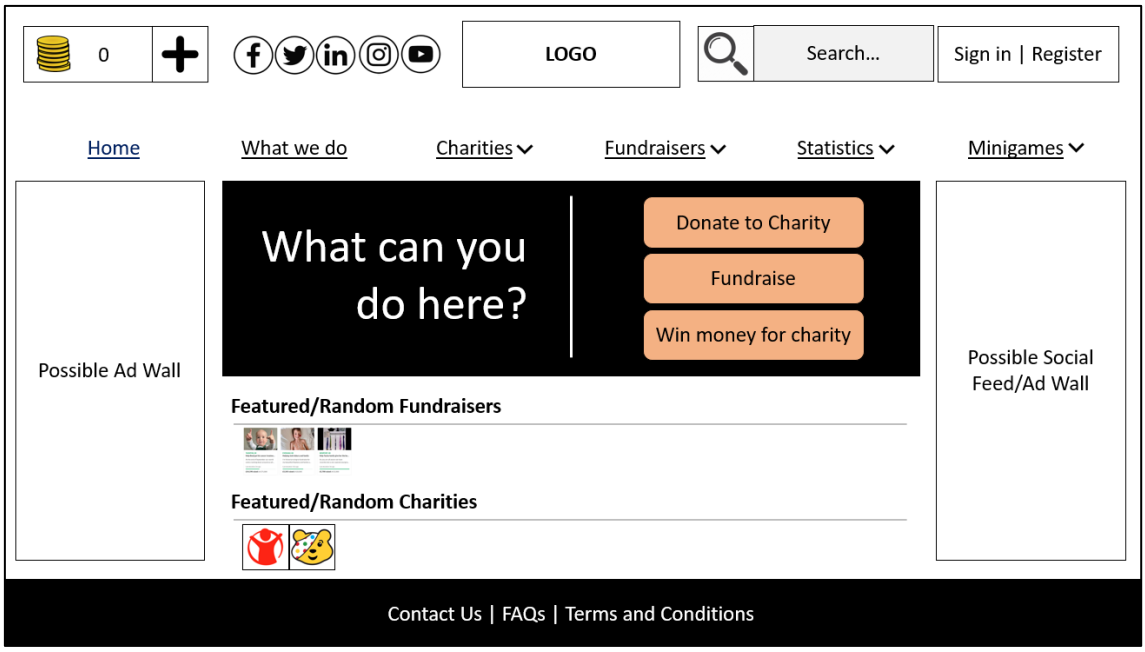


Figure 7. Home Landing Page Wireframe Design

Figure 7 shows a wireframe of how I intend my home landing page to look like. It is pretty self-explanatory. The navbar will allow users to navigate to their preferred page. The system will have a login/logout feature to differentiate the different.

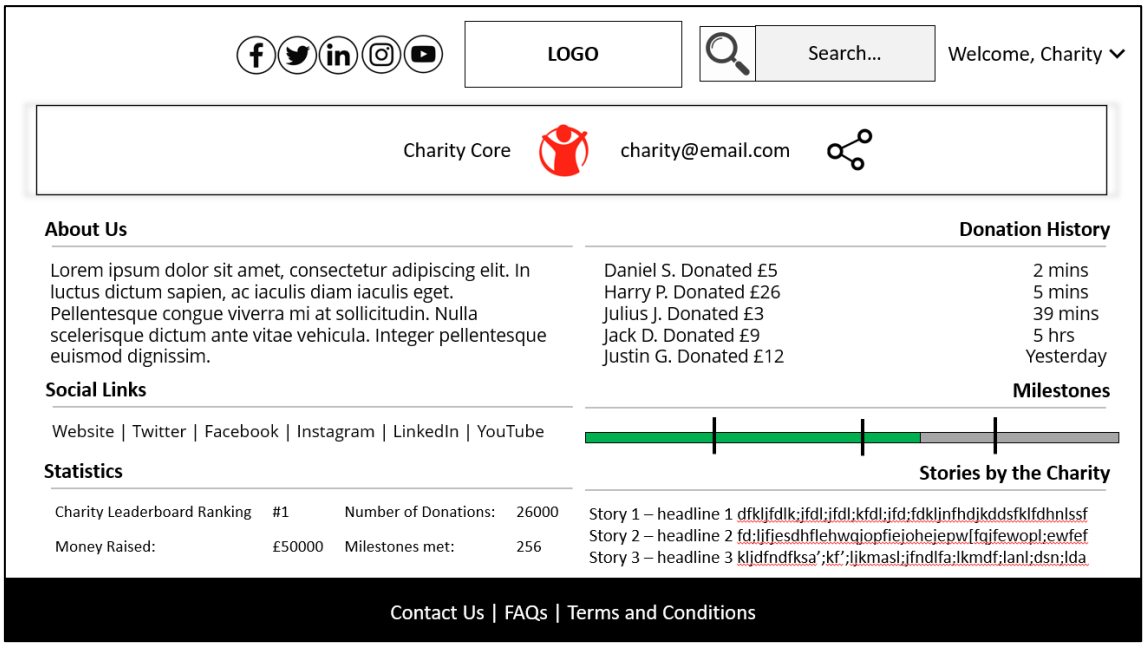


Figure 8. Charity Donation Page Wireframe Design

Figure 8 is another wireframe design which I hope to implement, and this webpage will hold information regarding the charity/fundraiser in question. As can be seen, there are many components to each website which may be used to provide users with an abundance of information, so they feel more inclusive with each ongoing project. Features such as the milestone bar and statistics would be great indicators to inform users how successful each campaign is.

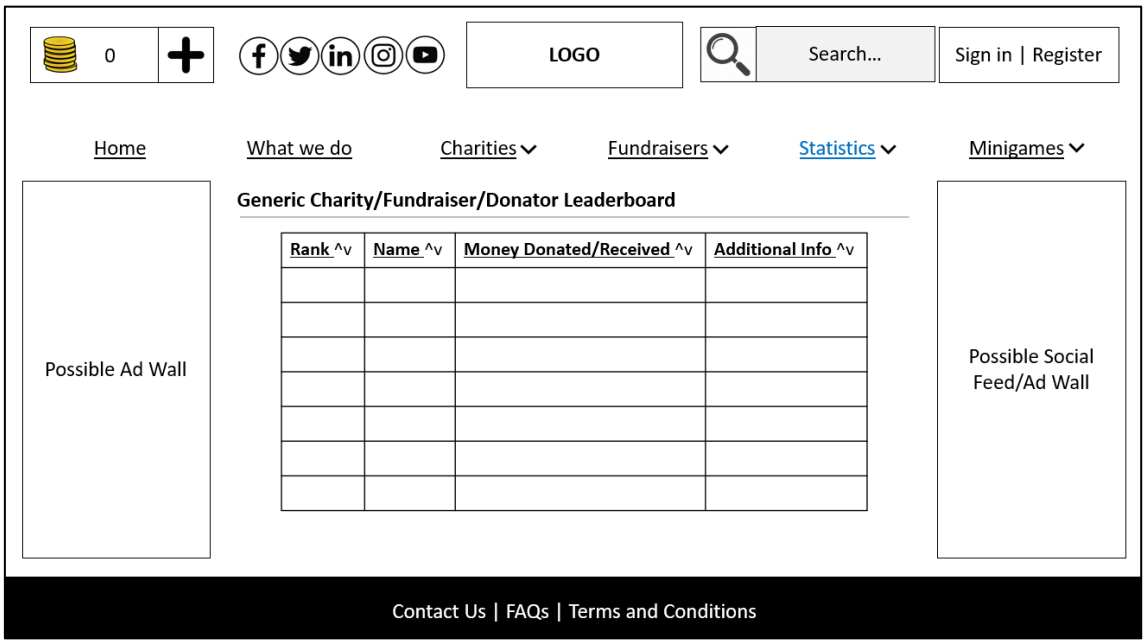


Figure 9. Leaderboard for Donations

A leaderboard system, as seen in figure 9, would be a great incentive for users to donate more frequently as they can see how they match up with other users on the system.

Disclaimer: With these wireframes, it can be seen initial ideas consisted of creating an in-website currency which can be used in the minigames. However, due to security concerns, and the withholding of money on the website, this idea has been scrapped. Minigames would now consist of a quiz instead.

4.3.2 Gamification Elements

As mentioned earlier, the use of gamification elements will hopefully boost donations. There will hopefully a wide array of these elements incorporated within the system

4.3.2.1 Badges

As certain milestones are met and a set amount of donations may be made, or even collected, users should be able to collect badges which they can compare with other users. The aspect of achievements will promote the idea of fulfilment and will hopefully drive-up donations as well as use of the system generally.

4.3.2.2 Leaderboard

Leaderboards will be host to information about each and every account on the system. It will allow users to compare themselves with other users and aim to get on the leaderboard as it will give them exposure. We live in a generation where publicity has become prioritised and so this tool can be seen as a sort of challenge for each and every user. People will have to work hard to appear on the leaderboards and it will give each and every user a fair chance.

4.3.2.3 Minigames

Compared to other fundraising sites, the idea of minigames is very rare. So the introduction of a game within the site could shift attention to the website as it is quite a unique thing to do.

4.4 Chosen Technologies

4.4.1 Languages and Frameworks

4.4.1.1 React JS

React JS is a front-end JavaScript library which is used to build interfaces via the use of components. React is great as it allows you to reuse components and process data without needing to refresh the page. It is fast and very scalable, and very easy to learn. Furthermore, it has multiple versions such as React Native which is useful for porting a software across devices and uses mostly similar language.

4.4.1.2 Node JS

Node JS is an open-source server run time environment which uses asynchronous programming and is useful for eliminating waiting times as it simply carries on with the next request. Because it is non-blocking, it is very memory efficient. Node JS can generate dynamic content and can add, delete, and modify data in databases. It runs scripts on the server to produce dynamic web pages before sending it back to the user. I hope to use NodeJS for backend development as it helps with API's which I will require for my project to source different charities (instinctools, 2021).

4.4.1.3 Express JS

Express JS is a backend framework for NodeJS and is used to build web applications and APIs. It helps with responding to requests with route support so we can write

responses to specific URLs. It is lightweight and helps organise web applications on the server-side into a more organised model-view-controller architecture.

4.4.1.4 MySQL

MySQL is a relational database management system that is based on SQL. It allows you to query databases and retrieve results as well as store data and is easy to use.

4.4.2 Web APIS

PayPal Button API

It has been over two decades since PayPal came about and today; it is one of the most trusted online payment systems and is widely established. It allows you to transfer money as well as receive money and its buyers protection helps safeguard any fraud and relay your money back to you if such a situation may arise. It also gives you an extra layer of security as sellers cannot see your payment information.

The PayPal Button API makes it very easy to transfer money to a client and all that needs to be done ,to make it work, is create an object with the merchant's unique PayPal id and submit a value for how much across. This then gets processed on PayPal's servers and can alert you when payment has gone through.

Chapter 5: **Implementation**

5.1 Connection with SQL database and backend connection

For this project, I was working on the application locally and to access the web page, I used localhost. When configuring a server, you need to state the address where the application is being hosted as well as specify a port. I implemented a file called CORS (cross-origin resource sharing) which benefits the security side of things as it protects the server from being accessed by any other frontend's on other websites which it does not permit access to.

I created another file in the backend (mysql.js) which creates a connection to the SQL Server as well. To do this, you need to provide the username, port, and password of where the SQL database is located as well as the database name so it can extract the correct information from the right location. I then exported this connection so I can use this within other files on the backend.

To properly connect this to the frontend side, express JS is used as it can handle a multitude of http requests and to make it easier to make http requests from the frontend, I inserted a file called (ApiCall.jsx) and this connects both segments (client and server) of the application. This is where all the REST API configuration is held. I decided to use Axios (uses promises) rather than the in-built Fetch API to consume the backend data as we can benefit from easier to read and asynchronous code. Asynchronous code is great for speeding up execution times and to maximise performance/efficiency of the system. Axios also has a built-in features that protects the client side from CSRF (cross-site request forgery).

Within the backend, I also installed express sessions as I will be using this to help create user sessions after they have been authenticated. This would promote authorisation of users before giving them control over certain aspects of my website to which they should or shouldn't have permission over.

```
const baseUrl = "http://localhost:4000";
axios.defaults.withCredentials = true;

const ApiCallGet = (path, redirect = true) => {

  return axios.get(baseUrl + path, { withCredentials: true })
    .then((response) => {
      if (response.status === 202 && redirect)
        window.location.replace(response.data);
      else
        return response;
    })
    .catch((err) => {
      throw err;
    });

}
```

Figure 10. API GET request template using Axios

The image shown above is an example of a template for GET API calls using Axios. Axios makes it very easy for us as all we need to do is specify the URL which we are following and on the backend, we need to create a route which picks up these calls and then manipulates the data as per the developers wishes.

Finally I created another file in the backend (Routes.js) which will handle the API calls from the backend to the frontend and vice versa. This is the file which will manage all the frontend-backend queries and responses and will be used to manipulate the data in the manner which we desire. Since we exported the SQL connection to this file, I was able to use this file to act on user requests and pass on responses based on queries to the database.

5.2 Configuring the frontend

For the frontend, I used react-router DOM which allowed me to have dynamic routing. All of my webpages were given routes, and this could be seen in the App,js file. React is helpful as it prevents the need to duplicate code and therefore each webpage is built via components which each have their own set tasks. These were stored in their own folder. This was useful as it allowed me to mix and match components on a web page. Any external images or resources I needed for my website was stored in a folder called assets.

5.3 Implementation of Wireframe Designs to create the Frontend User Interface

Since I did not have a 100% clear idea on the database models/class diagrams and what information should be stored for each entity just yet, I decided to start work on the frontend and get majority of the frontend done and based on my progress, I was hoping to gain clarity on what data may be required for these models in the backend. So to do this, I focused on trying to implement my wireframe designs using React.js, CSS and HTML.

It is important to mention this was being hardcoded at the time due to me not starting work on the backend at this stage

I started off with setting up the paths and the home path would point to the original directory (localhost) when launching the server. For this directory, I created three components. One was the navbar (visible in majority of the webpages), another was the footer (again visible in majority of the webpages) and finally a video component which was exclusive to the home page.

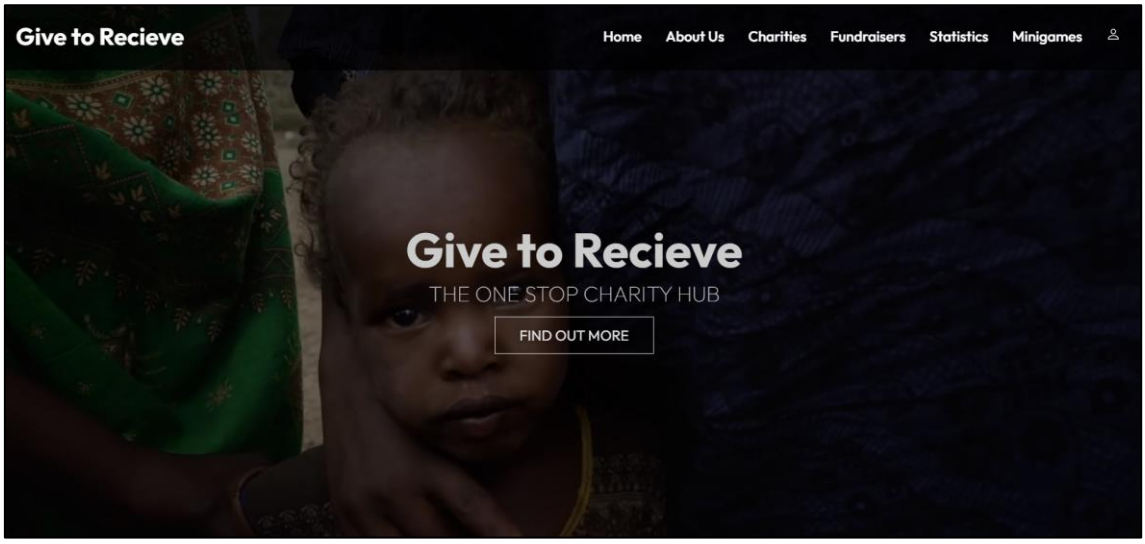


Figure 11. Home page PRE-BACKEND CONNECTION

A similar structure was used to create webpages for the other sections within the website with components being reused if necessary.

The following page which was worked on was the about us section, which would hold informative content explaining why this website was being created. This web page consisted of containing information regarding to my reasonings for this project and goals by the end of the development stage.

The webpage I decided on embarking after the about us page was the charities web page. Here, the charities themselves, upon registering to the system, would have a dedicated page for them where they can setup an income stream and advertise themselves to an extent. This is where they would have been able to inform others about their work, redirect them to their own personal website as well as set milestones and accept donations. This can be seen in figure 13.

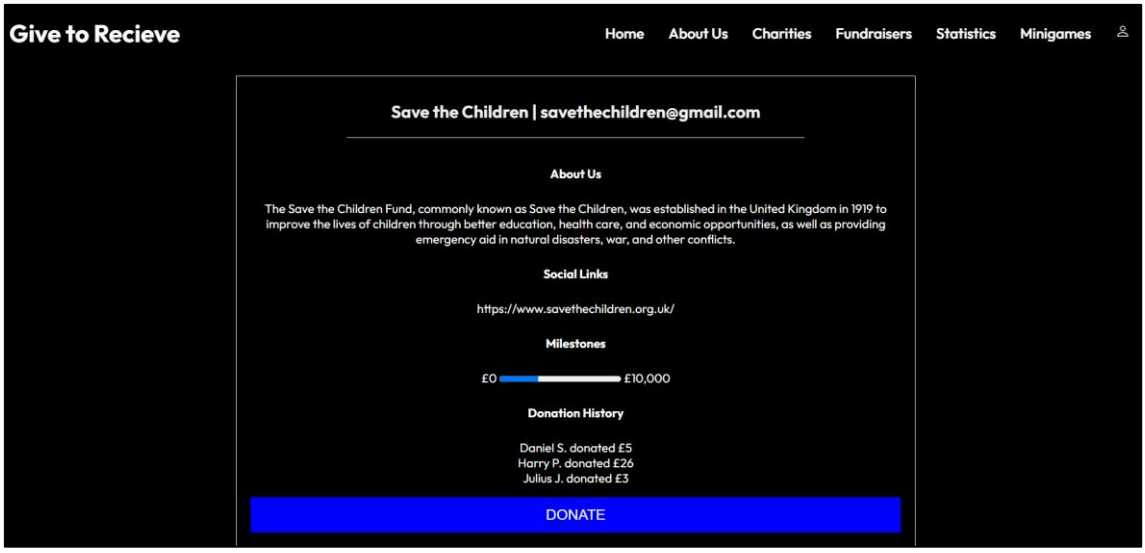


Figure 12. Charity Landing Page PRE-BACKEND CONNECTION

A similar concept was used for the fundraisers section which could be hosted by individual users. Each individual account user would have the possibility to host a maximum of one fundraiser at a time and similar to a charity, they would have the option to customise their landing page according to their liking. But this service may only be used if and when required or the user themselves may altogether prefer not to use it.

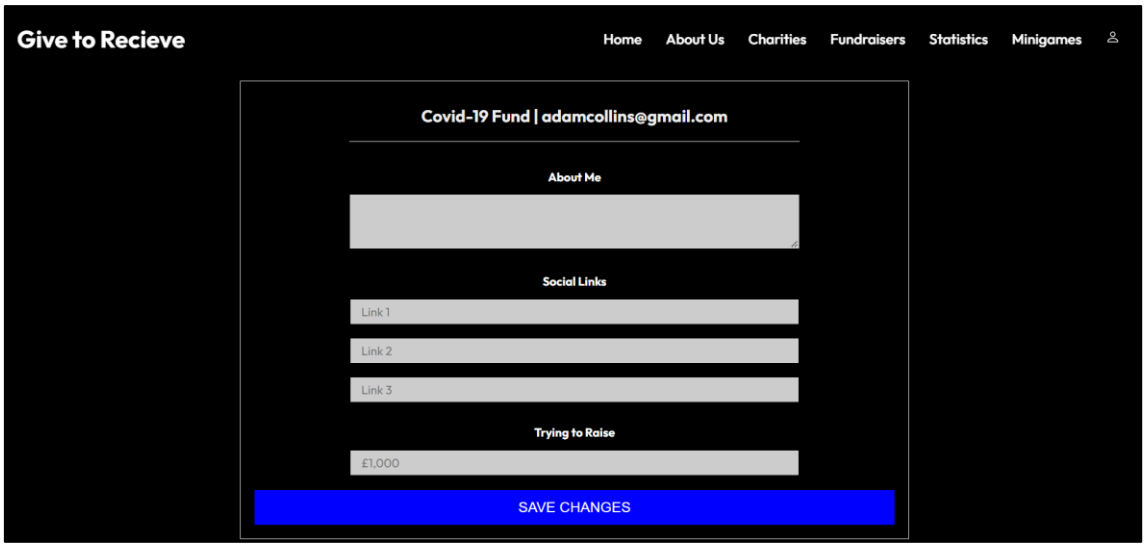


Figure 13. Fundraiser Edit Landing Page PRE-BACKEND CONNECTION

Figure 14 shows how edits would be made to charity landing pages and fundraiser landing pages so users can customise their landing page with their own content.

The final section which I was able to start designing was the statistics section. Here, there would be leaderboards for charities, fundraisers as well as individuals so they can see how they compare to other events/users in the system and try and earn a spot in the leaderboards.

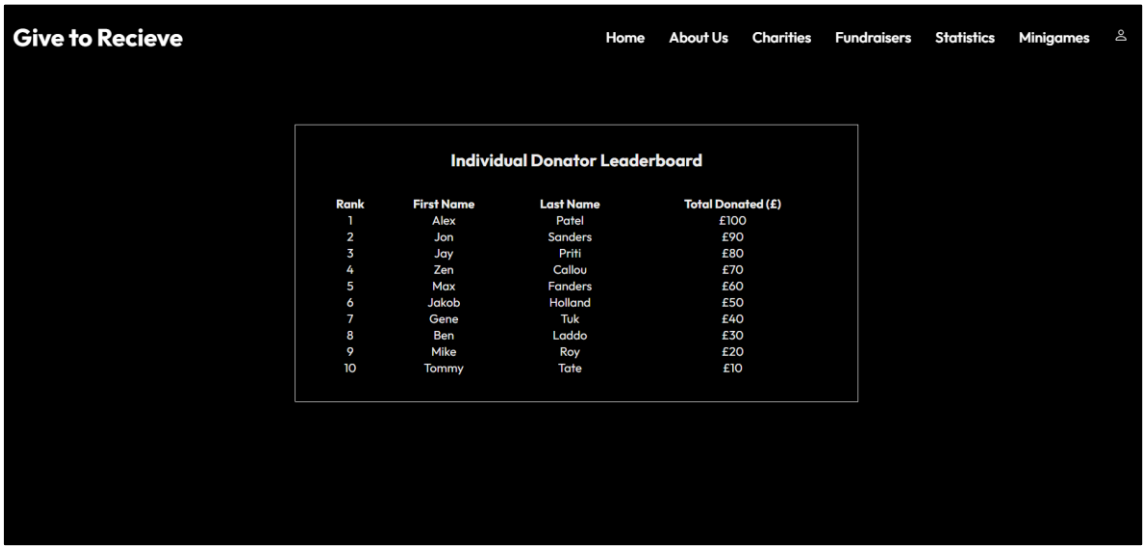


Figure 14. Statistics Leaderboard Page PRE-BACKEND CONNECTION

For now, I left the minigames webpage out of development hoping to return to it at a much later stage due to the complexity of trying to figure out how I could incorporate a payment system.

5.4 Developing a Responsive System

At the start of this report, I have mentioned the idea of having a mobile port to enlarge the consumer base and drive-up donations. Due to the circumstances and size of the

project, a short-term fix that I found that could allow my system to be responsive was the use of multimedia queries when styling the system so if viewed from a mobile device, it should be responsive.

```
@media screen and (max-width: 980px) {
  .card-containerST {
    max-width: 90%;
    margin: auto;
    grid-template-columns: 1fr;
  }
}
```

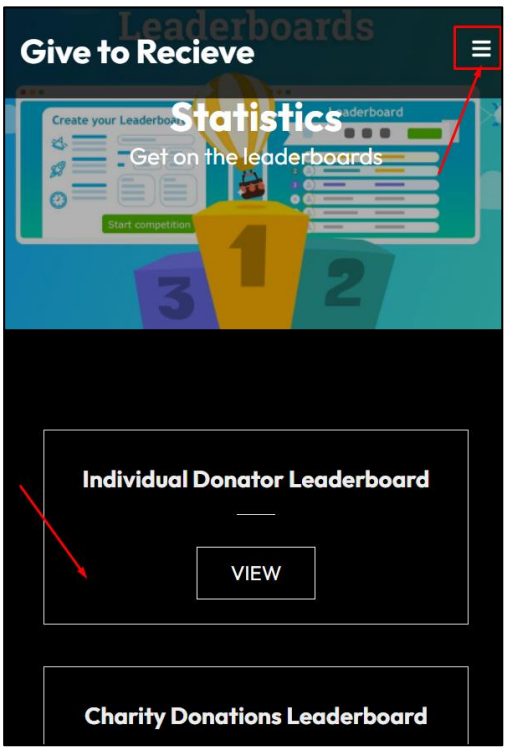


Figure 15. Statistics Page PRE-BACKEND CONNECTION

As can be seen in figure 16, by using multimedia queries, my website is much more responsive towards any device screen. The navigation bar collapses into a hamburger icon on smaller screens and the content on my web page shrinks and rearranges itself to fit the screen snugly.

5.5 Revised Data Model

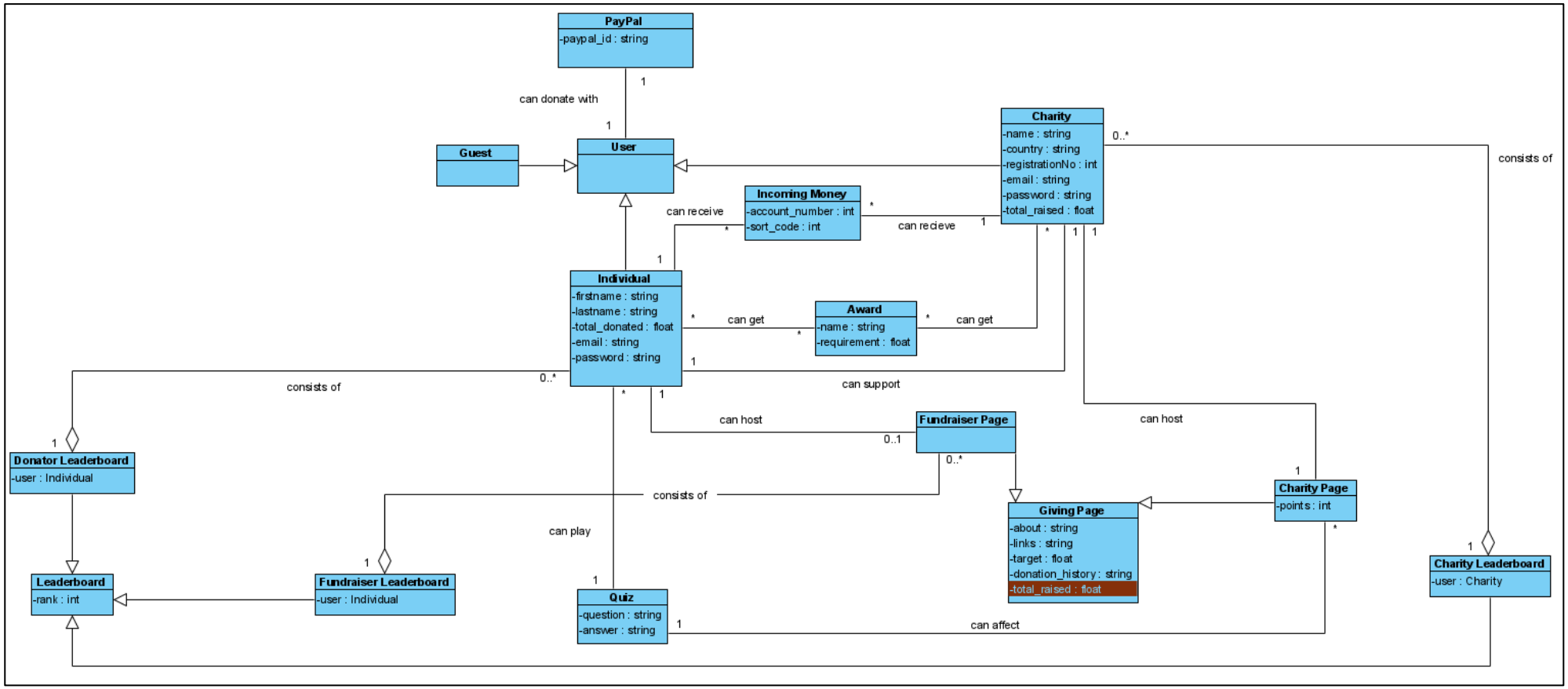


Figure 16. Revised Charity Hub Class Diagram

After completing the user interface on the frontend, the backend needed to be worked on. As I had hoped, by designing the frontend first, I was able to gain a much clearer insight on what data was to be required in order for me to build a charity platform website. Figure 17 shows the updated version of my initial class diagram to better fit the status quo and flow of data throughout the website. Also this class diagram was useful in helping create tables in my database with the right schema.

The coinflip minigame which I had hoped to implement got replaced with a quiz minigame as I believed it was sending the wrong signal out on a donation-based website and promoted the likes of gambling but for a good cause. It also would have been quite difficult to create a new in-house currency and maintain the economy of the system and was moving towards the gaming element rather than staying within the constraints of gamification.

5.6 Creating Databases

With the help of the new class diagram, as well as MySQL workbench, I started to create the databases which were going to be needed for my website. I created four new database tables, and these were the charity user table (figure 18), the individual user table, the fundraisers table, and the quiz table. I needed to change the field datatypes accordingly as well as creating foreign keys to connect the fundraiser, charity, and individual tables all together. Further down the line, sometimes when querying with foreign keys implemented, views were created to amalgamate all of the relevant information in one place and make it easier to pull out key sections of data.

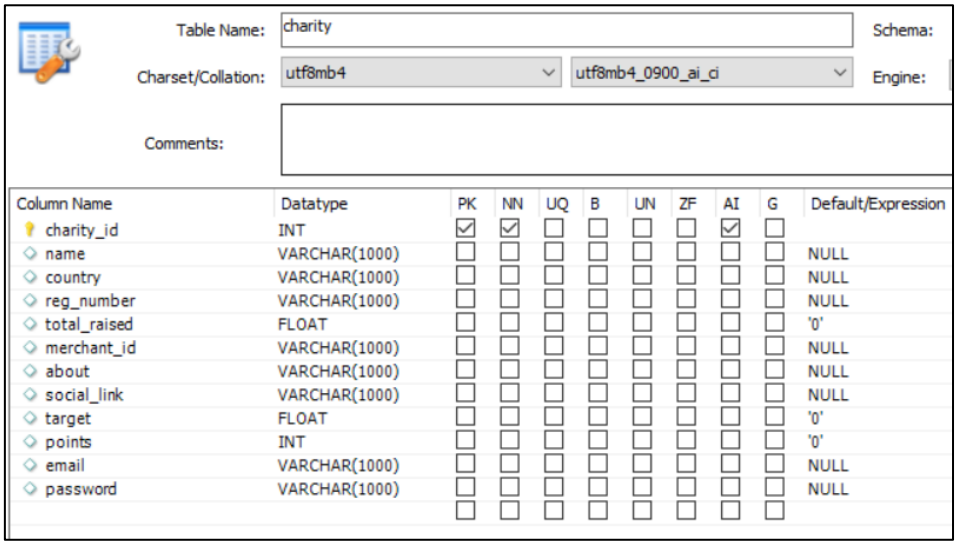


Figure 17. Charity database being created within MySQL workbench

5.7 Registering and Login system authentication

The first section of development which occurred with the backend was the account section. Here, the user was able to create an account (normal/individual or charity) and they were able to login once they had created their account.

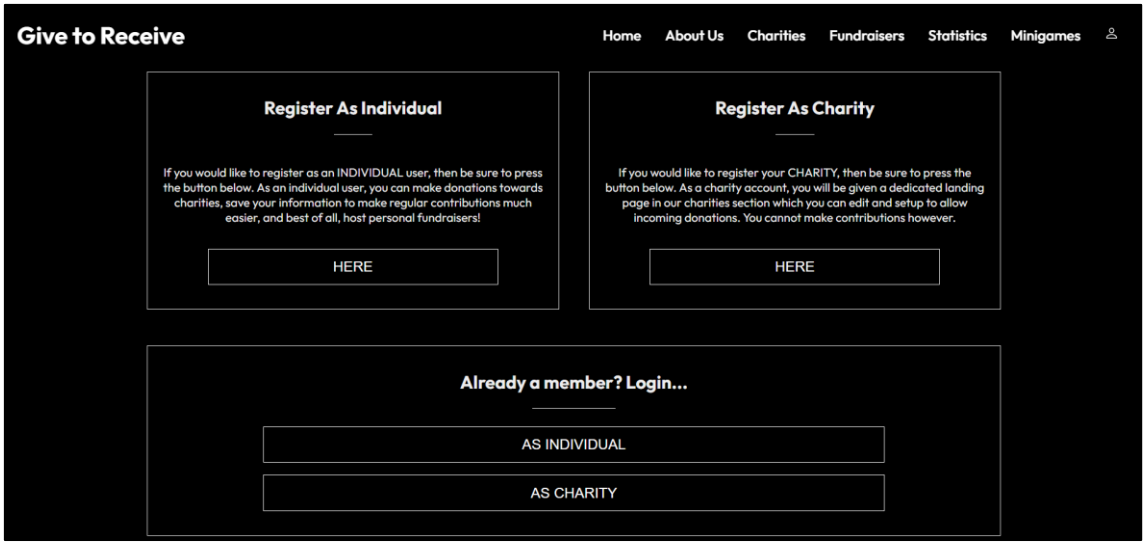


Figure 18. Register/login dashboard

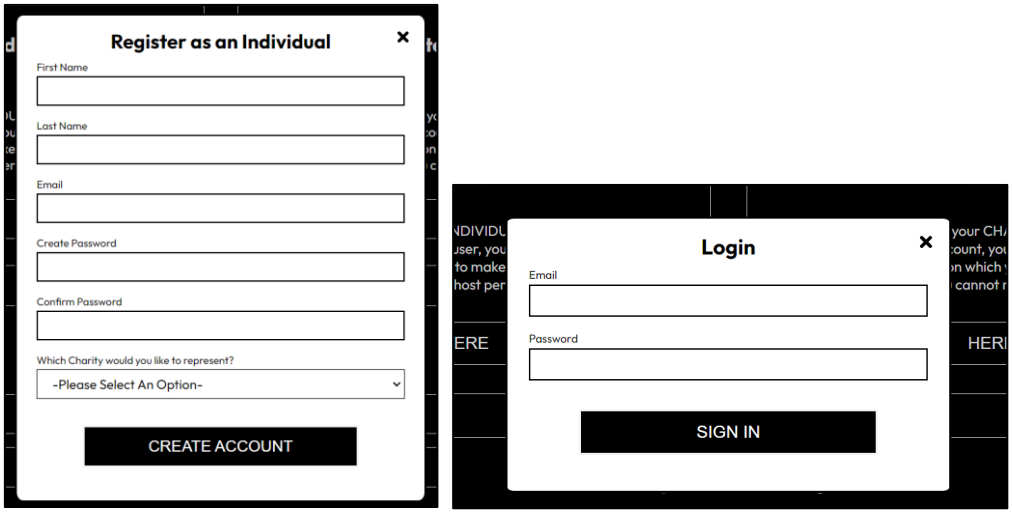


Figure 19. Register and login popup when pressing the buttons

The way this worked was the register function, upon pressing create a new account, would send a POST request using Axios, to the backend with all of the data which was filled out. Via the use of express, I was able to pick up this request with all of the relevant information, and subsequently write a SQL query which created a new record for this within the users table (individual users table if it was a normal member and charity user table if it was a charity registering).

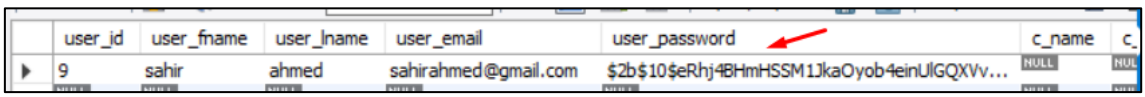


Figure 20. Password hashing when creating a new record in the database

Just before the record was created within the database, the password supplied was hashed using BCRYPT which is a password-hashing algorithm so whenever the user tried logging in, another POST request would be sent to compare the email supplied

along with the password which would be hashed with the data in the database via another query, to verify the user's legitimacy. This is to ensure data is kept secure.

Figure 21. Possible notifications shown when a user attempts to login

If the user entered the correct credentials, they would receive a success notification. However, if they did not, they would receive an error message letting them know that the credentials entered were incorrect.

5.8 Rendering the relevant pages according to the database

As I had gone through the implementation of the website, it can be seen that as mentioned earlier, most of the data thus far had been hard-coded. It was time to change all of the panels/pages on the system to be reflective of what was stored in the database. This part of the implementation stage took the longest amount of time to execute due to the sheer repetitiveness of making API calls to the backend and then manipulating database data to pull the right chunks of data out back to the frontend and displaying it in the correct format.

Most calls made the use of GET requests whereas others POSTED to the database. I also made use of the DELETE request to delete normal users from the system. PATCH was used to update certain fields of a table whenever edits were made.

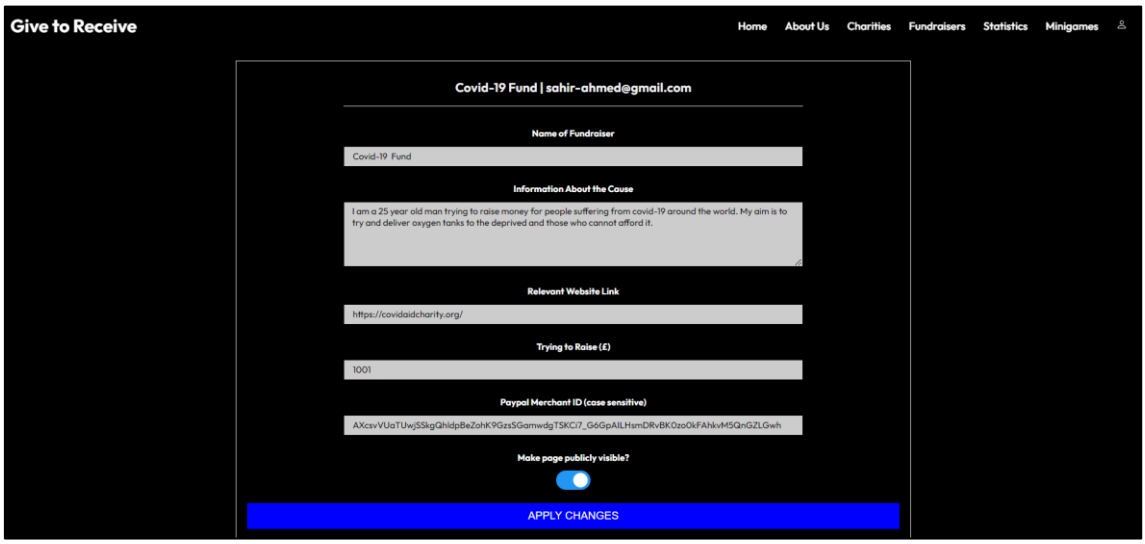


Figure 22. GET requests and PATCH requests were used to render this page

Figure 23 shows an example of where I have used both GET and PATCH http request. A typical API call to the backend looked like:

```
const handleFeditlandingpageGet=async()=>{
  try{
            let response =  await ApiCallGet('/fundraisereditlandingpage/' + userid)
            console.log(response);

            setName(response.data[0].name);
            setEmail(response.data[0].email);
            setAbout(response.data[0].about);
            setSociallink(response.data[0].social_link);
            setTarget(response.data[0].target);
            setMerchantid(response.data[0].merchant_id);

            if (response.data[0].enabled === 1){
              setEnabled(1);
              document.getElementById("tickpublic").checked = true;
            }

    } catch (error) {
```

Figure 23. API GET call to retrieve data from the database

The GET call in figure 24 uses asynchronous code to wait for the API to retrieve the information it needs and render the content from the database in real-time. The response which is given back to the API call is a http response and the data is withheld within the data variable. After accessing the necessary information from this variable, the information is saved in states which can then be used within the html code and shown on the user interface.

```
routes.get('/fundraisereditlandingpage/:id', (req, res) => {

        let query = `SELECT * FROM fundraiserowner
                    WHERE individual_owner_id = ${req.params.id}
                ;`;
```

Figure 24. API GET call from Figure 24 but on the backend

Figure 25 shows what happens in the backend with the API call. Using the information, which was embedded in the URL, we were able to query the database and set conditions. After retrieving results, we then send it back to the frontend which then has to decipher the response and pull the data out.

```
return (
  <div className='clpcard'>
      <div className='card-containerCLP'>
          <div className='cardCLP'>
              <h3>{name} | {email}</h3>
              <span className='barCLP'></span>
```

Figure 25. Using states to render the data retrieved from GET

Finally, figure 26 shows how the information gets rendered on the HTML page. This method had been imitated repeatedly for all the various web pages with slight differences depending on the API call used with some requiring additional data.

By doing the above, my website is now able to present data dynamically from the database and is no longer hard-coded.

5.9 Integrating the PayPal API

At this stage, I had charity landing pages as well as fundraiser landing pages which were dynamically loaded which had donate buttons that were non-functional. The webpage displayed a pop-up asking you how much money the user wanted to donate, however pressing donate had no effect yet.

Early on the start of the project, the first thought that came to mind was saving card information within the database. However, as I am new in this field, and this can be very dangerous to store if not kept secure (this could violate laws and regulations if ever deployed), I decided the best way to allow payments was through a trusted platform such as PayPal or Stripe. This decision removed any chances of recurring payments, but this was not necessary as PayPal makes it very quick and easy to transfer funds.

First of all, to use this API, I had to install the react-PayPal-button library. With this library imported, an object needs to be supplied with the user's PayPal ID or merchant ID. This is the ID they will use to accept payments. You can also supply the currency of the money being sent. The amount of money being donated also needs to be supplied and states were used to enter this amount via the input the user writes in the input field.

Secondly, I had to create sandbox accounts to test this on PayPal's developer website. On the website, you can create sandbox accounts with fake PayPal ID's and imitate payments.

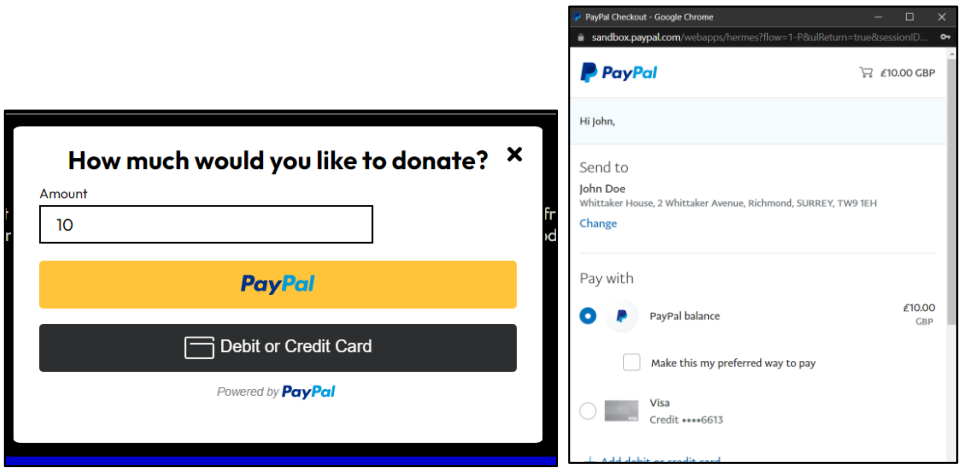


Figure 26. PayPal interface when donating money to a cause

The reason why the PayPal API is great to use is that it gives you the option to donate money via PayPal or even with your debit/credit card. So it circumvents the need to withhold this data within the database and gives you two methods to send money. What is even more spectacular about the way this API has been implemented is that, the money goes directly to the charity/fundraiser owner so there are literally no platform fees other than the minor fees PayPal has.

Upon completion of the transaction, PATCH requests are sent to the database to tally up the money raised by the charity/fundraiser as well as the money donated by the individual. This information would then be used to give out rankings/awards as well as a place in the leaderboards. This is to drive up donations via the competitiveness and comparisons against other people that is instilled within our human nature.

5.10 Creating sessions with cookies

Previously in our implementation, I had set up authentication via the login system. However, the next stage after that was authorisation and authorisation is used to give certain permissions to users that are logged into our system. This prevents tampering of webpages and other functionalities which you should not have access to. For example, we would not want a charity user to be able to donate to causes as they are only registered for receiving funds rather than giving them. This is where cookies come in.

```
req.session.logged_in = true;
req.session.individual = true;
req.session.user = result[0].individual_id;
```

Figure 27. Creating variables within our cookie and setting values upon logging in

When the user logs in to the system, we want to give them a token/indication throughout their duration of the application that lets the system know they are logged in. I implemented this in the backend first by creating a variable within the cookie called 'logged_in' and setting it as true. To determine the type of user, which is logged in, I used another variable. I also passed through the user id with the cookie to show which exact user it is.

```
routes.post('/logout', (req, res) => {
    req.session.destroy();
    res.json({logged_out: true});
});
```

Figure 28. Deleting cookie upon logging out

Similarly to logging in and gaining a cookie, I also created a route in the backend which would destroy the cookie upon logging out and thus removing access. This cookie is essential for validating user actions as well as generating content that is personable to them.

```
routes.get('/checkloggedin', (req,res) => {
    if (req.session.logged_in){
        res.json(req.session);
    } else {
        res.status(401).json({logged_out:true})
    }
});
```

Figure 29. API call in the backend used to see whether the user is logged in at the moment of call

This implementation is fine if all we want to use it is for the login system we implemented earlier. However, if we want to authorize certain actions based on the cookie data, we also need to create another function that checks whether the user is logged in before completing the action, and this can be seen in figure 30.

5.11 Minigames Page and Card Rankings

The final page which was left to complete was the minigames/quiz page. This page retrieved questions from a question bank set in the database and allowed normal users who were logged in to help boost their charity's ranking on the charities page by getting questions correct.

```
const Minigames = () => {

  const[userid, setUserid] = useState(null);
  const[individualacc, setIndividualacc] = useState(null);

  const handleCheckLoggedInStatus=async ()=>{
    let response =  await ApiCallGet('/checkloggedin');
    console.log(response);

    if(response.data.logged_in){
      setUserid(response.data.user)
      setIndividualacc(response.data.individual)
    } else{
      console.log("NOT LOGGED IN")
    }
  }

  useEffect(() => {
    handleCheckLoggedInStatus();
  }, []);


  return (
    <div>
        <NavBar />
        <MinigamesImage heading='Trivia' text='Boost Charity Rankings' />
        {(userid != null) && (individualacc === true) ? <QuizCard userid = {userid} /> : <h1>This section is only available for Individual
```

Figure 30. Code in frontend which checks if user is logged in and displays component if they are the specified audience

Before executing the component which had the quiz, I first wanted to identify the user who was logged in which can be shown by the function. Since only logged in NORMAL users could have access to this page, I needed to restrict other users from using this page. I managed to do this by using a ternary operator which displayed the component if the user was the correct user. Otherwise it displayed a message saying, the section is not available to them.

```
function QuizCard({userid}) {
```

Figure 31. Function showing argument is passed to component to help generate the correct information so I could render it

If the component did get displayed, the ID of the user has to be passed through to the component as seen in figure 31 to help retrieve the relevant information from the database and be illustrate it when required.

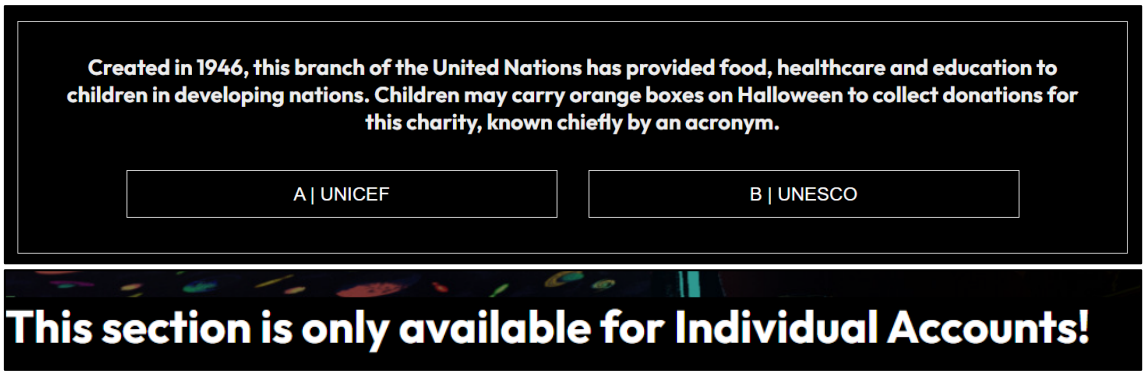


Figure 32. User logged in vs User not logged in

5.12 Gamification Elements

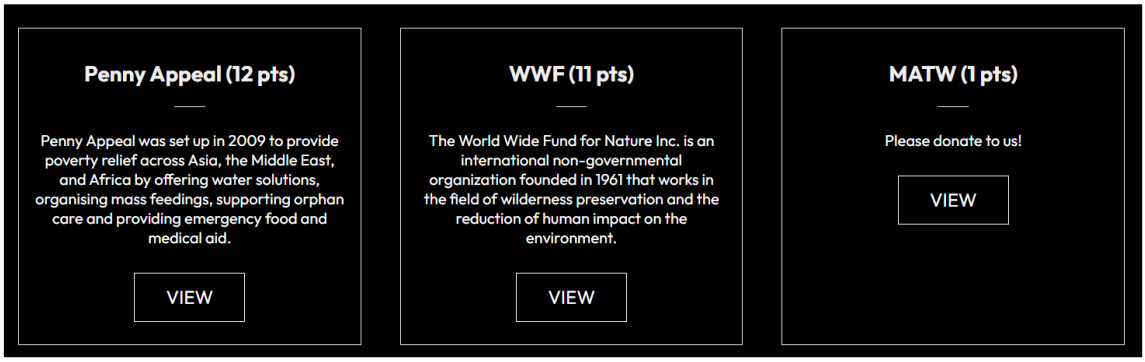


Figure 33. Points are shown next to the charity title

The website I have built has three gamification elements to it. The first one is the multiple-choice quiz on the minigames section as well as the impact it has on the charities section. Based on how many answers which are gotten correct on the quiz, the different charities within the charity section are ranked in terms of points so regardless of how well-known you are, you are ranked based on how many active supporters you have.

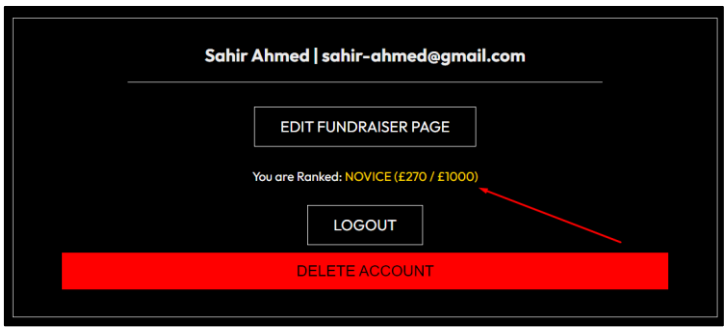
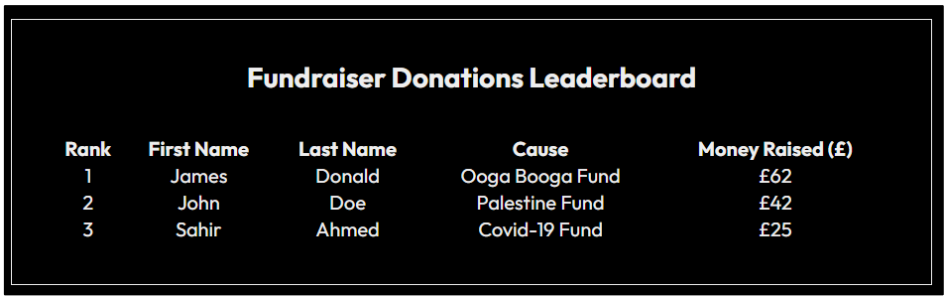


Figure 34. Rankings of user on profile dashboard

The second gamification element which is built into the system is the ranking the system gives based on how much has been donated by an individual or how much has been raised by a charity. Hopefully, this urges people to donate and push through to the next rank.



Fundraiser Donations Leaderboard

Rank	First Name	Last Name	Cause	Money Raised (£)
1	James	Donald	Ooga Booga Fund	£62
2	John	Doe	Palestine Fund	£42
3	Sahir	Ahmed	Covid-19 Fund	£25

Figure 35. Fundraiser donation leaderboard ranked in terms of money raised

The final gamification element which is implemented is the leaderboards which pulls information from the database and encourages people to get the highest rank and maintain that position.

Chapter 6: **Testing**

6.1 Unit Testing

Unit testing is a software testing method which breaks down a source code into lots of small, isolated functions and checks whether or not it meets the needs of its purpose. React is great for this as it is split into multiple components in the frontend and Node can be judged via its API calls in the backend.

Frontend

To test the frontend units, I will be checking if these pages are being rendered with all the required components and correct data.

Routes	Result
About.js	Success
CDLB.js	Success
Charities.js	Success
CharityEditLandingPage.js	Success
CharityLandingPage.js	Success
FDLB.js	Success
FundraiserEditLandingPage.js	Success
FundraiserLandingPage.js	Success
Fundraisers.js	Success
Home.js	Success
IDLB.js	Success
Minigames.js	Success
Profile.js	Success
ProfileCharity.js	Success
Register.js	Success

Statistics.js	Success

Backend

To test these units, I will be making use of console.log() and checking the terminal to see whether it does what it needs to do to fulfil its purpose.

API Call	path	Function	Result
GET	/checkloggedin	Check if user is logged in	Success
Profile Page			
GET	/charityrepresentid	Selection when registering as individual	Success
POST	/individualregister	Registering as individual	Success
POST	/charityregister	Registering as charity	Success
POST	/loginIndividual	Verifying login as individual	Success
POST	/loginCharity	Verifying login as charity	Success
POST	/logout	Logout	Success
Profile Dashboard Pages			
GET	/profile/:id	Information for individual dashboard	Success
DELETE	/profile/:id	Delete individual user	Success
GET	/profilecharity/:id	Information for charity dashboard	Success
Charities Page			

GET	/charitycards	Listing all charities	Success
GET	/charitylandingpage/:id	Getting relevant information for landing page	Success
GET	/charityeditlandingpage/:id	Getting relevant information for landing page placeholders	Success
PATCH	/charityeditlandingpage/:id	Updating landing page	Success
PATCH	/charitylandingpage/:id	Update total_raised for charity	Success
PATCH	/individualdonateeffect	Update total_donated for individual	Success
Fundraiser Page			
GET	/fundraisercards	Listing all fundraisers	Success
GET	/fundraiserlandingpage/:id	Getting relevant information for landing page	Success
GET	/fundraisereditlandingpage/:id	Getting relevant information for landing page placeholders	Success
PATCH	/fundraisereditlandingpage/:id	Updating landing page	Success
PATCH	/fundraiserlandingpage/:id	Update total_raised for fundraiser	Success
PATCH	/individualdonateeffectfund	Update total_donated for individual	Success

Minigame Page			
GET	/quiz	Getting quiz questions and answers	Success
PATCH	/quizcorrect	Updating points by 1 if correct answer	Success
Leaderboard Page			
GET	/individualleaderboard	Getting individual leaderboard	Success
GET	/charityleaderboard	Getting charity leaderboard	Success
GET	/fundraiserleaderboard	Getting fundraiser leaderboard	Success

6.2 Testing against Functional Requirements (Refer to chapter 3 for results)

DISCLAIMER: Trying not to duplicate text

6.3 Testing against Non-Functional Requirements (Refer to chapter 3 for results)

Chapter 7: **Evaluation**

Examples of how my interfaces turned out can be seen in Appendix D.

7.1 Quantitative/Qualitative

7.1.1 What was successful?

When this project was started, the end-product I desired was a platform which could amalgamate all charities that register to it under one website in order to make charitable donations much easier. In addition to this, it was an attempt to exterminate factors which are out of our control (e.g., how much funds a charity has to advertise themselves) and create a level playing field for any charity to use and try and garner support for their cause. Similarly, I tried to create an environment which is easy to setup and does not have 1000's of checks to go live on the website and this in turn, pushes individuals to go and start fundraisers for whatever causes they believe in. I wanted to use gamification elements to help boost donations and although this can only be observed with time, I was able to successfully implement them which is going to create competition and yearning to improve their rankings.

An aim I had prior to developing this application was to be transparent with the use of funds as it can be seen nowadays that not all donations go to the cause you intend it to go to. In order to avoid this, I am happy that I was able to allow users to directly send their money to the charity/fundraiser owner without having me authorise it as it means I do not have any platform fees and therefore maximising the percentage of the contribution.

The frontend was mainly successful for me as I did not find it too difficult to implement what I wanted based on my designs. I believe I was able to match the original agenda of maintaining a minimalistic design as it was not excessively crowded, and it was fairly responsive as well.

The greatest achievement for me upon completing this application is that I was able to learn how to make API calls to the databases all the way from the front-end and receive a response and manipulate the response to extract the data I needed. This is something which was completely new to me as I had very little knowledge on backend development.

I believe this project benefitted me as it gave me insight into full-stack development and tested my technical understanding. It also emphasised the importance of having a well thought out design before advancing into implementation stages.

7.1.2 Challenges

Due to the difficulty of the project and my lack of knowledge, I wasted enormous amounts of time trying to set up the backend just so it can communicate with the frontend. This led me into looking up to various tutorials with no avail as they started to get very complicated. This was an issue as this was the first time I had attempted a full-stack project by myself and put me under a lot of intense pressure. However, fortunately, I managed to get through it eventually.

With the way I developed my application, creating the frontend was fine but after that, I was left with the backend, and this played a huge role for why majority of the components were not functional and caused delays as it was a relatively new concept for me.

Another challenge I faced was when I was working on the authorisation function. I managed to create the authentication function fairly easy, but authorisation proved to be difficult at the very start due to the different number of ways you could get about it. Furthermore, it was extremely difficult to understand tutorials when different frameworks and stacks were used and as someone who does not find it easy to understand code which is new to me, it proved to be strenuous when trying to translate it to my technologies.

7.2 Further Work

Although I met majority of the requirements which were set out at the start, there are a multitude of ways I can improve this web application.

If I were to work on this project in the future, I would first attempt to create a search bar to narrow down charities/fundraisers because as of now, this application is not very scalable in the sense that the user interface will start to get cluttered, and we will have to endlessly scroll through pages to find the charity/fundraiser we desire. It can also help to categorise them based on their causes.

Secondly, I would make the landing pages of both charities and fundraisers more customisable in order for them to stand out more. As seen in my wireframe designs, a donation history as well as stories/updates would be great to add more information to causes. Another thing I would try and implement, is the option to share the link of the landing page across a user's social media to raise awareness and hopefully try and boost donations.

A possible third improvement that I would make is trying to find a way to allow recurring donations. At the moment, with the way the PayPal API is set up, it is not possible to send recurring donations but single transactions instead. This may involve implementing the API in a different manner where the processing of funds occurs on my website but is still hosted by PayPal.

We live in a day where there are internet trolls present so, if I was to work on this in the future, I would try and find a way to prevent troll accounts and verify users without making it too difficult to put up a simple fundraiser page. This reason alone could justify the creation of an administrative role which could delete unwanted content as per his/hers wishes.

Finally, to ensure charities are legitimate, it might be helpful to call in an API which gets a list of legitimate charities and allows the user to register using this rather than them having to enter their registration code manually and may not be real.

7.3 Conclusion

In conclusion, the project set out to create a hub for charities and centralise them all under one website and if charities were to make use of this website, it would be functional and serve their purpose well. I would class this journey as a huge success, and I have picked up a lot of technical knowledge from developing this application. I was able to test my boundaries and cross them to learn something which I never expected to have and understand more about what I would do differently if I were to do something similar again. Although there were times I was at a standstill, by persevering, I was able to complete the project.

Chapter 8: **References**

Aleecia M. McDonald, L. F. C., 2010. Internet User's Understanding of Behavioural Advertising. *Beliefs and Behaviours,* pp. 1-31.

Bennet, R., 2009. Impulsive donation decisions during online browsing of charity websites. *Journal of Consumer Behaviour,* 8(2-3), pp. 116-134.

Children in Crossfire, 2021. *Top 8 Benefits of JustGiving.* [Online] Available at: https://www.childrenincrossfire.org/reasons-to-use-justgiving/ [Accessed 6 November 2021].

Dawson, W., 2020. *React Native Web Vs React- A Comparison Guide.* [Online] Available at: https://codersera.com/blog/react-native-web-vs-react-a-comparison-guide-for-2020/#:~:text=Although%20react%20native%20web%20is%20a%20great%20way%20to%20build%20your%20website%2C%20but%20I%E2%80%99d%20say%20use%20it%20only%20when%20you%20have%20plans%20to% [Accessed 18 November 2021].

Deborah E. Rosen, E. P., 2004. Website Design: Viewing the web as a cognitive landscape. *Journal of Business Research,* 57(7), pp. 787-794.

Free the Ocean, 2021. *About Us.* [Online] Available at: https://www.freetheocean.com/about-us/ [Accessed 8 November 2021].

Free the Ocean, 2021. *How it Works.* [Online] Available at: https://www.freetheocean.com/how-it-works/ [Accessed 8 November 2021].

Free the Ocean, 2021. *Shop.* [Online] Available at: https://shop.freetheocean.com/ [Accessed 8 November 2021].

Gaille, L., 2019. *17 GoFundMe Pros and Cons.* [Online] Available at: https://vittana.org/17-gofundme-pros-and-cons [Accessed 7 November 2021].

Gaurav Kumar, P. K. B., 2012. Impact of Agile Methodology on Software Development Process. *nternational Journal of Computer Technology and Electronics Engineering (IJCTEE),* 2(4), p. 48.

GoFundMe, 2021. *About GoFundMe.* [Online] Available at: https://www.gofundme.com/en-gb/c/about-us [Accessed 7 November 2021].

GoFundMe, 2021. *GoFundMe.* [Online] Available at: https://www.gofundme.com/en-gb [Accessed 25 October 2021].

GoFundMe, 2021. *Raising money for multiple charities at once.* [Online] Available at: https://support.gofundme.com/hc/en-us/articles/360054692351-Raising-money-for-multiple-charities-at-once [Accessed 7 November 2021].

GoFundMe, 2021. *The GoFundMe Guarantee.* [Online] Available at: https://www.gofundme.com/en-gb/c/safety/gofundme-guarantee [Accessed 7 November 2021].

Hamari, J., 2015. Do badges increase user activity? A field experiment on the effects of gamification. *Computers in Human Behaviour,* Volume 71, pp. 469-478.

instinctools, 2021. *Why is Node.Js so popular for rest API?.* [Online] Available at: https://www.instinctools.com/blog/why-is-node-js-so-popular-for-rest-api/ [Accessed 18 November 2021].

JustGiving, 2021. *About Us: JustGiving.* [Online] Available at: https://www.justgiving.com/about [Accessed 5 November 2021].

JustGiving, 2021. *JustGiving.* [Online] Available at: https://www.justgiving.com/ [Accessed 25 October 2021].

JustGiving, 2021. *JustGiving 101.* [Online] Available at: https://www.justgiving.com/about/media-centre/justgiving-101 [Accessed 6 November 2021].

Katerattanakul, P., 2002. Framework of Effective Web Site Design for Business-To-Consumer Internet Commerce. *INFOR: Information Systems and Operational Research,* 40(1), pp. 57-70.

Krig, B., 2019. Implementing gamification with the purpose to enhance user engagement and increase user retention. *Student Thesis,* pp. 14-16.

Lauth, I., 2015. *A BRIEF HISTORY OF CHARITABLE GIVING [INFOGRAPHIC].* [Online] Available at: https://blog.winspireme.com/a-brief-history-of-charitable-giving-infographic [Accessed 24 October 2021].

Matzka, E., 2021. *A guide to overcoming challenges in fundraising.* [Online] Available at: https://www.goodbox.com/2021/10/biggest-challenges-in-fundraising/ [Accessed 26 October 2021].

Monic Sun, F. Z., 2013. Ad Reveneue and Content Commercialization: Evidence from Blogs. *Management Science,* 59(10), pp. 2314-2331.

Rubens, P., 2020. *When should small charities use paid advertising?.* [Online] Available at: https://charitydigital.org.uk/topics/topics/when-should-small-charities-use-paid-advertising-7860 [Accessed 27 October 2021].

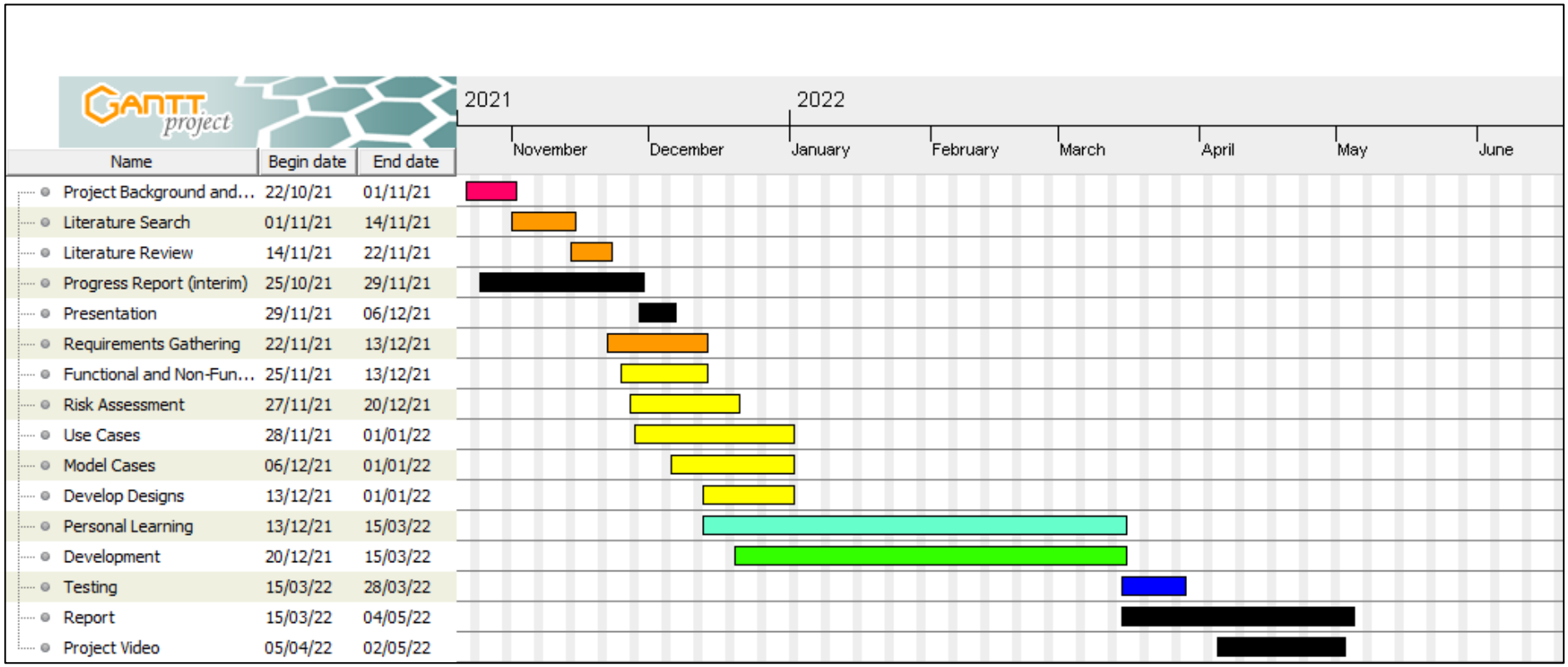
Webb, A., 2020. *The Problem with JustGiving and the best fundraising alternatives.* [Online] Available at: https://becleverwithyourcash.com/the-problem-with-just-giving-and-virgin-giving/ [Accessed 6 November 2021].

Appendix A – Original Project Schedule

Task Name	Start Date	End Date	Duration (Days)	Section
Project Background and Statement	22/10/21	1/11/21	9	1
Literature Search	1/11/21	14/11/21	13	2
Literature Review	14/11/21	22/11/21	8	2
Progress Report (Interim)	25/10/21	29/11/21	35	-
Presentation	29/11/21	6/12/21	7	-
Requirements Gathering	22/11/21	13/12/21	21	2
Functional and Non-Functional Requirements	25/11/21	13/12/21	18	3
Risk Assessment	27/11/21	20/12/21	23	3
Use Cases	28/11/21	1/1/22	34	3
Model Cases	6/12/21	1/1/22	26	3
Develop Designs	13/12/21	1/1/22	19	3
Personal Learning	13/12/21	15/3/22	91	-
Development	20/12/22	15/3/22	85	4
Testing	15/3/22	28/3/22	13	6

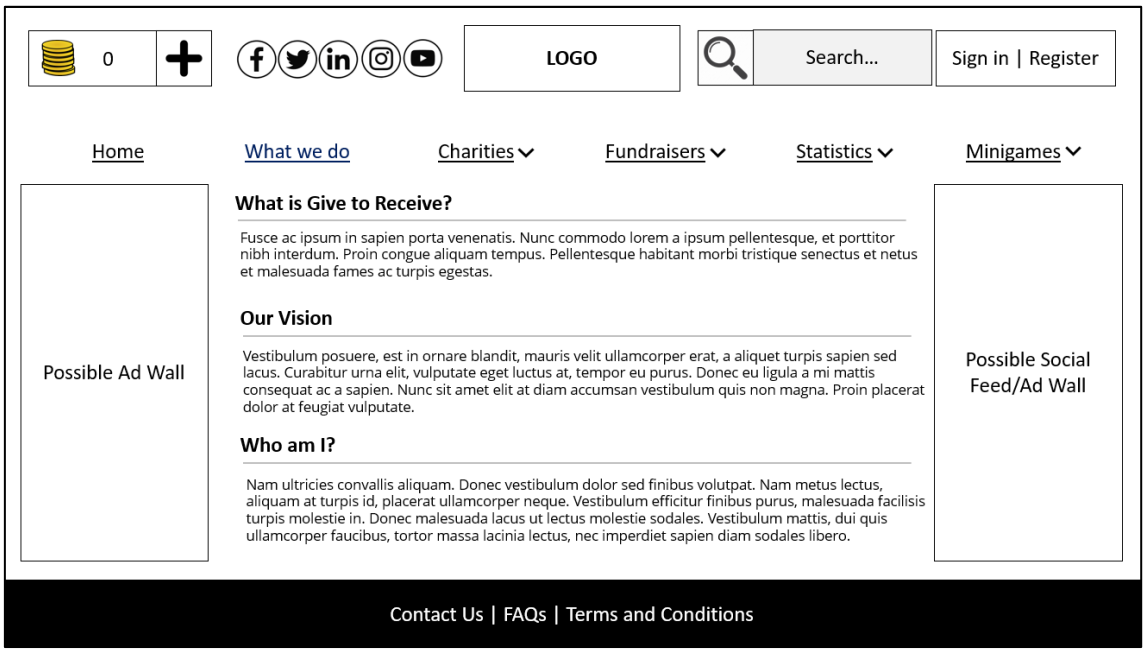
Report	15/3/22	4/5/22	50	-
Project Video	5/4/22	2/5/22	27	-

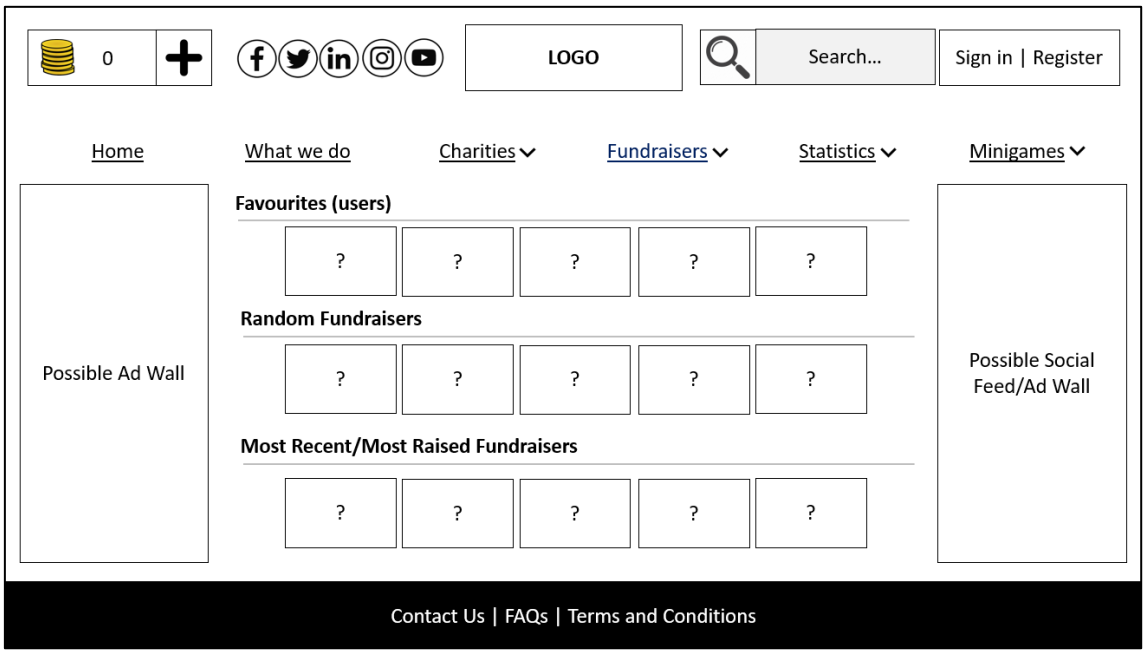
Appendix B – Original Gantt Chart

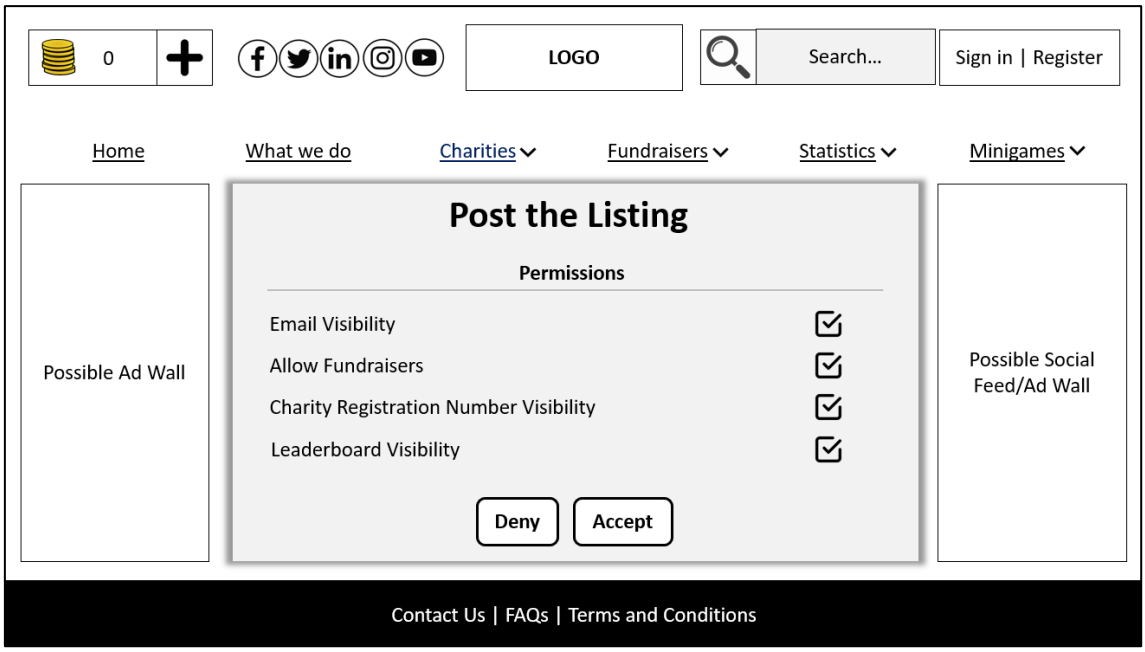


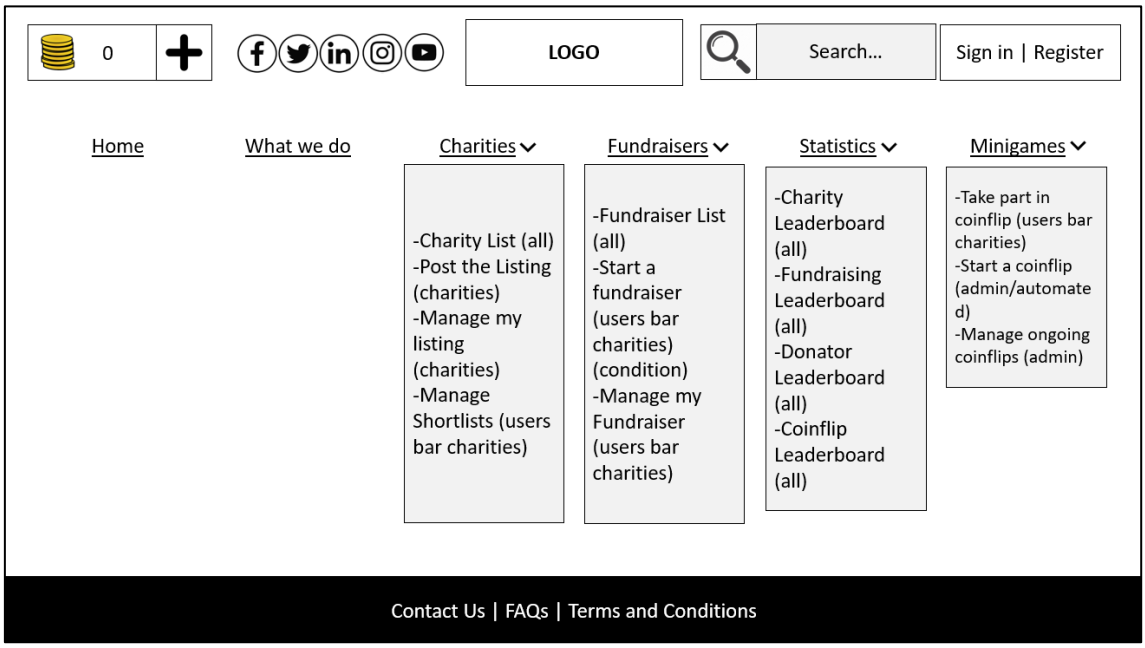
Name	Begin date	End date
Project Background and...	22/10/21	01/11/21
Literature Search	01/11/21	14/11/21
Literature Review	14/11/21	22/11/21
Progress Report (interim)	25/10/21	29/11/21
Presentation	29/11/21	06/12/21
Requirements Gathering	22/11/21	13/12/21
Functional and Non-Fun...	25/11/21	13/12/21
Risk Assessment	27/11/21	20/12/21
Use Cases	28/11/21	01/01/22
Model Cases	06/12/21	01/01/22
Develop Designs	13/12/21	01/01/22
Personal Learning	13/12/21	15/03/22
Development	20/12/21	15/03/22
Testing	15/03/22	28/03/22
Report	15/03/22	04/05/22
Project Video	05/04/22	02/05/22

Appendix C: Wireframe Designs

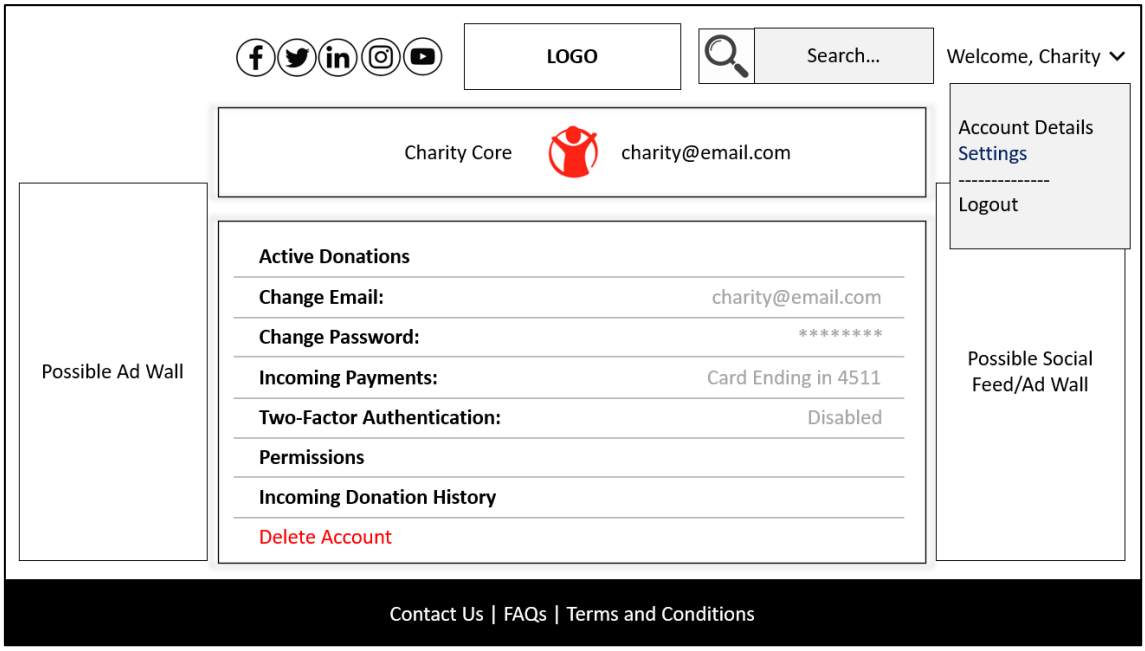




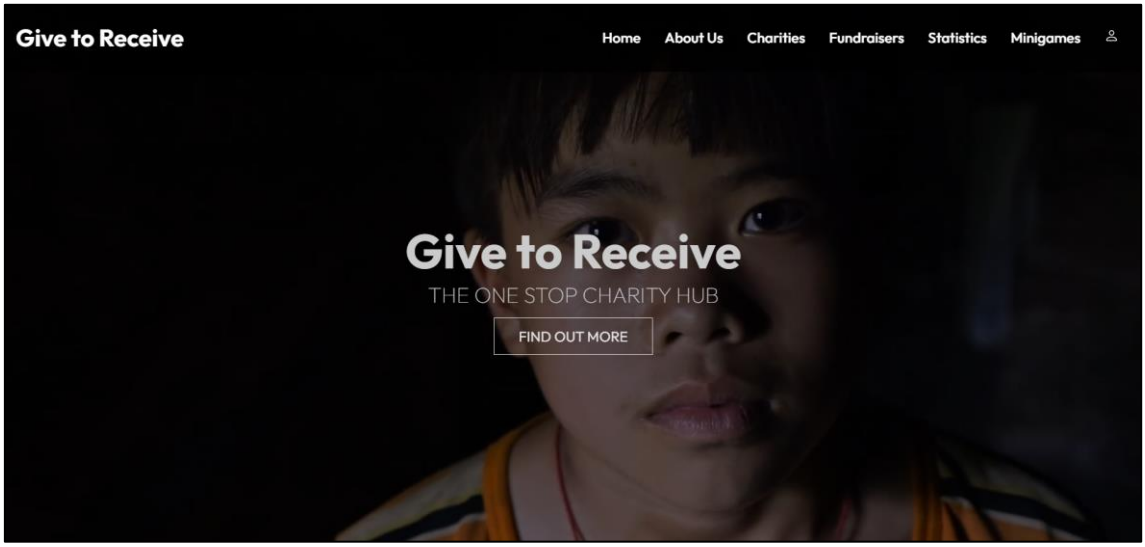

0
LOGO
Search...
Sign in | Register
Home
What we do
Charities
Fundraisers
Statistics
Minigames
Possible Ad Wall
Post the Listing
Permissions
Email Visibility
Allow Fundraisers
Charity Registration Number Visibility
Leaderboard Visibility
Deny
Accept
Possible Social Feed/Ad Wall
Contact Us | FAQs | Terms and Conditions

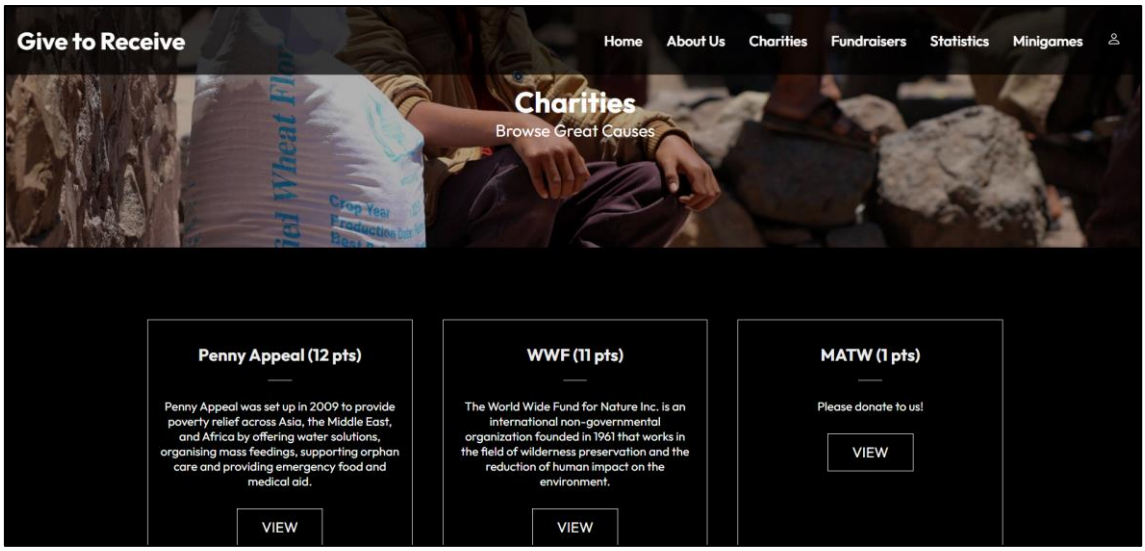

0
LOGO
Search...
Sign in | Register
Home
What we do
Charities
Fundraisers
Statistics
Minigames
-Charity List (all)
-Post the Listing (charities)
-Manage my listing (charities)
-Manage Shortlists (users bar charities)
-Fundraiser List (all)
-Start a fundraiser (users bar charities) (condition)
-Manage my Fundraiser (users bar charities)
-Charity Leaderboard (all)
-Fundraising Leaderboard (all)
-Donator Leaderboard (all)
-Coinflip Leaderboard (all)
-Take part in coinflip (users bar charities)
-Start a coinflip (admin/automated)
-Manage ongoing coinflips (admin)
Contact Us | FAQs | Terms and Conditions

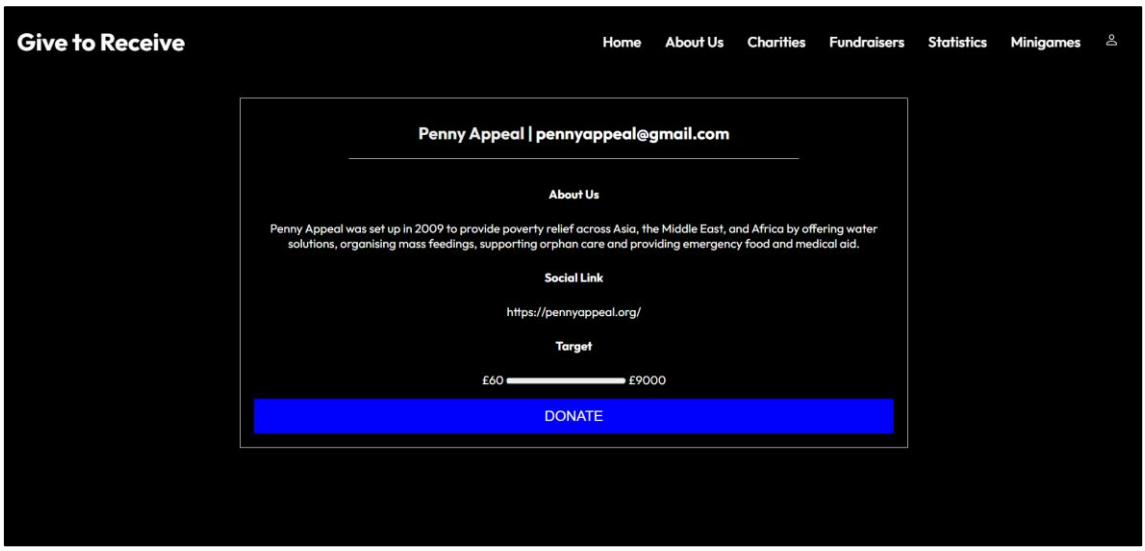

LOGO
Register as a Charity
Charity Name
Country
Charity Registration/Reference Number
Email
Create Password
Confirm Password
Sign Up
Contact Us | FAQs | Terms and Conditions

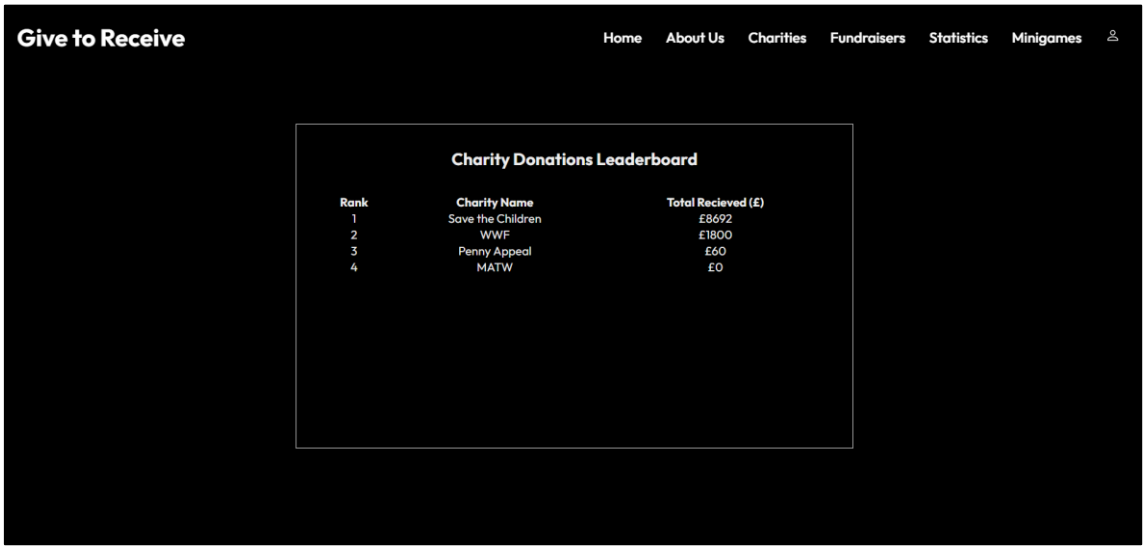

LOGO
Search...
Welcome, Charity
Account Details
Settings
Logout
Charity Core
charity@email.com
Possible Ad Wall
Active Donations
Change Email:
charity@email.com
Change Password:
****
Incoming Payments:
Card Ending in 4511
Two-Factor Authentication:
Disabled
Permissions
Incoming Donation History
Delete Account
Possible Social Feed/Ad Wall
Contact Us | FAQs | Terms and Conditions

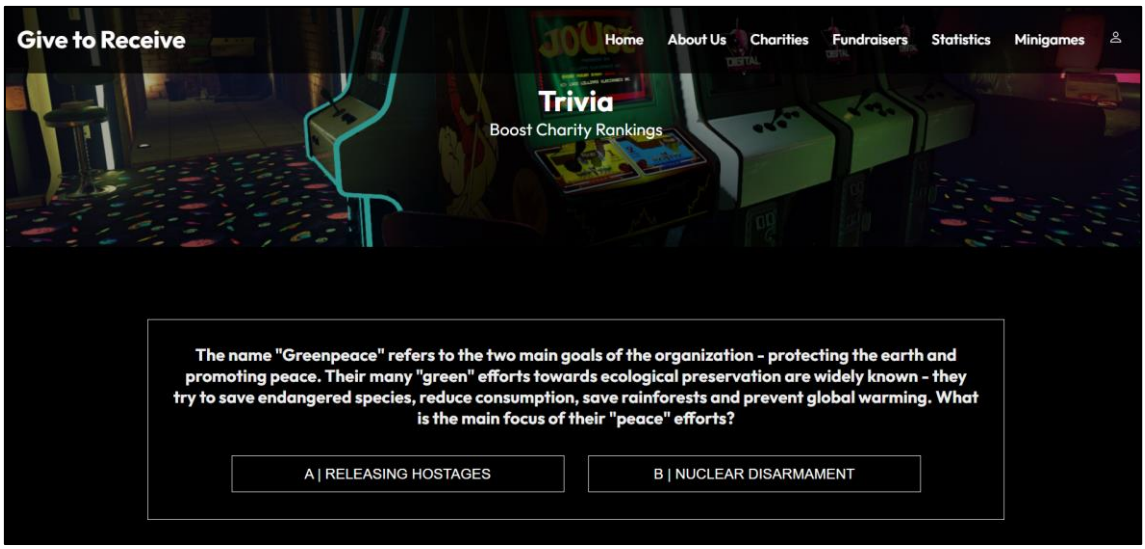
Appendix D: Evaluation

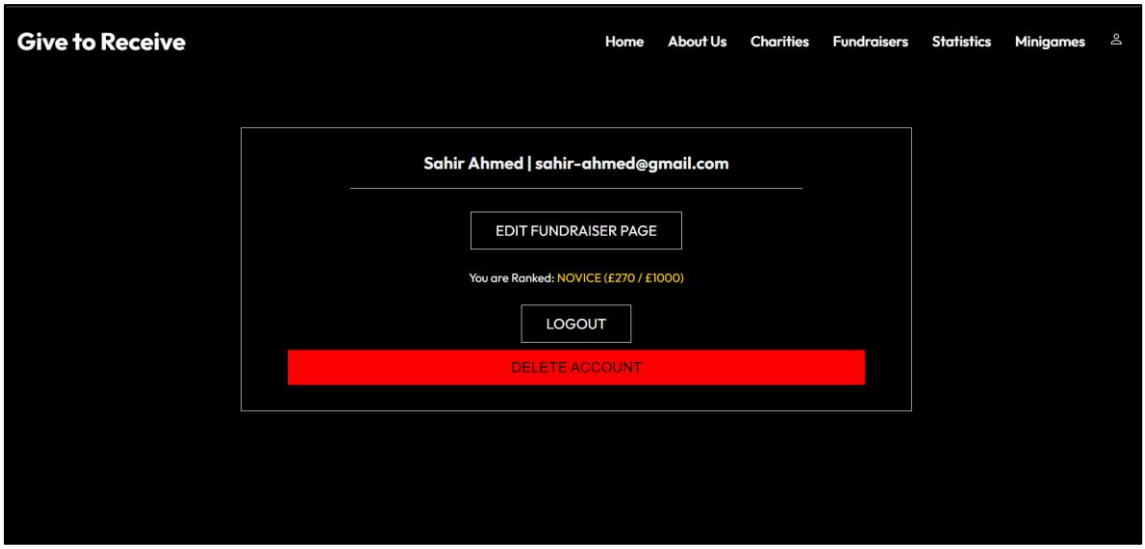







Give to Receive
Home
About Us
Charities
Fundraisers
Statistics
Minigames
Charity Donations Leaderboard
Rank
Charity Name
Total Recieved (£)
1
Save the Children
£8692
2
WWF
£1800
3
Penny Appeal
£60
4
MATW
£0


Give to Receive
Home
About Us
Charities
Fundraisers
Statistics
Minigames
Trivia
Boost Charity Rankings
The name "Greenpeace" refers to the two main goals of the organization - protecting the earth and promoting peace. Their many "green" efforts towards ecological preservation are widely known - they try to save endangered species, reduce consumption, save rainforests and prevent global warming. What is the main focus of their "peace" efforts?
A | RELEASING HOSTAGES
B | NUCLEAR DISARMAMENT


Give to Receive
Home
About Us
Charities
Fundraisers
Statistics
Minigames
Sahir Ahmed | sahir-ahmed@gmail.com
EDIT FUNDRAISER PAGE
You are Ranked: NOVICE (£270 / £1000)
LOGOUT
DELETE ACCOUNT

Appendix E: Actual Project Timeline

Task Name	Start Date	End Date	Duration (Days)	Section
Project Background and Statement	22/10/21	1/11/21	9	1
Literature Search	1/11/21	14/11/21	13	2
Literature Review	14/11/21	22/11/21	8	2
Progress Report (Interim)	25/10/21	29/11/21	35	-
Presentation	29/11/21	6/12/21	7	-
Requirements Gathering	22/11/21	13/12/21	21	2
Functional and Non-Functional Requirements	25/11/21	13/12/21	18	3
Risk Assessment	27/11/21	20/12/21	23	3
Use Cases	28/11/21	**5/2/22**		3
Model Cases	6/12/21	**14/2/22**		3
Develop Designs	13/12/21	**17/2/22**		3
Personal Learning	13/12/21	**25/4/22**		-
Development	**28/12/22**	**25/4/22**		4
Testing	**1/4/22**	**25/4/22**		6
Report	**1/4/22**	**3/5/22**		-

Project Video	**17/4/22**	**4/5/22**		-
Implementation Resubmission	**14/7/22**	**25/7/22**	11	
Report Resubmission	**25/7/22**	**29/7/22**	4	
Project Video Resubmission	**27/7/22**	**29/7/22**	2	