COMPUTERBANK: A COMMUNITY-BASED COMPUTER DONATION PLATFORM USING MACHINE LEARNING AND NFT

Eric Gong[1] and Yu Sun[2]

[1]University High School, 4771 Campus Drive, Irvine, CA 92612
[2]California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

Many people donate money to fund organizations, but very rarely do those donors have information about where those donations go. Donation platforms are both non-transparent and also leave a large portion of potential donors unnoticed: gamers [1]. This paper explores the concept of utilizing blockchain technology and its existence as a web3 token-based platform in order to provide transparency for donation routes, showing donors and other companies exactly where donations are coming from and where that money is going. Our application utilizes HTTP requests in order to greatly increase compatibility, and also uses multiple private key encryptions in order to ensure that any user data or information and monetary transactions are kept secure and private [2].

KEYWORDS

web3, donation, platform, decentralized.

1. INTRODUCTION

In recent years, the popularity of cryptocurrency has soared through the roof due to new accessibility and promises of striking it rich [3]. According to a study done by Insider Intelligence, around 34 million US adults own cryptocurrency [4]. Nowadays, it would be rare to find a person who has yet to hear of cryptocurrency. Moreover, in 2020, after the advent of the Corona virus pandemic as well as the racial justice movements of the time, Americans gave a record $471 billion in donations. Video games are played by hundreds of millions around the world, from the gamers who'll dedicate hours at a time to their hobby, to people who might just play on their phone every once in a while to pass a train journey. Video games are big business and have the potential to offer a lot of support to good causes [5]. According to a study done by the Charities Aid Foundation, around 58% of players were interested in donating while playing; 59% would be more likely to pay to remove adverts if some of the cost went to charity; and 63% would use funds from online wallets to donate [6]. Despite such promising numbers, the industry of video games has surprisingly not overlapped too much with the idea of donations. Instead of utilizing the traditional methods of campaigning our platform uses online wallets and the large plethora of gamers willing to donate in order to make money-raising much quicker and more transparent.

Existing donation platforms exist, sites such as GoFundMe and DonorBox allow for patrons from all over the world to donate money to their favorite charities. They allow for organizations to create a fundraising campaign, with a description and the option for anyone to donate. The main

David C. Wyld et al. (Eds): SIP, CRBL, WiMNeT, BIOM, EDUPT, NLDM, SOFTFM, COMIT - 2022
pp. 29-42, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121704

way that they gain traction is through affiliate links and the organization's own efforts to spread word regarding their fundraising. However, these websites require donors to go out of their way to find the website and then proceed to donate, which limits the scale of their reach. Another issue with this form of donation is that the proceeds are extremely hard to track. Many organizations can create a fundraising event and use the donations towards something else, with little to no transparency on where the money is actually going.

Our tool is a web3 token-based gaming donation platform. Because it is a dApp, it runs on a blockchain of computers and is free from the influence of a single authority [7]. In comparison to others. In comparison to other donation platforms who operate under one site owner and spread through word of mouth, Donahub's API is integrated into distributed video games and takes donations from a wide variety of players. A strong feature of using an online wallet for the organization to receive funds is that it offers transparency in where the money is going. By tracking the organization's wallet ID and its transactions, it guarantees that the organization is actually receiving the money and also spending it on the right things. Another feature that we have is the ability for an admin to determine how the donations will be spread among various organizations, allowing a single donation to benefit more than one organization.

In practice, Donahub demonstrates how the web3 token-based structured donation platform increases the effectiveness of donations. First, we have complete transparency, showing users the recent transactions list including addresses for easier tracking. Donahub also utilizes HTTP requests in order to make new donations, making it highly compatible with most games and coding languages [8]. Donations can be made from multiple sources as long as HTTP requests are being sent, letting you make donations from web browsers, games, and any other source. The app is also secure, with all the data being stored in a Moralis database using private keys to both encrypt and decrypt the information. Donahub has its own distribution algorithms, permitting users to donate equally among organizations or give priority to specific organizations.

The rest of the paper will follow this outline: Section 2 describes challenges we encountered during the process of creating and carrying out this experiment, Section 3 explains the methodology we used and the solution we came up with, Section 4 evaluates the experiment and provides extra details, Section 5 presents related works, and Section 6 gives concluding remarks.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Web3-Based Portal

The development of a web3-based portal [9]. As the idea of Web3, a decentralized version of the world wide web, is still extremely new, there is little to no help regarding the development of a platform based on it. Moreover, it is necessary to connect every little bit of data collected from the platform to a database as well as connect a wallet to its user. Because of the novelty of dApps, it is a challenge to figure out the connection process and make sure that data being collected is being stored in the right places and is also able to be called whenever needed. It is also difficult to develop both the backend and frontend of such a platform, figuring out how to handle the money as well as create a pleasing user interface that provides all the information necessary.

2.2. Integration Compatibility

The compatibility of the platform to integrate with different game applications. Due to the use of our platform depending on a game's code and integrating our SDK/API, it is a challenge to ensure that our platform is compatible with the plethora of games that exist. Because different developers use different tokens, different engines, different code, our platform might not be compatible with all kinds of games. Moreover, developers might want to integrate our SDK/API in their own way.

2.3. Fund Distribution Algorithm

The diversity of the fund distribution algorithm. Our platform is going to work by distributing donations given to us to a number of different charities using a distribution algorithm. However as a platform, we most likely will be wanting to use a wide variety of algorithms, encompassing individual developers' priorities. The algorithm also needs to be flexible, able to adapt to the needs of different donors.

3. SOLUTION

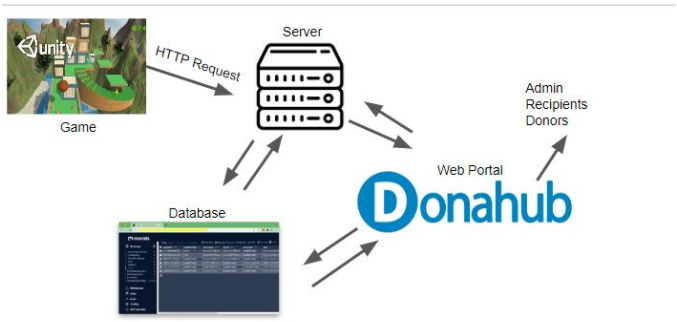


Figure 1. Overview of the solution

Donahub is a web3 decentralized blockchain-based donation platform that integrates into games in order to gain donations from players [10]. Users have two options on how to donate; either from a game or directly from our web platform. Our platform has 4 main components, the games, our server, a database, and our web portal. Players of integrated games will have the option to send an HTTP request containing their account name and donation amount to our server, which communicates with our database and our web platform. People donating from the web platform don't need to send their account name because they will have to be signed in to donate, instead only needing to send their donation amount to our server and database. Receiving an HTTP request from anywhere will create and save a new donation object in our database, which will then be displayed on our web portal for a final confirmation. The web portal houses important information such as past and pending transactions and a list of all recipients. The website is also where donors will be able to choose how their funds get distributed amongst the organizations, with an equal distribution function and a priority distribution one.

Moralis is a platform that provides the full-stack workflow for people looking to develop their own dApps. It provides you with a database that will automatically store all of the data that you have configured and is easily accessible from the frontend. All of the features that Moralis offers are cross-chain, so the data can flow easily between different blockchains. We utilized Moralis's functions in order to create our signup and signin functions as well as easily keep track of transactions and display them.

```
async function connectWithWallet() {
                let user = Moralis.User.current();
                if (!user) {
                  user = await Moralis.authenticate();
                }
                console.log("logged in user:", user);
                console.log("logged in user:", user.attributes.ethAddress);
                document.getElementById("input_walletid").value =
user.attributes.ethAddress;
                const DonaUser = Moralis.Object.extend("DonaUser");
                const query = new Moralis.Query(DonaUser);

                query.get(currentUser.objectId)
                    .then((donaUser) => {
                      // The object was retrieved successfully.
                      donaUser.set("ethAddress", user.attributes.ethAddress);
                      donaUser.set("ethUserObjectId", user.id);

                      donaUser.save().then((donaUser) => {
                          console.log("Update the object successfully.");
                          console.log(donaUser);
                      });
                    }, (error) => {
                      alert("Failed to update the user.");
                    });
                }
```

Figure 2. Screenshot of code 1

Portal

- Design
- For each screen/page
- User Homepage

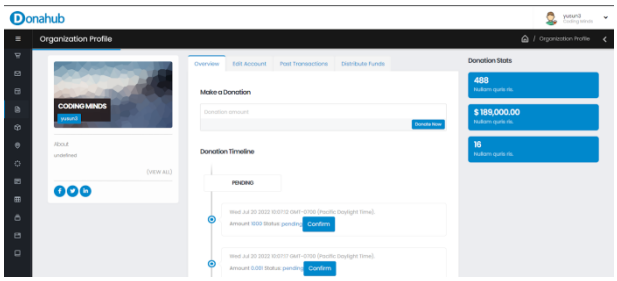


Figure 3. Screenshot of making donation

Our user profile page is where everyone is taken after they sign into their account. On the left hand side, we can see our organization logo along with its name, the user's username, and a short description of the organization. At the top middle section, we have 4 tabs: Overview, Edit Account, Past Transactions, and Distribute Funds. In Overview, we can either make a fast donation using the Make a Donation input box or confirm any pending donations that we have. Everything that we see on the left hand side can be edited using the edit account tab.

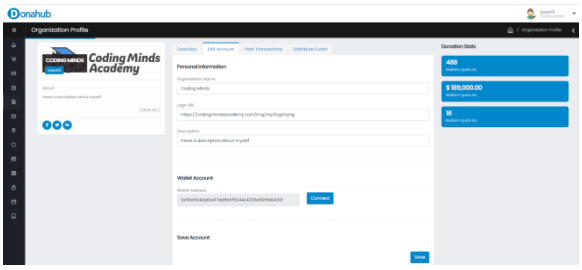


Figure 4. Screenshot of filling information

The Past Transactions tab is where you can see a timeline of every single past donation/transaction that has been made.

```
async function getUserTransactions() {
  let user = Moralis.User.current();
    if (!user) {
      user = await Moralis.authenticate();
    }
  // create query
  const query = new Moralis.Query("EthTransactions");
  console.log(user.get("ethAddress"));
  query.equalTo("to_address", user.get("ethAddress"));

  // run query
  const results = await query.find();
  console.log("user transactions:", results);
  for (var i = 0; i < results.length; i++) {
    var liItemHTML = '<li><div class="tm-box"><p class="text-muted mb-0">'
+ results[i].createdAt + '.</p>'
                + '<p>From <span class="text-primary">' +
results[i].attributes.from_address+ '</span> to <span class="text-primary">' +
results[i].attributes.to_address + '</span></p></div></li>';
      document.getElementById("transaction_list").innerHTML += liItemHTML;
    }
  }
```

Figure 5. Screenshot of code 2

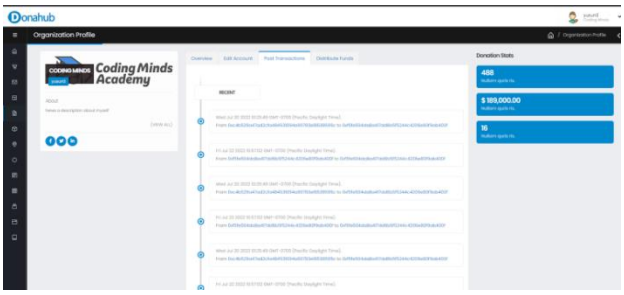


Figure 6. Screenshot of post transaction

The Distribute Funds tab is where you can decide on how you are going to donate. An equal distribution button allows for you to enter an amount. That amount will then be donated to every organization that has signed up with DonaHub.

```
async function equalDonation() {
  var amount = document.getElementById("equal_distribution_amount").value;
  const DonaUser = Moralis.Object.extend("DonaUser");
  const query = new Moralis.Query(DonaUser);
  const results = await query.find();
  console.log("Successfully retrieved " + results.length + " users.");
  if (results.length > 0) {
    for(var i = 0; i < results.length; i++) {
      donationAmountForWallet(results[i].attributes.ethAddress, amount);
      console.log("donated " + amount + " to " +
results[i].attributes.ethAddress);
    }
  }
}
```

Figure 7. Screenshot of code 3

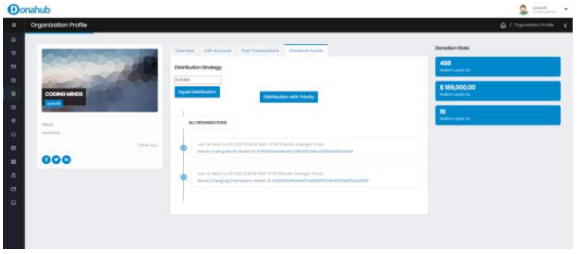


Figure 8. Screenshot of Funds

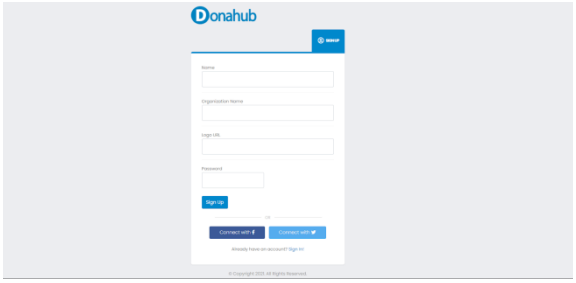


Figure 9. Sign up page

The signup page consists of 4 fields. Submitting your information here and clicking sign up will save all 4 fields into a single object (Donauser) stored in our database. There is also a button at the bottom that will allow you to sign in if you already have an account.

```
function signup() {
    // 1. collect all the values from the input boxes
    console.log("sign up clicked");
    var username = document.getElementById("input_username").value;
    var password = document.getElementById("input_password").value;
    var organization = document.getElementById("input_orgname").value;
    var logourl = document.getElementById("input_logourl").value;

    // 2. send it to server to save the user info
    const DonaUser = Moralis.Object.extend("DonaUser");
    const donaUser = new DonaUser();

    donaUser.set("username", username);
    donaUser.set("password", password);
    donaUser.set("organization_name", organization);
    donaUser.set("logourl", logourl);

    donaUser.save().then((donaUser) => {
        console.log("Save the object successfully.");
        console.log(donaUser);
        // 3. show a message "Signup request submitted"
        window.location.href = "pages-signin.html";
    });
}
```

Figure 10. Screenshot of code 4

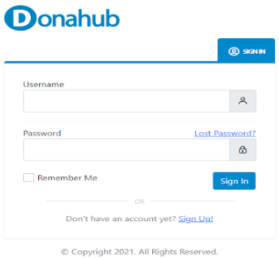


Figure 11. Sign in page

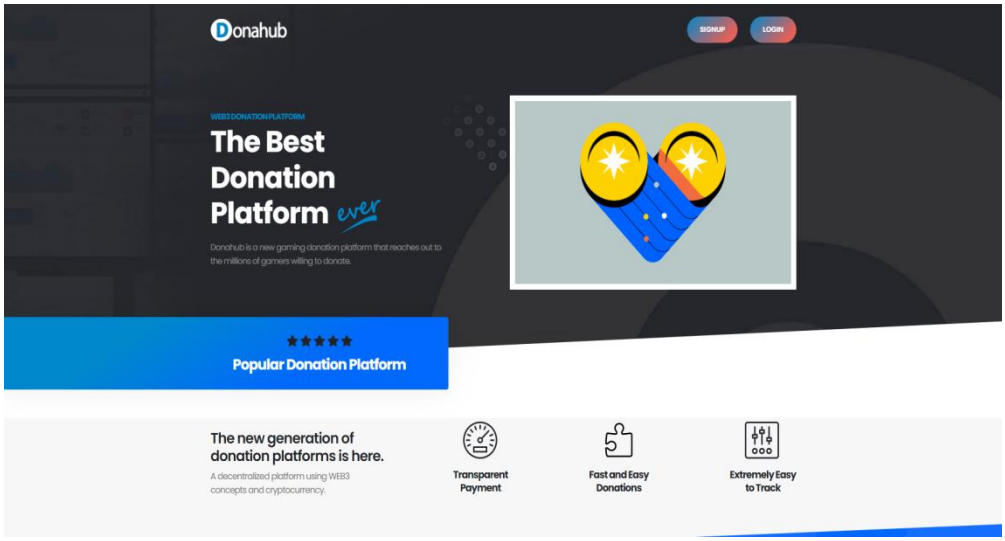
The sign-in page is where users will input their username and password in order to access their organizations profile page. The input forms are collected and compared to the data in the database in order to check for a match.

```
async function login() {
        console.log("signup clicked");
        var username = document.getElementById("input_username").value;
        var password = document.getElementById("input_password").value;

        console.log(username);
        console.log(password);

        const DonaUser = Moralis.Object.extend("DonaUser");
        const query = new Moralis.Query(DonaUser);
        query.equalTo("username", username);
        const results = await query.find();
        console.log("Successfully retrieved " + results.length + " users.");
        // Do something with the returned Moralis.Object values
        if (results.length > 0) {
          const object = results[0];
          console.log(object.id + ' - ' + object.get('username'));
          if (object.get("password") == password) {
            console.log(JSON.stringify(object));
            localStorage.setItem("currentUser", JSON.stringify(object));
            alert("sign in successful");
            window.location.href = "pages-user-profile.html";
          } else {
              alert("Incorrect or invalid password");
            }
        } else {
            alert("Invalid Username");
          }
    }
```

Figure 12. Screenshot of code 5

Homepage

- The homepage contains much needed information about our donation platform as well as provides two buttons in the top right; sign-in and sign-up. It is written using Javascript and HTML and CSS.



Server

- This server allows us to externally call upon our donation functions, enabling people to make donations from outside of our portal. This is how we will be integrating our donation platform with other games, by having them send an HTTP request to our server.
- The server uses Node.js, a back-end JavaScript runtime environment that works to execute JavaScript code externally; outside a web browser. This allows us to use JavaScript for server-side scripting and also allows us to use Express.js, which is a back-end web app framework for Node.js. Express.js is what actually allows for us to send external HTTP requests and is what we use to create all of our APIs.

- We are able to send requests through a website url, a command-line shell, and through unity. In all of these methods, the account name and amount trying to be donated is required in order to successfully make the donation.
- Code for Server:

```
app.get('/makedonation', (req, res) => {
  const Donation = Moralis.Object.extend("Donation");
                                const donation = new Donation();

                                donation.set("account", req.query.account);
                                donation.set("amount", parseInt(req.query.amount, 10));
                                donation.set("date", (date.getMonth()+1) + "/" + date.getDate() +
"/" + date.getFullYear());
                                donation.set("status", "pending");

                                donation.save().then((donation) => {
                                        console.log("Save the object successfully.");
                                        console.log(donation);
                                });
    res.send('Donation received')
})

app.get('/showall', (req, res) => {
  const donation = Moralis.Object.extend("donation");
        const query = new Moralis.Query(donation);
        const results = query.find();
        console.log("Successfully retrieved " + results.length + " users.");
    res.send('Here are all the donations')
})
```

Figure 13. Screenshot of code 6

Database

- We created a database using Moralis.
- When signing up, each user is stored in the database as an object called a donaUser. The donaUser object stores the organization's name, the user's username, logo url, and password.
- Whenever someone creates a donation, it creates a new donation object within the database.
- Game Integration with the Server API
- Within Unity, we can send an HTTP request by utilizing a coroutine that uses Unity's web request functions.
- public class NewBehaviourScript : MonoBehaviour

```
{
    // Start is called before the first frame update
    void Start()
    {
        Debug.Log("Started");
        StartCoroutine(MakeDonation());
        Debug.Log("Ended");
    }

    IEnumerator MakeDonation() {
            Debug.Log("Sending ...");
            UnityWebRequest www =
UnityWebRequest.Get("https://EricProject-1.sunyu912.repl.co/makedonation?account=yusun3&amount
=0.5");
        yield return www.SendWebRequest();
    }
```

Figure 14. Screenshot of code 7

Inside of the Unity game, we can utilize the provided TextMeshPro packages to create input fields and a button. All we need to do is have the button send an HTTP request using the code above except replacing the account name and donation amount with the values inside of the text input fields. We can do this easily by simply creating two TMP input field variables and then adding their text value to the UnityWebRequest.

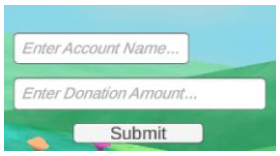


Figure 15. Put donation information page

```
public class SendDonation : MonoBehaviour
{

    public TMP_InputField account_inputField;
    public TMP_InputField donation_inputField;

    public void sendDonation()
    {
        Debug.Log("Started");
        StartCoroutine(MakeDonation());
        Debug.Log("Ended");
    }

    IEnumerator MakeDonation()
    {
        Debug.Log("Sending ...");
        UnityWebRequest www =
UnityWebRequest.Get("https://EricProject-1.sunyu912.repl.co/makedonation?account=" +
account_inputField.text + "&amount=" + donation_inputField.text);
        yield return www.SendWebRequest();

        if (www.result != UnityWebRequest.Result.Success)
        {
            Debug.Log(www.error);
        }
        else
        {
            Debug.Log("Request completed!");
        }
    }

}
```

Figure 16. Screenshot of code 8

4. EXPERIMENT

4.1. Experiment 1

On account of DonaHub being built on a blockchain network, it automatically becomes more secure than regular donations. This is because, while it is still susceptible to cyber threats, all transactions are encrypted and recorded on a distributed ledger, any hackers would need to change the entire chain in order to alter the records.

HTTPS is the protocol that we use to send information between the website and the web browser. HTTPS is a secure protocol due to its use of an asymmetric public key infrastructure. It contains a public and a private key. The public key is used by everyone who wants to access the website, encrypting information sent. The private key is used to decrypt information encrypted on the public key. The data encrypted on the public key can only be decrypted by the private key, making the website more secure.

HTTPS has been used in all of the communications in our Web3 app:

- Calling our nodejs server
- Moralis communication
- Moralis database

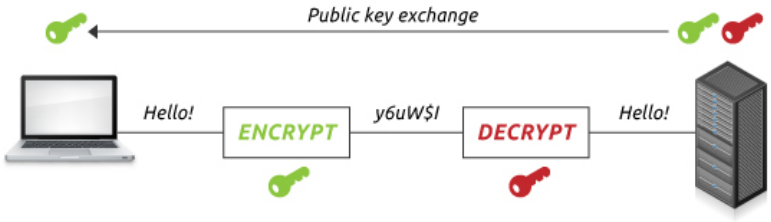


Figure 17. HTTPS security

Data encryption is important simply due to the extra security it provides. Especially because our platform handles matters regarding money, if unencrypted data was gained by a third party, they would be able to easily read the data and use it. All of our data is stored in a Moralis database which is securely encrypted using private keys. Private keys are used to both encrypt and decrypt the data in this case.

All of the data is stored in the Moralis database, which is securely encrypted using private keys.

4.2. Experiment 2

Our donation platform is shown to greatly increase transparency in transactions and in tracking what donations are used for. Due to the usage of blockchain technology as well as our web portal's display functionalities, all transactions of the digital wallet are displayed in the user homepage, allowing for easier tracking and much more transparency compared to other donation platforms, which will accept donations and never track where the money actually goes.

Donations can be made whenever someone sends an HTTPS request to our server. This creates a new donation object inside of our database. When the object is created, the donation is automatically marked as pending. Within the admin account, users will be able to click the confirm button, which will finalize the transaction and actually send the donation. Once the donation has been made, it will be publicly available to see through different ETH explorers such as: https://etherscan.io/.

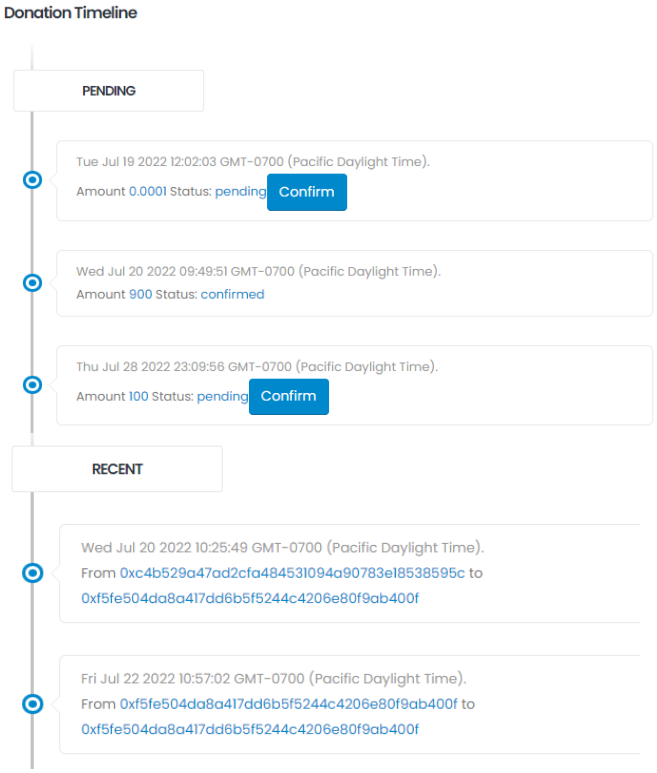


Figure 18. Donation timeline

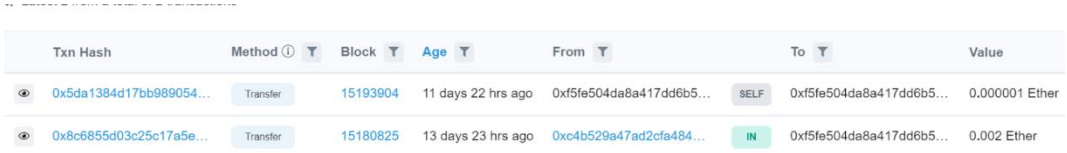


Figure 19. Files (https://etherscan.io/address/0xf5fe504da8a417dd6b5f5244c4206e80f9ab400f)

It is also possible to have the donations distributed among the different organizations that have signed up with DonaHub. Inside of the admin account user homepage, there are two options: Equal distribution or priority distribution. Equal distribution takes the amount you want to donate, splits it evenly among the number of organizations, and then sends the donations. Priority distribution allows for you to choose which organizations will receive the funds, either allowing some to get either most or all of the donated money.

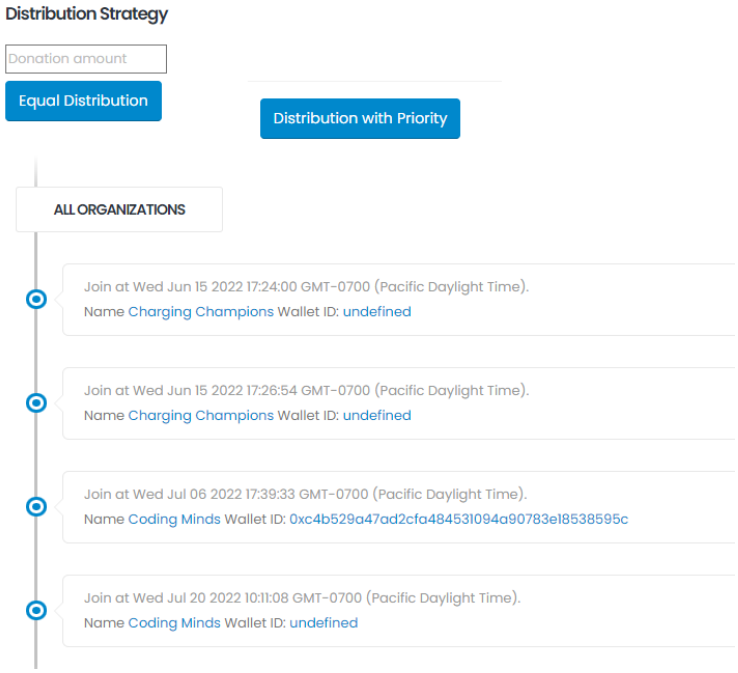


Figure 20. Distribution strategy

Our platform is extremely compatible with most programs, regardless of different game engines, coding language, etc. The only necessary component that an application needs to connect to our services is the ability to send an HTTP request. Because nearly all coding languages have incorporated HTTP requests, it becomes especially simple to send a donation request to our server.

Game Type	Library
Unity	UnityWebRequest https://docs.unity3d.com/Manual/UnityWebRequest.html
ReactJS	AJAX https://reactjs.org/docs/faq-ajax.html
GameMaker	HTTP Request Function https://manual.yoyogames.com/GameMaker_Language/GML_Reference/Asynchronous_Functions/HTTP/http_request.htm
Other Game Engine …	
Other Game Engine …	
Web - HTML/Javascript	
iOS - Swift	
Android - Java	
Flutter - Dart	Dart HTTP Library https://pub.dev/packages/http

Figure 21. Table of game type and library

5. RELATED WORK

This work focuses on the decreasing trust and transparency within the field of donations as well as offers some solutions to aforementioned issues [11]. The work offers two use cases, with the first one solving transparency issues utilizing blockchains and declaring an organization's status. The second use case focuses more on a general approach towards donations and presents a rough draft for a donation model with a DonationToken that has a special ID and features. This work was much more broad in terms of execution and was not as focused on actually creating a functioning web3 donation platform.

This work discusses the distrust regarding the usage of donated money [12]. This work mainly focuses on blockchains and their ability to create transparency and discusses the implementation of a platform that would track donations based on blockchain technology. The work states that this would offer transparent accounting and provide a transparent donation route. This work is much more broad in terms of its research, also mainly focusing on discussing the implementation as compared to actually executing.

This work analyzes the feasibility of a blockchain based financial product that would solve the many challenges as well as optimize the process of donation [13]. The authors created a dApp and tested it using Ethereum and fully evaluated the throughput and feasibility of such a blockchain based product. The work is similar in its creation of a financial product, however it focuses more on testing the feasibility rather than creating a practical product.

6. CONCLUSIONS

In summary, DonaHub is a Web3 token based donation platform that seeks to provide transparency and security in the field of fundraising [14]. Utilizing blockchain technology, DonaHub allows for easy and traceable transactions that tell donors and organizations exactly where or to where the money is going. DonaHub also provides its own distribution system, allowing users to either equally distribute funds among organizations, or give priority to specific ones. The application is also easily integrated into most games, tapping into the wide variety of gamers willing to donate money. The only factor necessary in integrating DonaHub is the ability to send an HTTPS request to our server.

The application used requires game developers to send HTTPS requests to our server in order for donations to work. There are limitations to this because, while rare, some engines do not support easily sending such requests. Moreover, there is also an issue of simply getting organizations to sign up for our portal. Another limitation is that crypto currency and digital wallets are still relatively new and not as many people have them.

The first issue can be solved by simply allowing our server to take more different kinds of requests other than just HTTPS ones, such as JSON-RPC. Other limitations can be fixed by just expanding our reach and through reach out to companies. Lastly, the number of people owning a digital wallet will gradually increase over time as cryptocurrency grows in popularity [15].

REFERENCES

[1] Almeida, Fernando, and Alexandre Cunha. "Digital Donation Platform for Nonprofit and Charity Organizations." International Journal of Information Communication Technologies and Human Development (IJICTHD) 10.3 (2018): 14-27.

[2] Rao, Thammavarapu R. N., and K-H. Nam. "Private-key algebraic-code encryptions." IEEE Transactions on Information Theory 35.4 (1989): 829-833.

[3] Mukhopadhyay, Ujan, et al. "A brief survey of cryptocurrency systems." 2016 14th annual conference on privacy, security and trust (PST). IEEE, 2016.
[4] Liu, Yukun, and Aleh Tsyvinski. "Risks and returns of cryptocurrency." The Review of Financial Studies 34.6 (2021): 2689-2727.
[5] Funk, Jeanne B. "Reevaluating the impact of video games." Clinical pediatrics 32.2 (1993): 86-90.
[6] Piper, Holly. "Case study: Charities Aid Foundation and St Mungo's." Demystifying Social Finance and Social Investment. Routledge, 2020. 140-145.
[7] Kovacova, Maria, Jakub Horak, and Michael Higgins. "Behavioral Analytics, Immersive Technologies, and Machine Vision Algorithms in the Web3-powered Metaverse World." Linguistic and Philosophical Investigations 21 (2022): 57-72.
[8] Nielsen, Henrik Frystyk, et al. "Network performance effects of HTTP/1.1, CSS1, and PNG." Proceedings of the ACM SIGCOMM'97 conference on Applications, technologies, architectures, and protocols for computer communication. 1997.
[9] Wang, Qin, et al. "Exploring Web3 From the View of Blockchain." arXiv preprint arXiv:2206.08821 (2022).
[10] Zheng, Zibin, et al. "Blockchain challenges and opportunities: A survey." International journal of web and grid services 14.4 (2018): 352-375.
[11] Enria, Luisa, et al. "Trust and transparency in times of crisis: Results from an online survey during the first wave (April 2020) of the COVID-19 epidemic in the UK." PloS one 16.2 (2021): e0239247.
[12] Callen, Jeffrey L. "Money donations, volunteering and organizational efficiency." Journal of Productivity Analysis 5.3 (1994): 215-228.
[13] Zhang, Zhonghua, et al. "Recent advances in blockchain and artificial intelligence integration: feasibility analysis, research issues, applications, challenges, and future work." Security and Communication Networks 2021 (2021).
[14] Vesterlund, Lise. "The informational value of sequential fundraising." Journal of public Economics 87.3-4 (2003): 627-657.
[15] Rathore, Hem Shweta. "Adoption of digital wallet by consumers." BVIMSR's journal of management research 8.1 (2016): 69.