

**Department of Design Engineering and Mathematics**

# **Raising Capital on the Blockchain with EthRaiser: An Ethereum-Based Crowdfunding Platform**

**Mohammed Jaffar**

B.Eng. Computer Systems Engineering 2020-2023

**April 2023**

## ABSTRACT

EthRaiser is an approach to the problems that come with traditional systems of finance, and crowdfunding; such as lack of security for stakeholders, regulatory bottlenecks, costs, time efficiency, global accessibility, and lack of privacy. EthRaiser is a project on decentralized web3 crowdfunding which involves the blockchain, smart contracts, and front-end development with React along with several more components. The output of a project is a platform that, holds the security of stakeholders by automating functions and alleviating of any human error, voluntary or involuntary. It provides a way to apply legal and regulatory policies by coding the smart contract. Every transaction made on EthRaiser is openly visible on the blockchain, thus bringing transparency. Increased global accessibility as all that's needed to donate to a website is funds, a Metamask wallet, and internet. Increased privacy as no personal information is stored of the donating users. Reducing costs and time efficiency by automating transactions and removing the need for purchasing or maintenance of hardware such as servers, and data centers. EthRaiser is a leap towards the blockchain era, with research provided to take well-informed steps. The EthRaiser technical report will provide a background to describe its core concepts, a well-informed and equilateral review of related literature, its design approach and implementation. Apart from the research conducted by EthRaiser, further research is encouraged in smart contract security, zero-knowledge proofs, ring signatures, homomorphic encryption, and consensus mechanisms.

## CONTENTS

<b>Acknowledgement .....</b>	<b>Error! Bookmark not defined.</b>
<b>Abstract.....</b>	<b>2</b>
<b>Table of Figures and Illustrations.....</b>	<b>5</b>
<b>Introduction.....</b>	<b>8</b>
Background .....	9
Blockchain .....	9
Consensus Mechanisms .....	9
Ethereum .....	12
Smart Contracts .....	12
Ethereum Virtual Machine .....	13
Wallets.....	13
Web2 vs Web3.....	13
Crowdfunding.....	14
Motivation & Research Problem .....	14
<b>Literature Review .....</b>	<b>16</b>
Introduction .....	16
Overview.....	16
Scope of literature review .....	17
Blockchain & Ethereum.....	17
Characteristics of Blockchain Technology .....	17
Advantages and Limitations of Etheruem .....	18
Ethereum's Limitations For DeFi.....	18
Smart Contract Development .....	18
Benefits and Challenges.....	18
Smart Contract Language, Tool Limitations and Solutions.....	19
Smart Contracts in Financial Automation and Regulation .....	20
Finance & Investment Applications On Ethereum .....	20
Potential and Challenges of Ethereum-based DeFi Ecosystems.....	20
Risks of Investing in Ethereum-Based Financial Products .....	20
Assesment of Ethereum Transforming Traditional Finance.....	21
Crowdfunding On Ethereum.....	21
Ethereum's Potential in Crowdfunding .....	21
Challenges in crowdfunding.....	22
Crowdfunding Models With Ethereum .....	22
Crowdfunding On Ethereum.....	23
Ethical Issues .....	24
<b>Design Development and Implementation .....</b>	<b>24</b>

Pre-Design .....	24
Research Methodology:.....	27
Development Methodology .....	27
Resource & Material Allocation.....	28
Design.....	31
Design Elements .....	31
Contract Design .....	33
Contract Deployment.....	35
Project Brand and UI/UX design .....	36
Front-End Development .....	40
Incentivization Proposal.....	41
<b>TESTING .....</b>	<b>42</b>
Contract Testing .....	42
Expected Results .....	42
Test Setup .....	42
Procedure .....	42
Results.....	44
EthRaiser Final Testing.....	45
Expected Results .....	45
Test Setup .....	45
Procedure .....	45
Results.....	48
<b>Discussion .....</b>	<b>48</b>
Related works.....	48
Kickstarter.....	48
StartJOIN.....	49
Limitations of EthRaiser .....	49
Future Improvements & Developments .....	50
<b>Conclusion .....</b>	<b>51</b>
Reflection .....	51
Findings and Contributions .....	51
Difficulties Faced.....	51
Author's Conclusion .....	51
<b>Appendix .....</b>	<b>52</b>
<b>References .....</b>	<b>54</b>

**TABLE OF FIGURES AND ILLUSTRATIONS**

<b>No.</b>	<b>Description</b>	<b>Page</b>
1.1	EthRaiser Logo	9
1.2	A depiction of the structure of a Blockchain	10
1.3	The flowchart of a PoW Mechanism	11
1.4	The flowchart of a DPoS Mechanism	12
3.1	Work Breakdown Structure	26
3.2	Gantt Chart	28
3.3	Waterfall method for backend development	29
3.4	Agile method for frontend development	29
3.5	Flowchart for EthRaiser	34
3.6	EthRaiser Logo	37
3.7	UI Wire Frame	38

EDITOR’S NOTE

Greetings, I have written this note to explain the format of this technical report. The structure is given below:

1
1.1
1.1.1
1.1.1.1

Visuals such as figures, illustrations, and snippets will be given in light-grey shaded boxes. Codes will be given inside a simple white rectangular box with a black outline. Notes will be given in light Ethereum Blue shaded boxes. Please do refer to the Appendices when mentioned, they contain definitions and further explanations.

Thank You,  
Mohammed Jaffar



## INTRODUCTION



Illustration 1.1 (EthRaiser Logo)

EthRaiser is a decentralized web3 crowdfunding platform. This project involves the blockchain, smart contracts, and front-end development with React. The main goal of this project is to alleviate businesses, startups, and charities from the deal-breaking drawbacks of traditional methods of crowdfunding and the traditional finance system. This is done using a decentralized system known as the Ethereum Blockchain. Traditional crowdfunding and finance systems have drawbacks such as a lack of security of funds, personal information, fraud, high transaction fees/durations, centralization, hacks, etc. EthRaiser provides a method to alleviate these problems. The project's development was done from February 14 to March 14 and includes features such as campaign creation, campaign donation, and much more on the way. The development of the project has involved various elements, such as:

### Back End:

- Goerli Testnet
- Smart Contracts
- Solidity
- ThirdWeb
- HardHat
- Remix IDE
- Metamask

### Front-End:

- React.js
- JavaScript
- Node
- Vite
- TailWindCSS

This introduction consists of the background information and concepts related to the project, rationale, problem definition, aims and objectives.



## BACKGROUND

The background is written to describe the concepts involved in the project in detail.

## BLOCKCHAIN

A Blockchain is an internetworked, distributed, and decentralized system where the same official ledger (of transactions) is on various miner nodes on the network. A ledger is known as a record that holds all the transactions and movements of money or assets of an entity. The blockchain can also be described as a chain of blocks that contain the ledger data, a cryptographic hash (Appendix 1.1) of the previous block and a cryptographic hash of the current block (Figure 1.1). Due to the hashing mechanism used, each block's current hash links all the way to the first block of the blockchain. Hence, making the blockchain immutable or tamper-proof as if any block is tampered with it will deem all the blocks after that particular block invalid. Each transaction on the blockchain is also publicly viewable. Blockchain explorers such as Etherscan, and Blockchain.com are widely used to view and verify if a transaction was successful, etc. This makes transactions traceable. As the blockchain is decentralized there is no space for human error or fraud, increased efficiency, and decreased costs. The decentralization and automated aspect of the blockchain will be described later in the report. Distributed ledgers can cause problems of double spending (Appendix 1.2) for which consensus mechanisms are used. Consensus mechanisms and peer-to-peer validation are also used to add security and tamper-proofing.

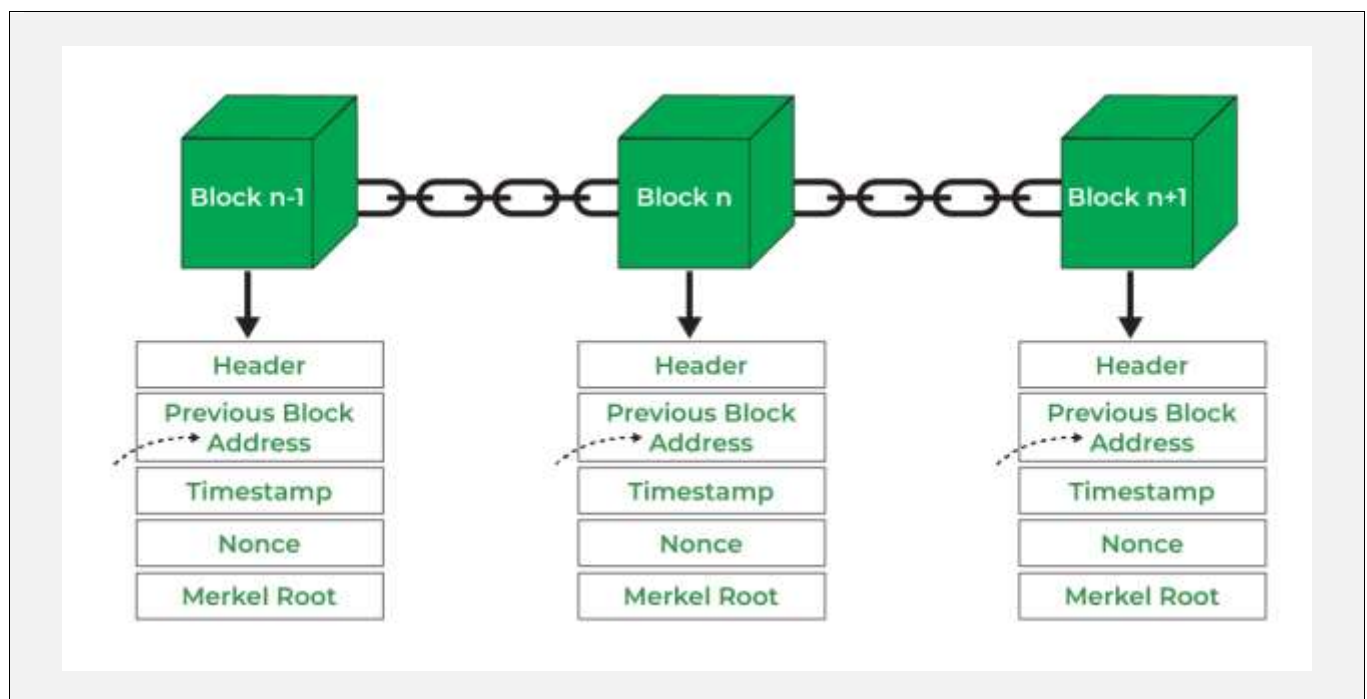


Figure 1.2 (A depiction of the structure of a Blockchain)

## CONSENSUS MECHANISMS

As described earlier, consensus mechanisms add security to the system, this is done by validating transactions based on a certain process. The process is for the miner nodes to come to an agreement on the state of the blockchain, and validate transactions based on that. The two most used consensus mechanisms are Proof-of-Work and Proof-Of-Stake. The distributed ledger of the blockchain means that if a hacker node had tried to change the data on their ledger, they would have to change every other node's ledger, and if they don't their ledger deems invalid discarding the transactions.. A property of consensus mechanisms is that the longer chain (longer chain of valid blocks) always wins and will be the chain the network will follow.

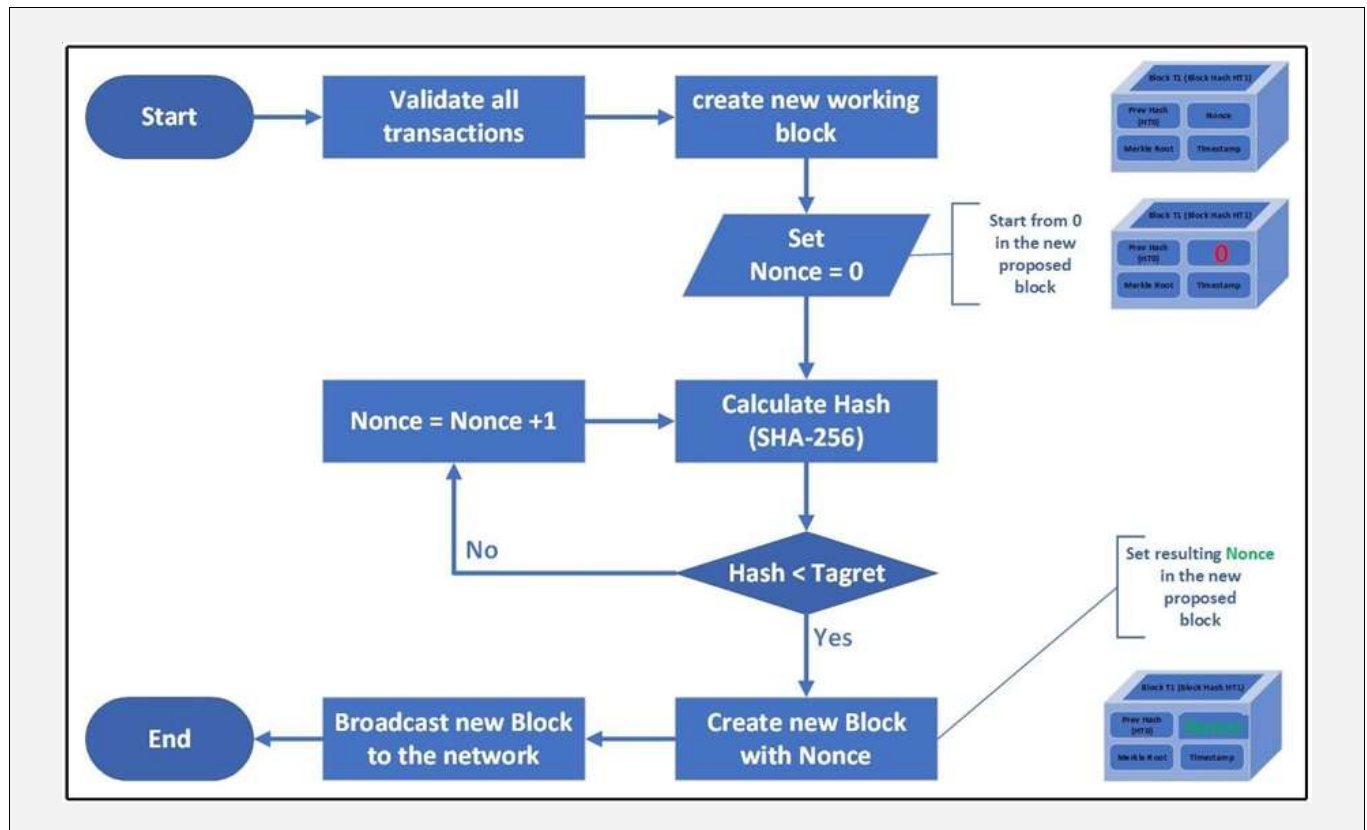


Figure 1.3 (The flowchart of a PoW Mechanism)

The Proof-of-Work mechanism involves each miner node solving a complex computational task to find a hash of an input that meets a criteria or condition generated by the blockchain network. The process of finding the hash is very complex and demanding, however each time a candidate hash is generated, it is easily possible to check if it is the right hash for the given input. Several miners compete against each other to solve the computational task, the winner gets to add a block to the blockchain after being validated by the rest of the miners. Miners also gain an incentive for their validations. Each computational task has its own difficulty level. In Bitcoin the difficulty level is varied in attempts to maintain a block creation every 10 minutes. When more miners join the network, or the computational power of the miner nodes gets higher the difficulty level increases to maintain the 10-minute interval. Ethereum's proof-of-work mechanism differs from Bitcoin. A hashing algorithm, Ethash is used to solve the computational task. This algorithm differs from Bitcoin in the way that it is more memory demanding than processor demanding. So, in Bitcoin having better CPUs and GPUs would help whereas in Ethereum more memory space helps. In Ethereum difficulty is adjusted as well. However, in Ethereum the difficulty level keeps increasing which makes it harder and more expensive to mine blocks. This is done as Ethereum is moving towards a Proof-of-Stake mechanism in the future so Proof-of-Work will be phased out. Each Bitcoin miner has a fixed incentive for mining which means with each block mined, Bitcoin is newly minted (Appendix 1.7). On the Ethereum Blockchain the incentive is given based upon the gas limit (Appendix 1.3) and fee of the block. The Proof-of-Work mechanism is largely criticized for its resource-intensive method of validation. A lot of energy is expended in solving the computational tasks adding to environmental damage. Another issue with the Proof-of-Work mechanism, although seems arbitrary, is the 51% attack. If any miner or mining system on the blockchain could have more than 51% of the entire computational power of the miner node network, it becomes dominant and has the power to hack the system.

Proof-Of-Stake is created to alleviate the problems we face from Proof-of-Work. The PoS mechanism is a lot more energy conserving and provides more security. Overall, Proof-of-Stake involves validator nodes staking their cryptocurrency (currency of the blockchain) to have the chance at validating, verifying transactions and adding to the blockchain. The gas fees of the transactions made in the block are the incentives for the validator node. Here, the main motivation to make right decisions is that the staked

amount will be lost if the validator node makes false validations. There are two popular and most used Proof-of-Stake Algorithms; Byzantine Fault Tolerance (BFT) and Delegated Proof-of-Stake (DpoS). Select stakers (validator nodes) are chosen to make decisions and add to the blockchain after validating transactions. The validators are usually chosen on the criteria of who staked the most. In order to make this process favourable not only to the richest validator, selection methods such as Randomized Block Selection and Coin Age Selection are used. In Randomized Block Selection in a group of node validators the one with the most staked amount but the least hash value (Appendix 1.4) is selected. Coin Age Selection gives priority to stakers who have staked the most for the longest. A simple formula to describe this is given below:

$$\text{Time Staked} \times \text{Amount Staked}$$

However, once a validator node has finished adding a block, there is a cooldown period after which they can start validating again. Once a validator decides to leave the staking pool, their stakes are held for a period before being released. This period is to wait and see if all the transactions validated by the node are legitimate. The arbitrary method of attacking a PoS mechanism is the 51% attack. Here, the fraudulent node would have to stake more than 51% of the Ether that is totally staked in the network. As of January 12<sup>th</sup>, 2023, the staked amount is about 22.38 billion USD worth of Ether.

A BFT consensus mechanism consists of a group of nodes exchanging the data about their ledgers with each other to check which one is legitimate. Each node will come up with an output for itself and will send it to all other nodes in the round. The other nodes will come up with their outputs. If any node comes up with the wrong output, the node will be voted out of the consensus process.

DpoS involves stakeholders electing delegates which would become the validator (delegate) nodes. Each stakeholder can vote a number of times that is proportional to the amount of cryptocurrency they have staked. This creates a sort of digital democracy. There are 21-101 delegates elected by the stakeholders of the network. The delegates are responsible for validating the transactions in the network. In a block each delegate has a set amount of transactions that they can validate, and the process is like clockwork. Each delegate has their own timeframe to add transactions or blocks. Failing to make the correct validations will result in impeachment of the delegate and loss of their stake.

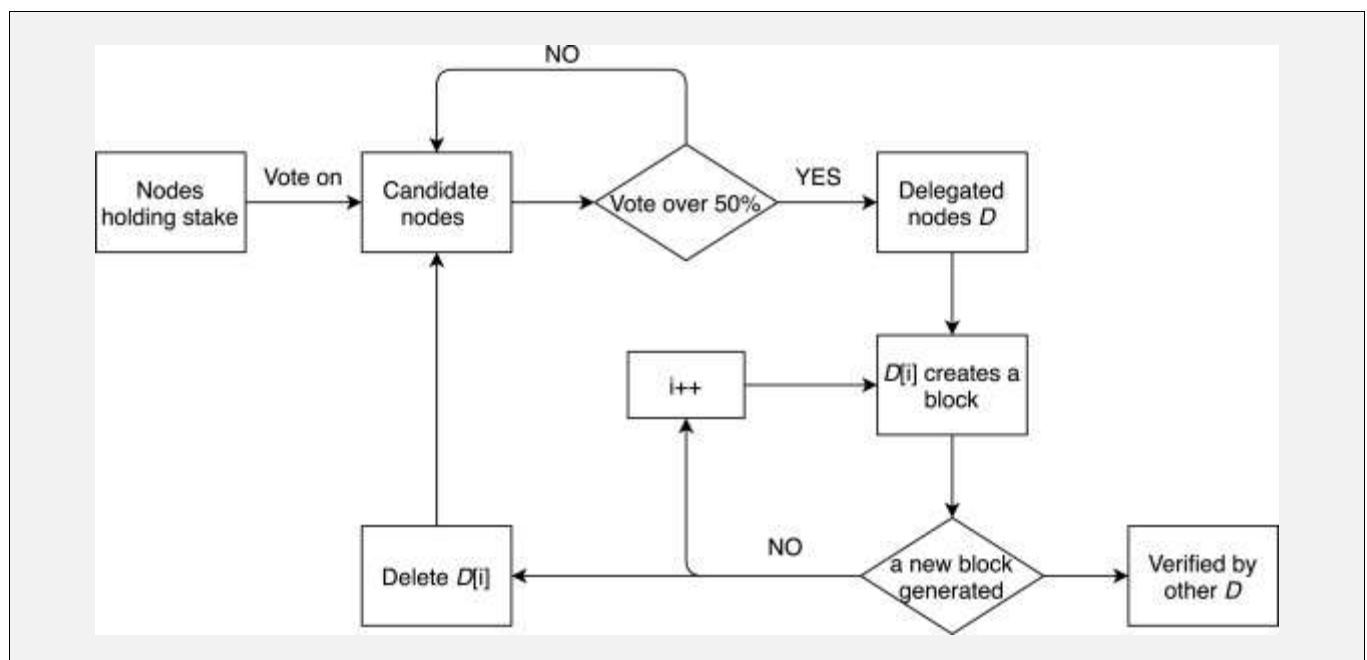


Figure 1.4 (The flowchart of a DpoS Mechanism [D for Delegate])

---

## ETHEREUM

The Ethereum blockchain (Appendix 1.5) was made with the goal in mind being creating a platform that is suitable for the development of Decentralized Applications (dApps) with the use of smart contracts. Smart contracts are automatically executing contracts which are on the blockchain, hence, they have all the advantages and properties the blockchain provides.

Ethereum consists of its native cryptocurrency Ether. Ether is also offered as an incentive to miner/validator nodes. As mentioned before Ether is also used for transaction fees on the Ethereum blockchain. Another use for Ether will be described in the Smart Contracts section. Ether is widely used in Decentralized Finance Applications which will be discussed later in the report. Ether is liquidated against fiat (Appendix 1.13) currency, so it has a real monetary value. Ether is highly liquid and widely traded around many exchange platforms. It is also sought as an investment opportunity as the value can increase or decrease based on supply and demand (Appendix 1.6). Refer to Appendix 1.7 for more on Ether. Ether is highly volatile, the value rapidly mainly due to events, developments, and media along with Supply & Demand. Network congestion, market sentiments, regulatory changes are other big reasons for the volatility. Ether is also subject to fluctuations due to whale movement (Appendix 1.8). These facts will be important and will be talked about more in the limitations of the project and in the Literature Review section.

Ethereum began with the PoW mechanism, however, it is migrating towards PoS. Proof-of-Authority consensus mechanism is also being explored by Ethereum.

There are various test networks like Rinkeby, Goerli, and Sepolia that run their own test blockchains. The properties of the test networks are the same as Ethereum, hence, they are used to deploy and test smart contracts without charging developers with fees. The test networks have their own valueless cryptocurrencies which mimic Ether. The test cryptocurrencies are available over the internet on various faucets for free. The Rinkeby test net, however, has been deprecated as of 12<sup>th</sup> April 2023. The Goerli test net will soon be deprecated by the end of the year 2023.

---

## SMART CONTRACTS

The main aspect of Ethereum that makes it such a great blockchain is the use of Smart Contracts. Smart Contracts are automatically executing contracts that are set with logic and parameters that will execute given the conditions. Smart Contracts are immutable, hack proof, automated, transparent, and secure. Smart Contracts are used as the backbone for building applications. Smart Contracts are deemed transparent as per everything on the blockchain. Various blockchain explorers such as Etherscan can provide the look inside the contracts of a lot of today's decentralized applications. Hence, it adds an open-source aspect to all of the decentralized applications as we can see how they are made. As Smart Contracts are decentralized, they don't need any input or intermediaries in the execution, this reduces the risk of fraud or human inefficiency along with costs and time. The disadvantages of using Smart Contracts will be mentioned in the Literature Review section.

Examples of using Smart Contracts in dApps are, insurance policies, investment schemes, loaning, real estate contracts, and staking.

Smart Contracts on Ethereum are programmed in Solidity. Solidity is a statically (Appendix 1.9) typed compiled programming language similar to C++, JavaScript, Python. The main paradigm in Solidity programming is OOP (Appendix 1.10) which increases programmability and complexity of the contract. OOP also enables modular and reusable code. Solidity comes with pre-built libraries and functions that aid in the development of more complex applications on the blockchain.

---

## ETHEREUM VIRTUAL MACHINE

The Ethereum Virtual Machine (EVM) is run on top of a miner node's machine. The miner can install an Ethereum (Ethereum Node) hypervisor (Type 2) such as Geth or Parity which will manage the EVMs. The EVM's architecture is specifically made for executing smart contracts. It can be thought of as a virtual CPU machine that can execute opcodes. Smart Contracts written in high-level languages like Solidity or Vyper get compiled down into bytecodes that map to Opcodes that are understandable by the EVM. All EVMs on every node in the world work on the same block at a time. The analogy of states comes into discussion here, where machines execute instructions by using the current state (world state) of Ethereum to calculate the next state of Ethereum. Ethereum can be described as a chain of states. One of the main goals of the EVM is to execute functions and make calculations in a consistent manner and method every time. A state can be defined basically as the current state of all the accounts on Ethereum. Accounts are of two types, one is the account that is a wallet (Externally Owned Account), and the next type is a contract account. A wallet account only contains the nonce (Appendix 1.15), and the balance of the account. Whereas the contract account contains; the nonce (Appendix 1.15), contract balance, contract storage hash (of the state of all the elements or variables in the contract), code hash (hash of the Opcodes in the contract). The externally owned account is controlled by the owner with their private-key, meanwhile the contract is controlled by the EVM Opcodes once it has been uploaded. An EVM is created in every miner node's machine to execute EVM code for a contract. Within this EVM machine there are elements such as Program Counter, Memory, Stack, Virtual ROM (stores all EVM code), Gas Counter (counts the amount of gas used), and a Storage (storing Storage hash from the contract). The stack is a data structure that will be used to store and compute values. Note that each opcode requires gas to be executed. The gas amounts vary for different opcodes. One mini-function of the EVM is also to cancel and discard the transaction if not enough gas has been provided by the contract or user that is trying to make the contract call. If a contract includes a sub-contract, another child EVM is created to run the contents of the sub-contract. To summarize, the EVM is a machine architecture that is specifically built for execution of smart contracts to maintain the security, decentralization, and efficiency of the Ethereum blockchain.

---

## WALLETS

(Software) Wallets on the blockchain have their own addresses. For example: '0x35eE0007feCb41bcEC36B9258e1Efc29A9392961'. Wallets can be software wallets and hardware wallets. Software wallets are stored on computers, cloud servers, mobile phones, etc. The software wallets are interfaced usually with software or applications. The Private-Keys are stored in cloud-servers mostly. This adds a security bottleneck to Software Wallets. Hardware Wallets on the other hand store private keys physically and are a lot safer. Hardware Wallets are interfaced through its own hardware. Each wallet has a Private Key and a Public Key (Appendix 1.11). The Private-Key is the digital signature of the sending wallet. On the blockchain the transaction data itself is encrypted, while the sender's address, receiver's address and amount transacted is visible to the public as it is used in consensus mechanisms and validation. There are various wallet providers such as Metamask (software), Coinbase (software), Ledger (hardware), ZenGo (software).

---

## WEB2 VS WEB3

The worldwide web and its web applications are generationally classified with the terms web 1.0, web 2.0, and web 3.0. There is no set and stone definition for each web, but this report assumes a majority standpoint on this and with relevance to the project concepts. Web 2.0 (2004-now) is often criticized for its targeted advertising and lack of privacy. Web applications such as Facebook, Google, etc. take our data and often use it for making their services better or selling them to external entities (data brokers, etc.) which earns them money. This pirates the privacy of the internet user. However, web users tend to still give up their data to receive better user experience on the application. The web 2.0 uses AI-ML extensively



to understand users and build profiles of each user. Web 2.0 uses fiat currencies like USD, GBP, or INR. Web 2.0 is centralized.

Web 3.0 introduces decentralization of applications throughout the web. Digital Profiles are used instead of real profiles which would increase privacy and decrease the risks associated with its invasion. Fiat currencies are replaced by cryptocurrencies. Decentralization removes the chance of censorship to content. The applications of web3 will be autonomous and run on mechanisms such as distributed consensus, Edge computing systems (Appendix 1.14), and peer-to-peer models. EthRaiser is an example of a Web3 application.

---

## CROWDFUNDING

Crowdfunding is the method of financing of a project, business or charity by pooling donations and backings from a large crowd. This method of financing is well sought after especially for charities and start-up businesses. Crowdfunding is done through websites or platforms which allow for a large audience. Crowdfunding for businesses hosts several advantages over methods of financing such as banks or investors. The first and foremost benefit of crowdfunding as a finance method is Validation. Putting out a product as a campaign, acts as a great marketing technique as well as fundraising. If the product attracts attention, it validates the viability of the product itself as well as raises funds for it. Creating such campaigns also helps garnish an audience or community for the product or the brand as the funders have a sense of unity when all funding the product together. A community can also provide critical feedback on the product itself. Crowdfunding comes without the drawbacks and risks of having to seek investment from equity-hungry (Appendix 1.12) investors or interest-hungry banks. Appealing to a large audience also increases the cap of the funds that can potentially be raised. When a campaign is released, the campaign owner has a way to raise revenue even before selling the product itself. There are four main types of crowdfunding:

- Donation-based
- Reward-based
- Equity-based
- Debt-based

Donation-based crowdfunding is the backers donating the funds without expecting anything much in return. This model of crowdfunding is more for charitable organizations. Examples of donation-based crowdfunding platforms are Kickstarter, GoFundMe, IndieGoGo. Reward-based funding is where entities raise funds in exchange for some reward that isn't exactly financially beneficial. This model is sought mostly after by digital creators or product-based entities. They reward items like early access to videos, pre-order products of limited stock, access to community, etc. Examples of such platforms are Patreon and YouTube Membership, or Twitch. The Equity-based crowdfunding model is about giving pieces of equity to investors for their funds. This model is mostly sought after by entrepreneurs and startups. Here the benefit to the crowd is the equity of the company whose price could increase or decrease based on the company's success. This model also keeps the major ownership in the hands of the company. Examples of Equity-based crowdfunding platforms are CircleUp, Fundable, and CrowdFunder. Debt-based crowdfunding is sought after by entities that do not have enough success or qualification for taking a loan from a bank, so they turn to taking a debt from a large crowd by promising interest in return. Examples of debt-based crowdfunding platforms are LendingClub, Funding Circle, and Kiva.

## MOTIVATION & RESEARCH PROBLEM

Traditional crowdfunding platforms come with significant limitations and drawbacks [**Problem Definition**].

- Donation-based crowdfunding limits the amount of donation that can be received since there is no incentive for the backer apart from satisfaction or goodwill.
- A major issue in most types of crowdfunding platforms is the legal and regulatory requirements. Each type of crowdfunding has its own challenges against legal and regulatory requirements.
- Most crowdfunding platforms use traditional methods of banking and finance which come with its own issues such as lack of transparency. Once the funds leave the backers and then to the business, they don't have any view over where the funds are going or how they will be used. This creates a chance of fraud and reduces the accountability of the campaign owner as their moves wouldn't be visible to the public or criticized.
- Crowdfunding platforms bear upfront costs, maintenance, and overhead for the servers and data centers for their applications. This cost translates to fees accounted to the backer or campaign owner.
- Global accessibility is limited in current crowdfunding platforms as they use traditional banking and finance systems. Platforms cannot have channels for investment in all the various countries in the vast globe, which adds a limiting factor to the potential of the crowdfunding campaign.
- On a centralized crowdfunding platform, the fees of transactions could be high due to the intermediaries. This adds to the lack of global accessibility limitation as international transaction fees can be high or insecure. This also means less of the backer's money is going to the campaign which is detrimental to the backer and the campaign owner.
- Adding on to the transaction fees, transaction durations also increase when using traditional systems for money transfer.
- Personal information can be stored on crowdfunding platforms, and since finance and money is involved, this is a significant limitation. Lack of personal privacy is also a concern when storing personal information. Hacking and frauds can occur causing a detrimental impact to the backers and campaign owners.

Such factors put together put a bottleneck on an idea or campaign's success by discouraging potential backers and campaign owners. In the new world, there are loads of new ideas and products which have huge potential in bringing a positive impact in the society or lifestyle. Many such projects and ideas don't have the right means to secure funding and make a global impact and the limitations of crowdfunding are adding to the matter. Apart from impacting the society or lifestyles of the audience, the owner of the product idea is also being affected. For many a business or a product idea is the only choice to gain success and enough funds to feed themselves or their family. The limitations also affect charitable organizations, which is the most negative aspect of it all. With such concerns it is rather important to add another channel to counter these limitations and make a positive impact in the lives of the people.

To summarize, the **stakeholders** affected by the limitations are:

- Entrepreneurs and self-employed people and the people depending upon them.
- Businesses and its employees
- Innovators
- Charities and the people depending on them.
- Backers and investors.
- Community
- The crowdfunding platform and its people.

Project EthRaiser [**Aims**] is set out to:

- Mitigate the challenges faced on the application of legal and regulatory policies on crowdfunding systems.
- Add transparency and traceability to every transaction made within the crowdfunding system.
- Make accessing and using the crowdfunding system worldwide.

- Decrease costs to the crowdfunding platform, backers, and campaign owners due to the upfront, maintenance, and overhead costs of the servers/data centers required for the crowdfunding platform applications.
- Decrease transaction fees so that more funds stay with the backer or go to the campaign owner.
- Decrease transaction times by a lot to encourage more users and speed up the whole pipeline.
- Decentralize the processes so that there is no room for human inefficiency or attempt to fraud which also means transaction fees and times decrease.
- Build a secure way for backers and campaign owners to use the website by eliminating the need for personal information to be stored.
- Propose more ideas that can increase the success of crowdfunding systems, specifically in how to encourage more use by providing more incentive or benefit to users.

#### By [Objectives]:

- Providing a way to apply legal and regulatory policies to crowdfunding systems by putting the rules directly into the code in Smart Contracts.
- Using the Ethereum blockchain to make sure each and every transaction within the crowdfunding will be visible in any Ethereum blockchain explorer.
- Using the Ethereum blockchain and a worldwide wallet provider: Metamask to access the website's donation and campaign creation functions.
- Removing the intermediaries using the Ethereum blockchain transaction times and costs are reduced.
- Removing the need for servers and data centers reducing costs and fees for all.
- Creating a system where no personal data needs to be stored.
- Decentralize the whole process by using the Ethereum Blockchain and Smart Contracts.
- Build a working donation-based crowdfunding platform.
- Propose a system that involves the crowdfunding platform's own cryptocurrency to be used as governing token on it which would have some value hence incentivize the users.

The use of the blockchain greatly advances on crowdfunding and its limitations, however, to fully understand the topic related works, need to be reviewed.

The main **research** topic is: What is Blockchain and Ethereum? How is it used in finance and crowdfunding and what is the viability of it legitimately fixing the current limitations of crowdfunding? Does it come with its own limitations?

## LITERATURE REVIEW

### INTRODUCTION

#### OVERVIEW

The Blockchain technology is described as a distributed and decentralized internetworked system that makes transparent and secure transactions between parties possible without having the involvement of any entities in between. As described earlier, the internetworked system can be seen in the ledger that is maintained by the network of nodes (miner nodes) which validate transactions through proof-of-work or proof-of-stake and record it. These transactions are tamper-proof which makes them immutable which makes the data authentic (Swan, 2015).

The Ethereum blockchain, is a programmable platform which is mainly used for creating smart contracts. Smart Contracts are self-executing contracts that execute themselves automatically given an input or



satisfaction of terms and conditions given in the contract. Smart contracts are used in applications such as finance, investment, trading, lending, etc., without the need of an authority in between (Buterin, 2014).

The design and programming of smart contracts on the Ethereum blockchain is done in Solidity. Solidity is a statically-typed programming language similar to Java, C++ and Python. The development and deployment of smart contracts is challenged by a few security risks and vulnerabilities. This includes development errors, network attacks, and data manipulation. Smart contracts need to be thoroughly tested and audited which could be done on test networks like Goerli, Rinkeby and Sepolia (Kshetri, 2018).

As said earlier, smart contracts are extensively used in the applications of finance. The elimination of the need for a third party or human factor, increases efficiency, speed, and accuracy. Specific applications include but don't limit to, Real estate transactions, corporate governance, stock and commodity trading, loaning, and crowdfunding. In crowdfunding applications, the usage of smart contracts is done to automate fund transactions, handovers, and related processes (Wright & De Filippi, 2015).

The traditional systems for finance, investment and crowdfunding can be transformed by Smart Contract systems in Ethereum. The problem definitions with the current system will be described in later parts of the report. However, it is important to note the issues and challenges that come with using smart contracts for this application such as legal, regulatory, and technical challenges.

---

## SCOPE OF LITERATURE REVIEW

This literature review critically examines and evaluates the potentials and shortcomings of blockchain technology, Ethereum, smart contract development, their implementations, specifically crowdfunding. It will also provide an overview and with some depth of the concepts involved, the analysis of the characteristics, challenges, implications, of decentralized finance applications. The review will also shed light on domains of further research and accentuate the evaluation and implementation of these systems in a critical and informed manner.

## BLOCKCHAIN & ETHEREUM

---

### CHARACTERISTICS OF BLOCKCHAIN TECHNOLOGY

According to Swan (2015), dominant characteristics of blockchain technology are decentralization, transparency, and security. These characteristics spell the significant potential of blockchain technology for finance and crowdfunding applications. Decentralization allows distributed systems that prevent the need for a central authority controlling the data and transaction over the network. The data and transactions are resistance to censorship, hacking, and other forms of manipulation. This enhances the privacy and security of user transactions and reduces the risk of corruption or fraud. Crosby et al. (2016) note that transparent records and verified transactions which have visibility, auditability, and immutability are enabled by blockchain technology. This enhances trust between the two end parties involved, and significantly reduces the cost of having intermediaries. It can also enable better accountability, auditability, and traceability of funds. Beck and Müller-Bloch (2017) explain that blockchain technology due to its cryptographic mechanisms, consensus protocols, and distributed system, the security is significantly increased. Use of digital signatures (on transactions) and public-key cryptography can provide secure authentication and verification mechanisms for transactions. Within the applications of crowdfunding, various conditions can be set which will release the funds only when they are met, this would further enhance security and integrity. Tapscott and Tapscott (2016) argue that while blockchain technology has the potential to transform various industries, it also has some limitations and challenges. For instance, decentralization makes it challenging to come to agreements and settle disputes. Blockchain's transparency can also lead to privacy issues and regulatory difficulties, particularly when dealing with sensitive or personal data. The security in blockchain technology, like in any other technology isn't

absolute, social engineering attacks or technical vulnerabilities are part of the security risks. (Antonopoulos, 2014) imply that, Ethereum is composed of a flexible and modular architecture, which makes integration with other systems and protocols simpler.

---

## ADVANTAGES AND LIMITATIONS OF ETHERUEM

Ethereum has the upper hand over other blockchain platforms such as Cardano, Polkadot, Solana in smart contract support and decentralized applications (dApps) as per (Swan, 2015). According to (Beck & Müller-Bloch, 2017) Ethereum smart contract development is more widely adopted which allows an open-source community, supporting innovations and development of new applications.

However, there are limitations to Ethereum as well. A prime disadvantage is scalability. Due to the Proof-Of-Work consensus mechanism that was being used, when transaction density increased on the network, the network was slowed down and gas prices increased, making it more expensive to use. The consensus mechanism is being switched to Proof-Of-Stake which is generally more energy-efficient, security (reduced risk of centralization). In (Atzei, Bartoletti, & Cimoli, 2017), security issues with Ethereum have been raised due to past vulnerabilities in smart contracts. (Reynolds, 2018) notes that, Ethereum has been questioned with some critics arguing the environmental unsustainability of the resource-intensive consensus mechanism. As per (Antonopoulos, 2018), platforms such as EOS and Cardano compete with Ethereum in terms of scalability issues and transaction durations.

---

## ETHEREUM'S LIMITATIONS FOR DEFI

As mentioned earlier, Ethereum is the main leading blockchain platform where decentralized finance applications (DeFi) thrive to exist. The excellent smart contract development environment allows programmable financial products and services. However, Ethereum has several limitations in the subject of decentralized finance and expansion to meet global needs.

As described in (Buterin, 2019), scalability comes with the issue of slower transaction times and higher gas prices, which leads to the concerns about Ethereum supporting large-scale DeFi applications.

Security limitations for Ethereum come in the form of smart contract vulnerabilities (Böhme et al., 2015), and 51% attacks (Wood, 2014). (Böhme et al., 2015) talks about how in the past smart contract vulnerabilities have been prone to exploitation in several high-profile hacks and scams which have in turn resulted in large financial setbacks.

Environmental unsustainability was an issue, which is now being tackled by Ethereum adopting the Proof-Of-Stake consensus mechanism.

In work (Christensen et al., 2019), we get a perspective which views DeFi applications on Ethereum to be centralized. The governance structure and decision-making processes of a DeFi platform might involve its small group of stakeholders and designers. This leads to possibility of conflict of interest and lack of input from the broader community that use the said platform.

In conclusion, there are several limitations with Ethereum for decentralized finance, and demand for global adaptation. The limitations mainly include, scalability, security, efficiency, governance.

---

## SMART CONTRACT DEVELOPMENT

---

### BENEFITS AND CHALLENGES

Various tasks and business processes are automated with smart contracts. However, there are both benefits and limitations that develop with smart contract development itself (Deloitte, 2018).

Benefits of smart contracts and their development have been mentioned earlier, such as increased efficiency of self-executing, reduction of costs, elimination of intermediary, increased transparency, and security.

A major challenge in smart contract development is code errors which result in unintended vulnerabilities and consequences which can be exploited. Contracts can be written in different languages and over different blockchains, so the lack of standardization can decrease interoperability between different contracts and systems (Buterin, 2019).

Smart contracts in most cases, fail to recognize as legal agreements in the jurisdiction. Due to this limitation, there could be uncertainty which would make businesses and entities overlook using smart contracts in business and finance domains.

The limitations don't completely debar the potential of smart contract development in various industries. However, it is important to approach the development and implementation of smart contracts to best use in a critical and informed manner.

---

## SMART CONTRACT LANGUAGE, TOOL LIMITATIONS, AND SOLUTIONS

As mentioned earlier, smart contracts can be used in automating various financial and business activities, but they do come with limitations with the current methods and state of developmental languages and tools. In this section an analysis of the limitations of current smart contract languages and tools, and areas of future improvements will be done.

As described earlier, a prime limitation with smart contract development is the lack of standardisation within contracts. Contracts are written with different languages and tools, and on different blockchains as well. This prevents or lowers interoperability between them. This is more of a developmental issue and concerns the developers. The introduction or adoption of a single language for smart contract development can help tackle this limitation.

According to (Zhang and Liu, 2018), smart contract development is stated to be more complex as they're written in lower-level languages (Rust, Solidity, Vyper). This could result in increased errors, code vulnerabilities, decreased security, and developmental inefficiencies. The introduction or adoption of a higher-level language that can abstract the lower-level languages which would decrease the development complexity, cost, and time.

Studied from (Atzei et al., 2017), When a particular smart contract is deployed to the blockchain, it is henceforth immutable, making it impossible to correct errors or tweak the contract. A new contract must be deployed to correct the previous one. This could create additional developmental costs in gas price and the need for thoroughly testing and auditing smart contracts before deployment. Introducing upgradeability that allows to upgrade or patch contracts would tackle with this limitation.

(Xu et al., 2020) found that smart contract languages have limited functionality so complex functions like computation, API calls/requests and such aren't possible.

As described before, blockchains which use the Proof-Of-Work consensus mechanism can be hacked with the 51% attack, other such exploitation can occur with code vulnerabilities and so on. The development of more secure smart contract languages and tools along with increased auditing and testing would help tackling this limitation.

---

## SMART CONTRACTS IN FINANCIAL AUTOMATION AND REGULATION

Smart contracts can have terms and conditions directly written into the code which automatically execute. Automation of financial and investment activities is made possible with smart contracts. Saving both parties time and cost.

However, there may be consequences in the domains of investor security, fiscal policy, and economic efficiency. Böhme et al., (2015) state that Sophisticated investors would understand the concepts of smart contract technology better than retail investors, so this creates a grey area in between, where in smart contract investment is less sought upon by retail investors.

It is still true to say, smart contracts can transform the way investment and financial activities are undergone. As said earlier, there are fixes such as standardization, regulation, and simplification that can help tackle the grey area described better by Pilkington, (2016).

## FINANCE & INVESTMENT APPLICATIONS ON ETHEREUM

---

### POTENTIAL AND CHALLENGES OF ETHEREUM-BASED DEFI ECOSYSTEMS

The majority of the Ethereum blockchain's applications are built for decentralized finance or DeFi. As described earlier, the main advantage being the absence of intermediaries and accessibility to anyone with internet and a wallet. DeFi systems have their own drawbacks and limitations which would be described below.

One of the major challenges in the past was scalability which existed due to the Proof-Of-Work consensus mechanism which limited the number of transactions, increased costs, and transaction times at peak as critiqued by Schär, (2021). Zohar & Eyal, (2015) noted that Ethereum 2.0 uses the Proof-Of-Stake consensus mechanism which mitigates the scalability problem.

As described earlier, smart contracts are immune to vulnerabilities and exploits which may result in loss of Ether. There was a DAO hack in 2016 (Atzei, Bartoletti, & Cimoli, 2018), which will be described later in the review. To mitigate such risks DeFi projects should undergo immense security audits and use best practices, and efforts for smart contract development.

Another limitation is the challenge of regulatory policies. DeFi applications are growing to be more mainstream and can handle large amounts of Ether, this comes with doubt and scrutiny from regulators. The absence of explicit regulatory frameworks for the DeFi applications create challenges in compliance which could lead to legal and regulatory issues as stated by (Nebbia,2021).

Despite the potential the mainstream adoption of DeFi applications is slow. A deep technical background in blockchain technology and cryptography is required to understand the development and usage of DeFi applications, this acts as a grey area again for users that don't have the background. According to Brody, (2021 DeFi applications' UI/UX are considered unintuitive and confusing for some which again act as a reason for slower mainstream adoption.

On the whole, while DeFi has the ability to revolutionize finance, It will take constant innovation and participation within the Ethereum community as well as globally to overcome these obstacles.

---

### RISKS OF INVESTING IN ETHEREUM-BASED FINANCIAL PRODUCTS

With the increasing popularity of DeFi applications, certain risks come with the usage of the Ethereum blockchain. The risks or challenges associated, and instances will be discussed about below.

A prime risk associated with investment in Ethereum-based financial products is smart contract vulnerabilities as mentioned earlier. Smart contracts aren't immune to error which leads to financial losses in cases of exploitation. An instance is the infamous DAO hack of 2016. This hack caused in a loss of 50 million USD worth of Ether tokens (Atzei, Bartoletti & Cimoli, 2018).

Market volatility, low trading volumes, and liquidity provider insolvency, manipulators, liquidity risks increase. Liquidity is the ease of converting a digital currency or token to another digital asset or cash without significantly affecting the market price. Liquidity is a prime aspect of dEx (decentralized exchange) platforms and other financial applications which involve exchange of asset or tokens. Baron, (2021) noted that We get to see flash loan attacks on various DeFi protocols which accentuate the liquidity risks that come with DeFi applications. These are caused by absence of regulatory oversight and vital mechanisms such as circuit breakers in the loaning process. Makoto & Oshiro (2021), talk about how possible amendments like dynamic circuit breakers, stable-coins, and liquidity pools can counter the issues.

As mentioned earlier, decentralized finance is largely unregulated and investor protection is compromised on a certain scale. This lack of regulatory oversight leads to price manipulation, money laundering, and fraud. Chohan 2021, states an instance of this is the yield farming trend which are a result of unaudited smart contracts and dubious investment strategies. Casey & Vigna 2018, wrote about proposals of regulatory sandboxes, seeking self-regulatory organisations, and development of regulatory frameworks by the critical financial institutions.

---

## ASSESSMENT OF ETHEREUM TRANSFORMING TRADITIONAL FINANCE

Within the field of finance itself, the use of Ethereum in decentralized application comes with its own cons with respect to the finance industry itself.

Elimination of intermediaries is a great benefit of using decentralized finance applications, however, as critiqued by Böhme et al. (2015) there are weighty disadvantages for existing institutions and stakeholders. The phasing out of banks when Ethereum substitutes the financial system would bring upon job losses and financial losses for the bank, and its stakeholders such as the employees or investors.

As mentioned earlier, in the largely unregulated environment of blockchain systems, regulatory authorities have little ability to keep up with regulating the innovations happening in blockchain. As Casey & Vigna (2018) described, companies can take advantage of this regulatory arbitrage.

The scalability issue with Ethereum based decentralized finance applications are also a disadvantage, however, solutions like sharding, layer 2 development, and Proof-Of-Stake consensus can counter the situation.

## CROWDFUNDING ON ETHEREUM

---

### ETHEREUM'S POTENTIAL IN CROWDFUNDING

Continuing from earlier, crowdfunding is treated as an alternative to seeking investment from an investor or bank. As described in works of Gerber et al., (2012), and Belleflamme et al., (2014), the aspect of validation is talked about wherein it is stated about how crowdfunding is used to create a feedback channel. Through which the business can understand if whether the idea really has demand in the market. This helps in understanding what ideas or what part of the ideas would be successful and appealing and which are not. This would let the business have the advantage of releasing a better final product.

Mentioned in Block et al. (2018) and Zhang et al. (2019), the aspect of wider reach is being talked about. As crowdfunding usually involves an internet platform or website it has the potential to reach a very large audience. Kemp (2021), stated about how the internet boasts about more than 4.66 billion active users



daily. The audience could include investors, or just increase the popularity or reputation of the product being marketed. The good reputation will also lead to traditional methods of finance more accessible if they weren't already. This proves to be true in cases such as small startups.

Described by Sornn Freise, 2018 The usage of smart contracts enables various fundraising mechanisms such as Initial Coin Offerings (ICOs) and Decentralized Autonomous Organizations (DAOs), which overcome the need for intermediaries such as interest-hungry banks or equity-hungry investors.

The automation of tasks minimizes space for human error, inefficiencies, and fraud.

Removal of intermediaries and the use of an automated system for transferring funds is time-efficient and cost-saving. Decentralized crowdfunding platforms don't have to bear the up-front cost for servers, data centers, etc. They neither have to deal with the upkeep and overhead to keep these systems running, hence reducing downtimes which increases efficiency and reducing fees which means more goes to the campaign owner or stays with backer (Liu et al., 2018).

---

## CHALLENGES IN CROWDFUNDING

One of the key challenges faced web3 crowdfunding platforms is the difficulty in verifying project quality. In contrast to traditional crowdfunding, where investors can evaluate the feasibility of a project based on concrete elements like market research and prototypes, Ethereum-based crowdfunding projects might not even have a tangible product or could just have a conceptual whitepaper. This creates a significant challenge for investors as it makes it hard to assess the project's quality and its potential for success. Usually on a web3 crowdfunding platform, campaigns would be related to web3 projects themselves. This might create a bottleneck for investors as high-level knowledge about the blockchain and smart contracts would be required to understand or assess the viability and potential of a project. Some investors would prefer having other entities doing the assessment and auditing for them which could cause expenses, and time-consumption for the investor. This could discourage the investor or lose a potential investment or donation to the campaign (Li et al., 2021).

There is risk of fraud though traceable. The nature of Ethereum and the blockchain along with the easily available ERC contracts make it easy for anyone to mint their own cryptocurrency and thereby launch an ICO. This can be used to create fraudulent projects. Like in traditional platforms, even here there could be fraudulent projects that appear legitimate due to their marketing, whitepapers, social media presence, staged advertising. However, like traditional platforms there are projects that aren't able to raise funds that lack in making themselves look legitimate but are legitimate projects. Some projects are inaugurated with goodwill and legitimate promises, however, lack of skills or resources, bad timing, and other such factors can cause failure (Böhme & Christin, 2018).

Since cryptocurrency is a volatile asset, whose price keeps fluctuating rapidly over the course of time, it might be a challenge to set investors or campaign owners to rightly assess what is the fiat value of their campaign funds.

---

## CROWDFUNDING MODELS WITH ETHEREUM

Donation-based crowdfunding remains vastly the same between traditional and Ethereum-based crowdfunding. Continuing liquidity risks, it is more difficult for to assess the exact dollar value being donated or has been donated. However, methods such as technical analysis can optimize income for the campaign. The value of the donations received can be held as cryptocurrency itself and liquidated when the price of the cryptocurrency is in a bull run. This makes more out of the donations that have been made (Mao, 2021).

Reward-based crowdfunding can be made more efficient by using the Ethereum blockchain. Rewards can be disbursed in cryptocurrency itself, which is more efficient, and creates a value stream for the project itself. Using proprietary cryptocurrency as a reward or governance coin on a project, creates marketing and rise in value of the project. Marketing is a result of the cryptocurrency circulating on crypto exchanges (Pilkington, 2019).

Equity-based crowdfunding can be done with the cryptocurrency acting as the tokens of owning the company or stake in the project. The tokens can be traded on cryptocurrency exchanges based on the value of the equity. Another model to equity-based crowdfunding on Ethereum is to create a decentralized application wherein the backers can (who have received token as equity) can stake their tokens to earn dividends proportional to their token balance (Liu, Li, & Liang, 2020).

Debt-based crowdfunding can also use the concept of cryptocurrency staking. Backers can put up their tokens to receive an interest on their staked amount, however, there is significant risk involved in debt-based crowdfunding so robust systems need to be made to prevent inefficiencies (Chen & Zhang, 2021).

Real-Estate crowdfunding is made greatly efficient with the use of smart contracts and the Ethereum blockchain. A real estate crowdfunding platform can create a decentralized application where it is possible to exchange, buy or sell their pieces of real estate. The use of smart contracts greatly reduces the chance of fraud and increases accountability. Liquidity risks can still be associated as the value of Ethereum would fluctuate, creating ambiguity in the market (Kokkinaki and Papadopoulos, 2019).

## CROWDFUNDING ON ETHEREUM

The findings of this literature review provide a critical aspect on the potential of Blockchain and Ethereum as an alternative to traditional systems. Various literature and their opinions were mentioned. As with every solution, even Blockchain and Ethereum have their drawbacks and limitations. However, solutions to the drawbacks also have also been discussed, though they require more research and testing before they could be implemented. To summarize the review suggests that while the technology has significant potential, it will require continued development and regulation to fully realize its benefits and mitigate its risks. Domains in which further research should be conducted are mentioned below.

Security within smart contracts, Further research is required to address vulnerabilities and ensure the security of Ethereum based applications, as smart contracts are vulnerable to issues such as bugs, hacks, and programming inefficiencies.

Privacy, though and it is an advantage to be a transparent and immutable system, not every application would require transparency. Hence, research is to be made in techniques of reserving privacy. Current fields of research include zero-knowledge proofs, ring signatures, and homomorphic encryption.

Consensus Mechanisms, existing mechanisms such as proof-of-work bring inefficiencies such as energy consumption, costs, complexity, and lack of network security.

Regulations and Policies, ways of regulating the use of smart contracts in financial and other such vital systems is needed to ensure investor protection, fraud prevention, promoting innovation and integration with traditional finance systems.

Scalability, research areas include new scaling solutions such as sharding, layer 2 technologies, and sidechains.

User education and adoption, the use of the blockchain in a manner which ensures security of the user is vital. Research fields include methods of educating the masses about the security practices which reduce risks of privacy breaches and security violations.

## ETHICAL ISSUES

The consequences of the project are not favourable in all cases, this section talks about the negative consequences and ethical issues that come up as a part of EthRaiser.

- Automation leading to loss of jobs: The job roles that have been removed due to the automation of the system are mainly bank employees and crowdfunding platform employees. Like the stakeholders of EthRaiser, these employees also depend on their jobs for a livelihood, and the loss of their job could be detrimental.
- Anonymity: The fact that it is easy to create a wallet and make transactions on the blockchain without giving out any personal information adds anonymity. When the use of Web 3.0 applications peaks, there might be applications where it might be easy to commit a crime or unethical act and get away. This could also lead to danger to life.
- Immutability: The immutability of the blockchain gives it significant power, this could result in unneeded immutable contracts that could cause harm to existing systems.
- Sustainability: Although this issue is to be mitigated with the adoption of PoS, for now the PoW causes a significant amount of energy usage and pollution.

EthRaiser believes the benefits outweigh the negatives, hence moving forward with the project. However, EthRaiser is motivated towards improvement and mitigation of the negatives in the near future.

## DESIGN DEVELOPMENT AND IMPLEMENTATION

### PRE-DESIGN

As per the objectives the deliverables of this project are:

- A donation-based crowdfunding platform built on the Ethereum blockchain with a web front-end.
- A proposal to incentivize backers to support a campaign.
- A technical report for the project.

A work-breakdown structure [**WBS**] is given below with the overview of the tasks which will then be put as milestones.



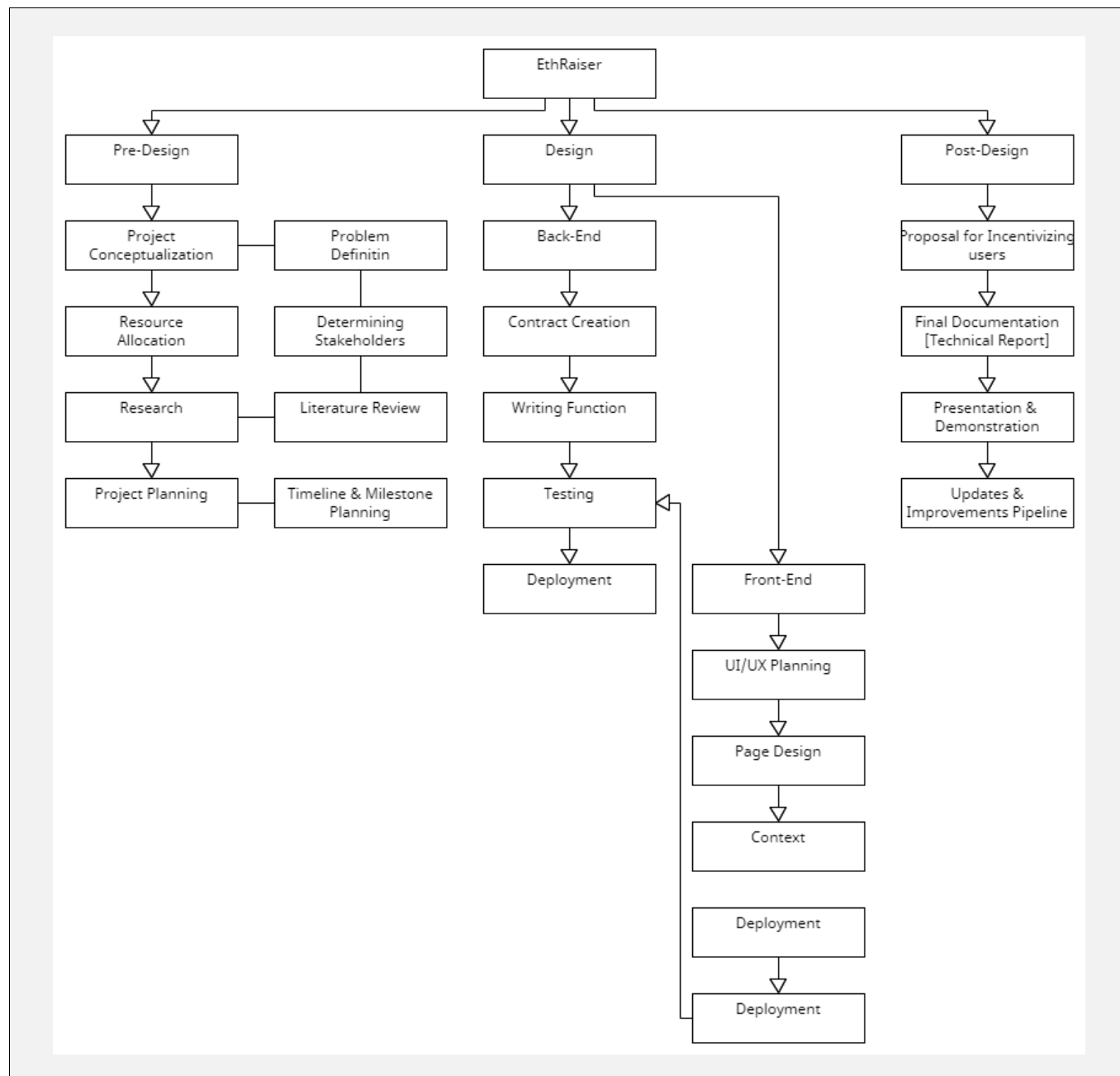


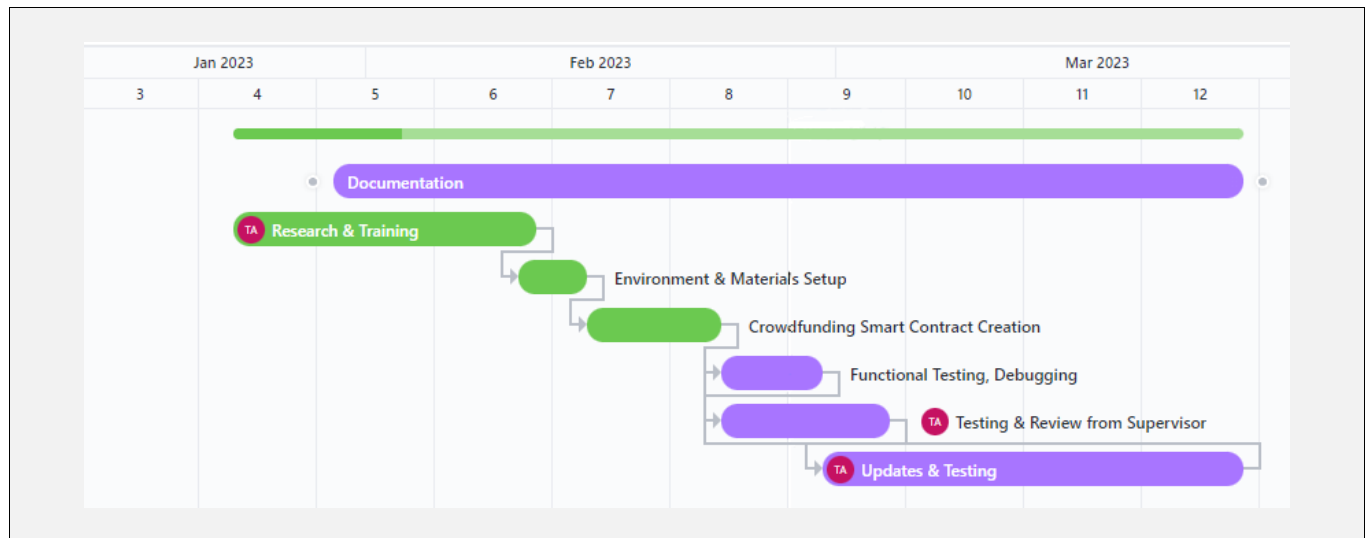
Figure 3.1 (Work Breakdown Structure)

The project development will consist of the following **milestones**:

Task	Sub Tasks	Start Date (DD/MM)	End Date (DD/MM)	Duration (Days)	Remarks
Research & Training		24/01	10/02	17	
Documentation	1] Technical Report 2] Weekly Activity Log 3] Blog 4] Meeting Form	30/01	24/03	25	15K-25K words Technical Report, Weekly Activity Log on Google Docs, Blog, Meeting Form on Google Docs [Links in Appendices]

Environment & Material Setup	1] Install required dependencies for JavaScript (ThirdWeb) 2] Create MetaMask wallet and fund with Goerli Test ETH. 3] Installation of VS Code with extensions (Prettify, ES7+,)	10/02	13/02	3	npm, npx. Metamask.io, Google Goerli ETH faucet
Crowdfunding Contract	1] Smart contract creation 2] Write Functions for: <ul style="list-style-type: none"> <li>• Creating campaign.</li> <li>• Contributing to campaign.</li> <li>• View backers for campaign.</li> <li>• View all available campaigns.</li> </ul>	14/02	21/02	7	Done on Remix IDE.
Functional Testing & Debugging	1] Deploy on test net [Goerli] 2] Functional Testing 3] Documentation	22/02	27/02	5	Done on Remix IDE, deployed on Goerli test net, test environment, and results documented in technical report.
Testing & Review	1] Test with supervisor 2] Get feedback	22/02	03/03	9	In supervisor meeting brief and test contract functions and receive feedback.
Updates & Testing	If time permits: 1] Add NFT donation code (and auction) 2] Create Front-End for contract.	28/02	24/03	25	
Updates & Testing: Front-End Development	1] Setup application with ThirdWeb 2] Build main-landing page 3] Set context (interface to contract) 3] Create all campaigns page 4] View my campaigns page 5] View single campaign page 6] Donation Page 7] Deploy with Vite	28/02	24/03	25	Using React.js, HardHat, ThirdWeb, Vite, TailWindCSS, deployment and testing on local network.
Updates & Testing	1] Updates 2] Deploy with Vite 3] Testing	28/02	24/03	25	

The planned timeline (**Gantt Chart**) of the project development is given below:



3.2 (Gantt Chart)

## RESEARCH METHODOLOGY:

The research involved in the project EthRaiser is applied research in blockchain engineering and software engineering. Applied research is the application of the knowledge and technologies gained through research to mitigate or solve a problem. EthRaiser aims to solve the real-world problems (mentioned in the problem definition) in traditional crowdfunding platforms by creating an implementation of a crowdfunding system on the blockchain and a proposal to further incentivize blockchain crowdfunding platform users. The research tasks involved in the project are as follows:

- Exploratory Research (Qualitative) for:
  1. Blockchain technology concepts
  2. Ethereum technology concepts
  3. Smart Contract development and deployment methods and concepts
  4. Decentralized finance application concepts, needs, and stakeholders
  5. Crowdfunding platforms, concepts, needs, and stakeholders
  6. Case Study of Heera Group of Companies
  7. UI design paradigms and concepts (minimalist design, colour psychology)
  8. Front-end development in JavaScript concepts, and methods using React, TailWindCSS, Vite, ThirdWeb
  9. Concepts of tokenomics, supply & demand, and trading.
  10. Technical report writing for engineering applications.

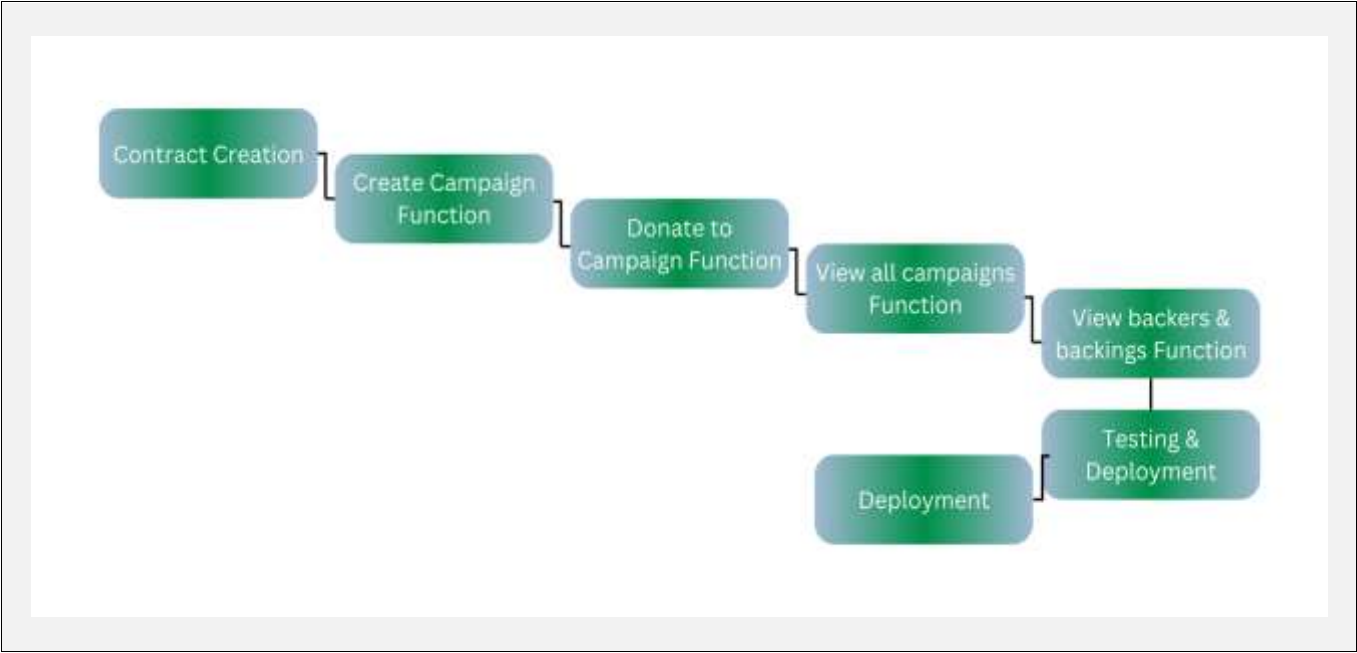
## DEVELOPMENT METHODOLOGY

The development of EthRaiser is done in two phases:

- **Back-end:**  
Involves the material setup, contract creation, development of contract functions, testing of contract functions and deployment of contract on Goerli Test Chain.
- **Front-end:**  
Involves material setup, development of different pages and elements for different pages of the UI, importing contract, adding functionality to elements, Testing functions.

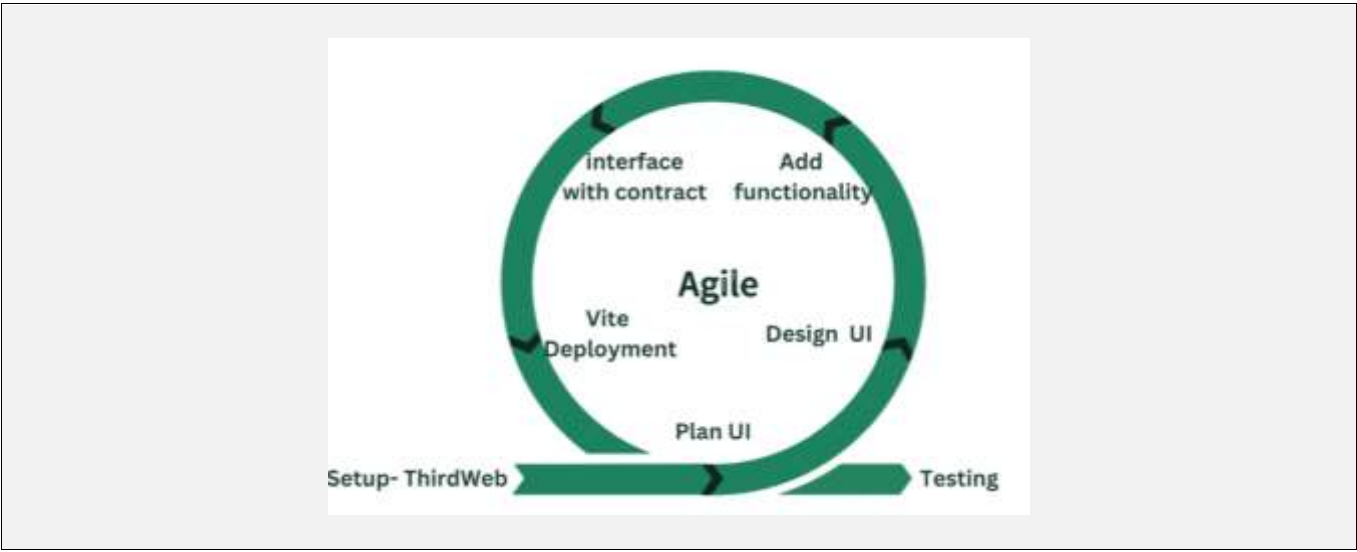
Constant critical feedback has been received from **Mr. Judhi Prasetyo, Project Supervisor** regarding the development of the project and the features.

The Back-end phase follows a **waterfall method** of development as the tasks are linear. A visualization of the process is given below:



3.3 (Waterfall method for backend development)

The front-end phase involves iterative design to make it function, look, and appeal better. The developmental process adapted to is the **Agile approach**. A visualization of the process is given below:

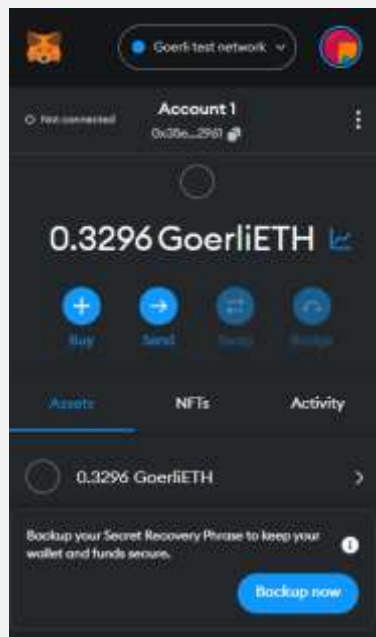


3.4 (Agile method for frontend development)

RESOURCE & MATERIAL ALLOCATION

METAMASK

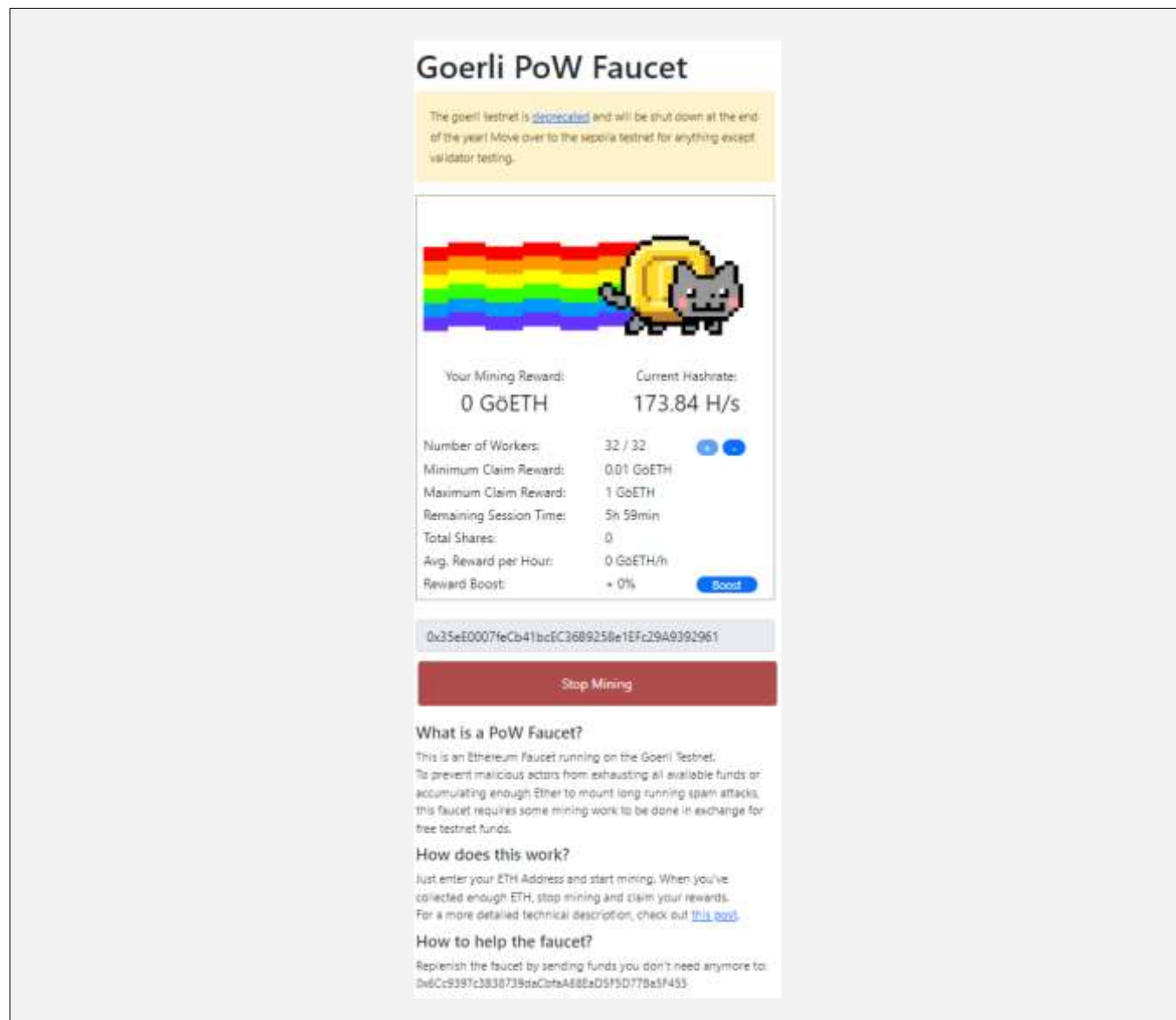
[Metamask](#) is a software wallet that is available as a chrome extension and a mobile application. It creates a wallet address with a private-key and public-key. In the development of EthRaiser, the chrome extension version of Metamask is used.



**Note:** The wallet should be configured to the Goerli test network in Settings

## GOERLI TEST ETH (GOETH) TOKENS

Goerli test ether is available on various faucets on the web. However, as of the time of development the test ETH has been hard to acquire even for free. The faucets used in the development of this project are the [Alchemy Faucet](#), and [Goerli PoW faucet](#). The Alchemy faucet simply requires an Alchemy account to retrieve 0.02 GoETH per day. The PoW faucet runs a miner node on the browser itself and pays GoETH for the computational work.



## FIGMA

Figma is available on the [Figma website](#), a google account is needed to access the instruments and save the work.

## CANVA

Canva requires a google account as well and is available on the [Canva website](#).

## VISUAL STUDIO CODE

Visual Studio Code is available for free at [Microsoft](#). ES7+ Redux, Solidity Compiler, prettify, and JavaScript compilers are available as extensions within the IDE itself.

## REACT.JS

Through ThirdWeb CLI the React application template is created (discussed in the Design section).

## NODE PACKAGE MANAGER (NPM)

Node.js and npm are obtained on the [website](#).

---

## HARDHAT

ThirdWeb CLI interfaces with HardHat. When creating a contract template using ThirdWeb CLI, HardHat is a choice of framework in the CLI prompts.

---

## THIRDWEB CLI

ThirdWeb CLI is accessed through npx commands.

```
Npx thirdweb@latest <commands>
```

---

## VITE

ThirdWeb CLI will set up the Vite scripts. When using ThirdWeb CLI to create application template it will prompt Vite as a choice of framework.

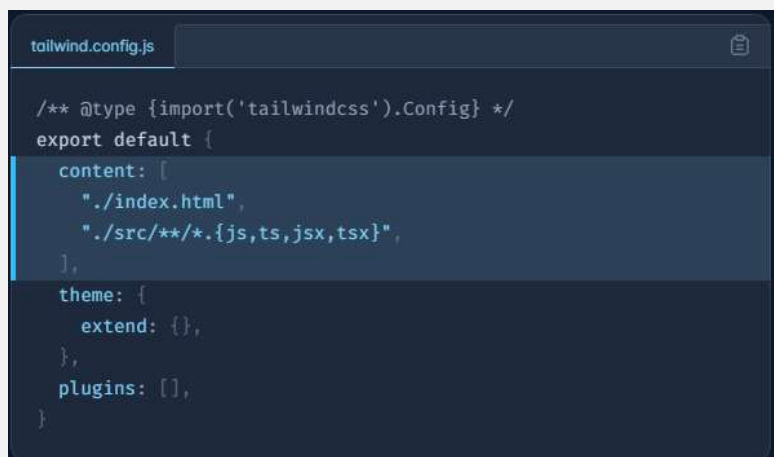
---

## TAILWINDCSS

Tailwind can be installed using Vite. Refer to [documentation](#).

- ```
1. npm install -D tailwindcss postcss autoprefixer
2. npx tailwindcss init -p
```

From 'tailwind.config.js':



```
tailwind.config.js

/** @type {import('tailwindcss').Config} */
export default {
  content: [
    './index.html',
    './src/**/*.js,ts,jsx,tsx',
  ],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

---

## DESIGN

---

### DESIGN ELEMENTS

---

#### THIRDWEB

ThirdWeb is an open-source library that has a wide collection of resources and tools that assist web3 developers in making decentralized applications. Usually, the development and deployment of smart contracts for web applications is a lengthy and laborious process that uses many different components. ThirdWeb aims to make this process intuitive and welcoming to more developers. ThirdWeb can also make it possible to import contracts into the web application and interact with the functions of the contract through the application. In this project ThirdWeb is used to create the contract template, deploy using HardHat framework, import contract, and interact with contract functions in web application. ThirdWeb CLI is a way to access the ThirdWeb features using the command line.

---

## REACT.JS

The React.js library contains various tools and resources that aid in front-end development (User Interfaces). React was created by Facebook and is a popular front-end development library due to its versatility, ease of use, and efficiency. React allows developers to create components that are reusable and editable at each instance if needed, this allows for complex designs. React has a declarative syntax. React has its exceptional efficiency in updating UI with user interactions using virtual DOMs (Appendix 3.1). React is interoperable with a wide number of frameworks and technologies making it flexible even in large scale applications. Due to the wide use of React, there is a vibrant and innovative community that helps the developers as well. React will be used in the front-end development of this project.

---

## VITE

Vite is a tool to build and deploy development servers and host web applications. It was created by Evan You using Vue.js. With Vite it is possible to compile and host the web application using the dev script easily on your localhost or a network. Vite also has the feature of Hot Module Replacement, which allows developers to see the changes they have made in the code live on the front-end. Vite is also interoperable with a wide number of technologies such as TailWindCSS. In this project Vite will be used to deploy the web application on the local host and test the application.

---

## TAILWINDCSS

TailWindCSS is a CSS framework that allows developers to create, style, and customize their components with the user of pre-defined classes and styles. The styles are directly applied to HTML and the codes in React which makes it easier for developers to stylize their Uis. The various pre-defined styles and classes have intuitive documentations as well. In this project, TailWindCSS will be used in the building and styling of the UI of the web application.

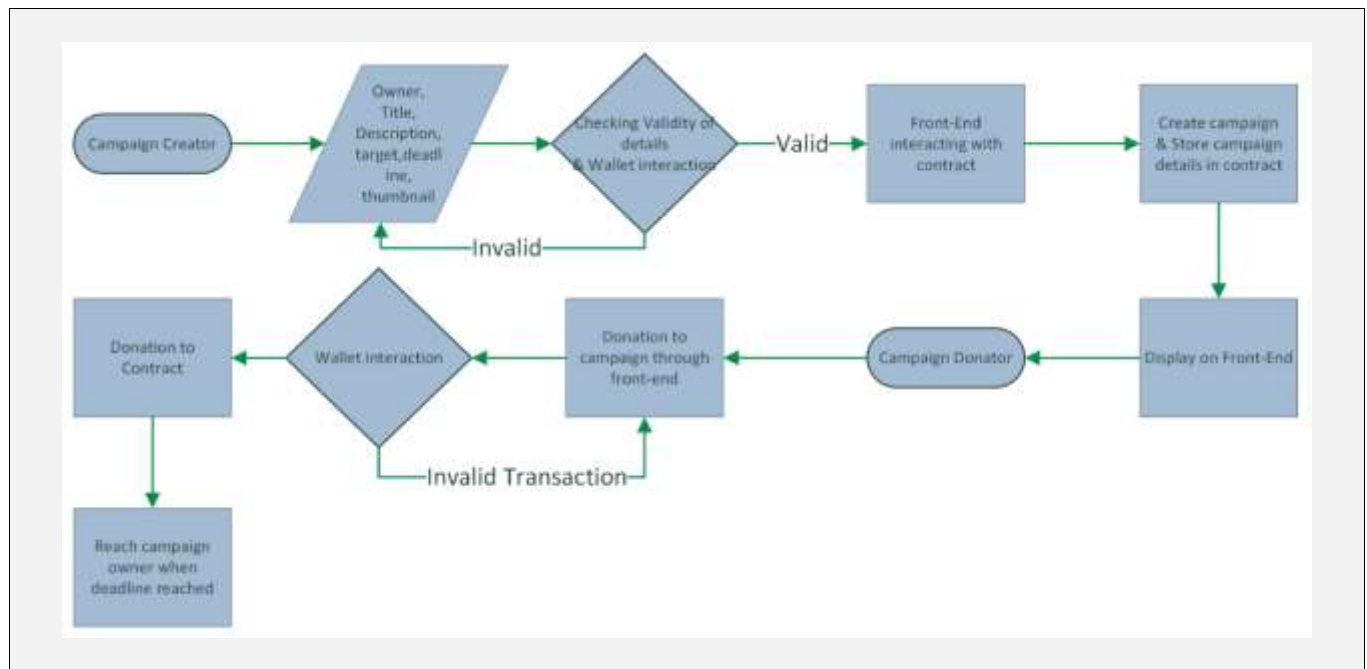
---

## HARDHAT

HardHat is a development environment for building, testing, and deploying smart contracts in the Ethereum blockchain. HardHat's main feature is its support for contract testing. It allows developers with unit testing, integration testing, and end-to-end testing of the smart contract's. It also comes with a shell to interact with contracts, a debugger, and a contract template generator. HardHat is also used in the deployment of contracts to Ethereum nets such as the main net and test nets such as Goerli, Sepolia, and Rinkeby. In this project HardHat will be used to create the contract template and deploy it to the Goerli test net using ThirdWebCLI.

The react application Interfaces w"th t'e contract to carry out the functions of the platform. The contract acts like a server in a web2 application. However, less computation is required as all the verification and transactions happen on the blockchain through miner nodes. Below is a flow-chart describing the functioning of the crowdfunding application.





3.5 (Flowchart for EthRaiser)

The design process starts from the backend, the contract is set to have 4 public functions.

- Create campaign.
- Donate to campaign.
- View all campaigns.
- View backers along with their backed amounts for each campaign.

As mentioned before in Solidity, OOP, and Functional Programming (FP) paradigms will be used. Each campaign will be of the composite custom data type created call struct (OOP paradigm).

## CONTRACT DESIGN

An optional yet important statement at the start of each contract is to mention the licensing model of the contract. The '// SPDX License-Identifier: <license model>' is a comment that is meant to hold the license model, this specifies the terms and conditions to be followed when distributing or reusing the code of the contract. Here SPDX (Software Package Data Exchange) identifier is Unlicensed, which means this code is not open-source, which means cannot be freely distributed or used without the permission of the owner. These licensing terms are clearly mentioned to mitigate the possibility of confusion or dispute.

The contract is first written in the Remix IDE for testing. Once satisfied, the contract needs to be deployed to the Goerli test-net so that it can be imported into the application. The contract will contain the functions mentioned above. More functions are to be added which will be discussed in the updates and future improvements section.

As mentioned before Solidity is compiled into bytecode which is executed by the EVM. Solidity has different compiler versions. This contract is written for Solidity version 0.8.9 or later.

The contract will consist of a main object (struct) 'campaign', this struct will be used in all the functions. The struct has various attributes of different datatypes (shown below).

```
1 // SPDX-License-Identifier: UNLICENSED
2 pragma solidity ^0.8.9;
3
4 contract CrowdFunding {
5     struct Campaign {
6         address owner;
7         string title;
8         string description;
9         uint256 target;
10        uint256 deadline;
11        uint256 amountCollected;
12        string image;
13        address[] donators;
14        uint256[] donations;
15    }
16 }
```

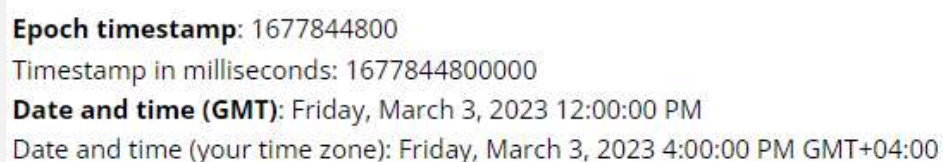
---

### CREATE CAMPAIGN:

When a user accesses the create campaign function to create a campaign, they must provide with their address, a campaign title, a description, campaign target, deadline, and a URL to thumbnail. The picture on the URL will show up in the web application as thumbnail for the campaign. The deadline must be provided as a Unix Epoch timestamp.

Each campaign is given a unique ID, which is done by creating an integer variable and an array of campaign structs. The integer variable steps each time a campaign is created, the campaign is placed at the integer variable index in the array of campaigns.

**Note:** The Unix Epoch timestamp specifies how many seconds have passed since January 1, 1970. This is used as a timestamp for the current block the contract will be uploaded to. An online converter can be used to convert human dates to Unix Epoch. For example, 12pm, 3<sup>rd</sup> March 2023 (GMT+4) is 1677844900 in Unix Epoch.



**Epoch timestamp:** 1677844800  
Timestamp in milliseconds: 1677844800000  
**Date and time (GMT):** Friday, March 3, 2023 12:00:00 PM  
Date and time (your time zone): Friday, March 3, 2023 4:00:00 PM GMT+04:00

The amount collected, list of addresses of donators, and list of their donations will be used in the other functions.

---

### DONATE TO CAMPAIGN

For donating to a campaign, the user must provide a campaign id as an input and send the amount to be donated along with the transaction. The function is payable so it would require the user to connect their wallet with the web application or with the contract. Whenever a user donates to a campaign, their address and their donation amount is put into an array. The index of their address in the address array corresponds to their donation amount in the donation amounts array. Each campaign has its own such arrays of donators and their donated amounts. This allows the web application to track the activity of the users or the campaign. The amount donated is also added into the campaign's amount collected. This will be used to gauge the amount of money the campaign has collected in the web application.

---

## VIEW DONATORS AND THEIR DONATIONS

To view the donators and donations, the user must provide the campaign ID. The function simply views the donators address array and the donations array. This will be used in the web application to track the progress of the campaign.

---

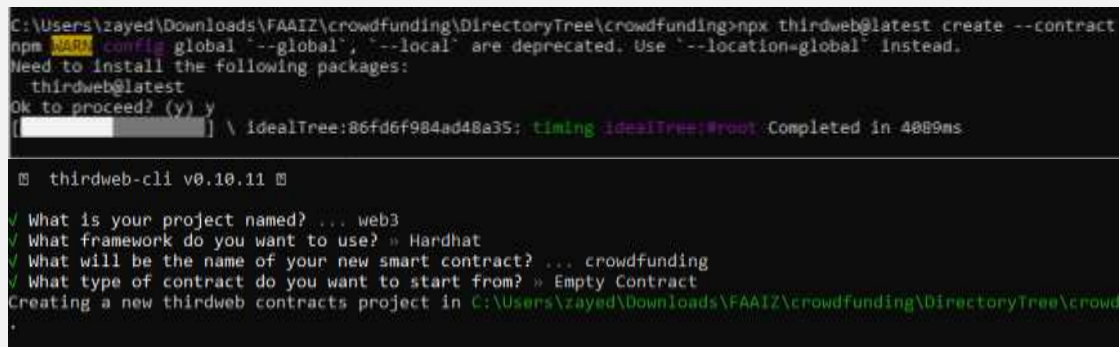
## VIEW ALLCAMPAIGNS

The view all campaigns function will loop through the array of campaigns to give the list of all available campaigns on the current contract. The function will return each attribute of each campaign. This will be used to design the campaign details page in the web application.

---

## CONTRACT DEPLOYMENT

The contract is deployed using ThirdWeb. ThirdWeb will setup HardHat as well, which will be needed for contract deployment. First the contract is put into the contract template given by thirdweb ('npx thirdweb@latest create --contract'). There are configurations for the deployment set in the 'hardhat.config.js' file. It would contain details such as the test net, RPC endpoint, private key(stored as environment variable in a .env file), solidity compiler version, etc. After setting the configurations, the below command is written in CMD (npm run deploy) to start the deployment.



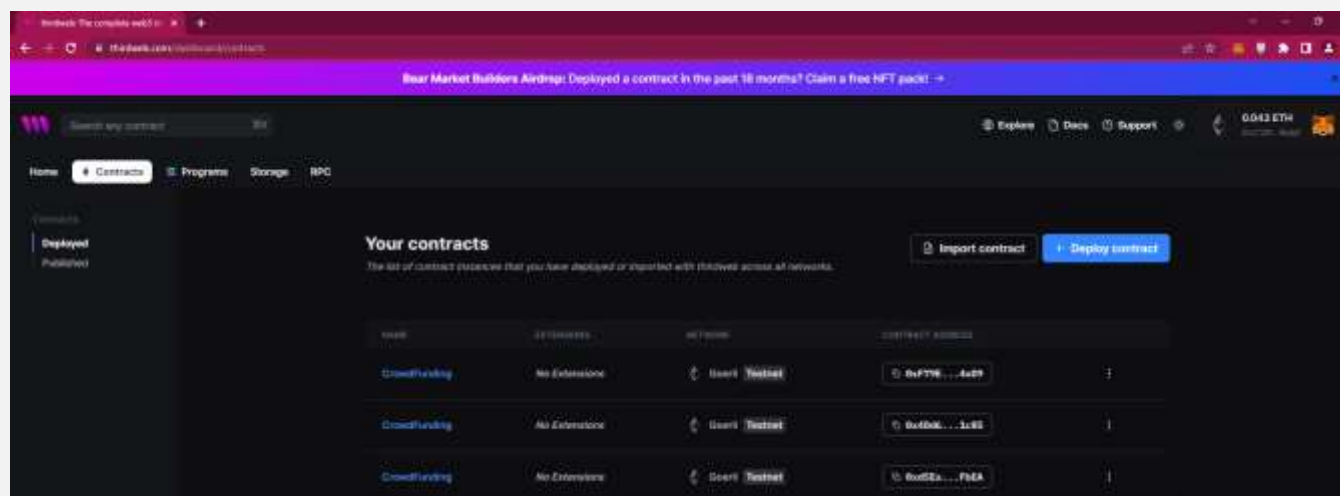
```
C:\Users\zayed\Downloads\FAAIZ\crowdfunding\DirectoryTree\crowdfunding>npx thirdweb@latest create --contract
npm WARN config global "--global", "--local" are deprecated. Use "--location=global" instead.
Need to install the following packages:
  thirdweb@latest
Ok to proceed? (y) y
[ ] \ idealTree:86fd6f984d48a35: timing idealTree:#root Completed in 4889ms

thirdweb-cli v0.10.11

✓ What is your project named? ... web3
✓ What framework do you want to use? » Hardhat
✓ What will be the name of your new smart contract? ... crowdfunding
✓ What type of contract do you want to start from? » Empty Contract
Creating a new thirdweb contracts project in C:\Users\zayed\Downloads\FAAIZ\crowdfunding\DirectoryTree\crowdfunding\web3
```

| name         | extensions    | network        | contract address |
|--------------|---------------|----------------|------------------|
| Crowdfunding | No Extensions | Goerli Testnet | 0xFTW...AaD9     |
| Crowdfunding | No Extensions | Goerli Testnet | 0x0b0k...3i8S    |
| Crowdfunding | No Extensions | Goerli Testnet | 0x0d2a...FMEa    |

Later, the developer needs to confirm the deployment and provide transaction signatures through ThirdWeb's deployment dashboard. The contract is then visible on the developer's deployed contracts dashboard.



The contract address is also given here, which will be used later to bring the contract to the react application.

---

## PROJECT BRAND AND UI/UX DESIGN

To give a unique identity to the project and its contents, it is necessary to create a brand (give it a name and a logo) and design language. The design language with its colours, shapes, and more, can signify the cause or underlying concept of the project. The design language should be uniform throughout the contents of the project which includes the UI/UX design, logo, demonstration, and technical report. The name of the project should signify the traits of the project itself.

---

### PROJECT NAME

The name of the project is straightforward and self-explanatory about the contents of the project. 'EthRaiser', 'Eth' signifying the project being built on Ethereum and the governing cryptocurrency is Ether, and Raiser describing the fundraising aspect.

---

### PROJECT LOGO

The logo should symbolize the gist of the project. Below is a description of the reasons for the design decisions in the logo making process.

- The diamond-like shape of the outline of the logo is to symbolize the use of Ether as the currency of donation and the application being built on the Ethereum blockchain.
- The upwards arrow to symbolize increment.
- Another detail to symbolize increment and growth is the dots which represent entities or businesses going through the arrow and growing from small to big.
- The Green colour is used to symbolize growth and increment.
- The Blue is the colour for Ethereum.



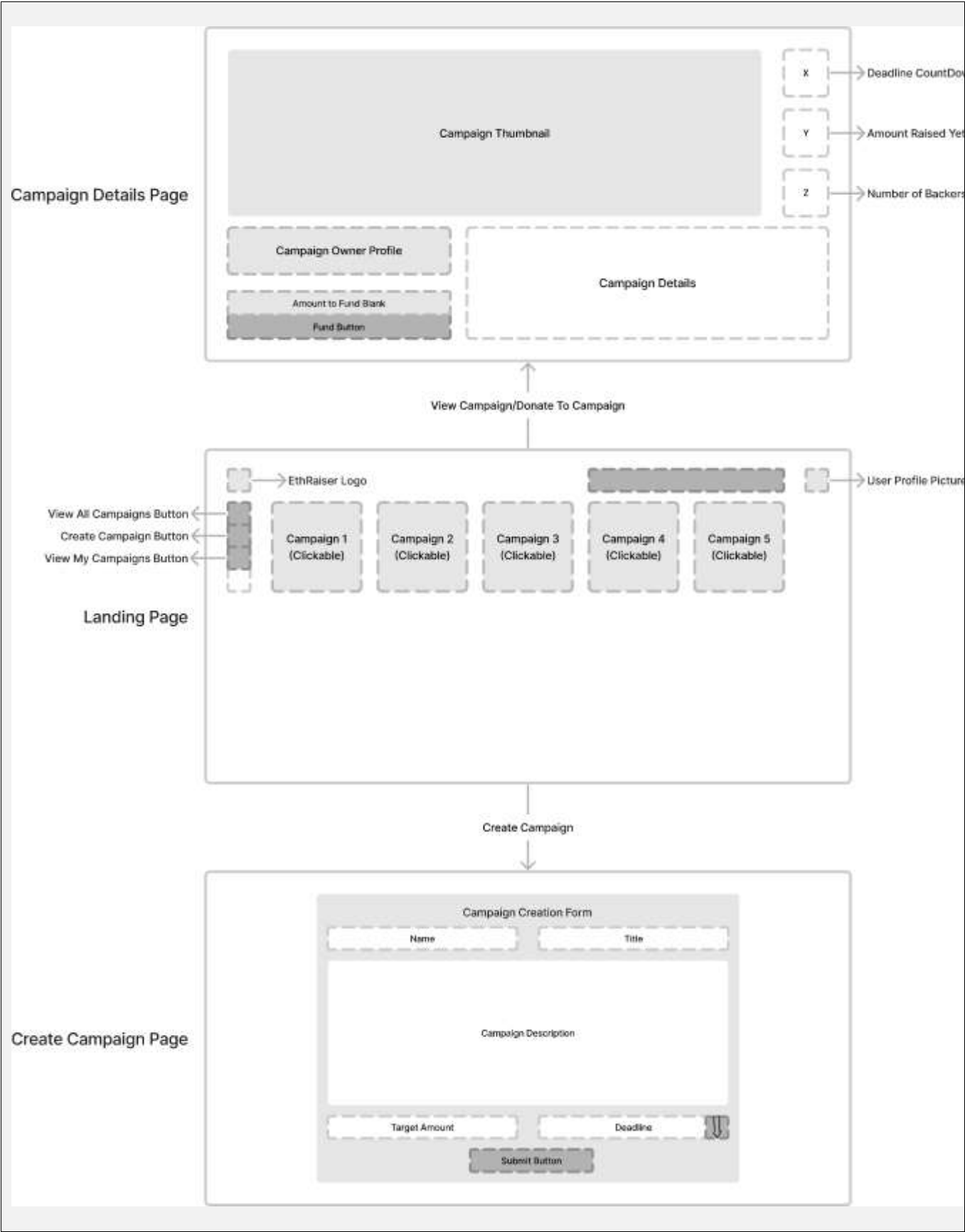
3.6 (EthRaiser Logo)

---

### UI/UX DESIGN

As mentioned earlier, the platform would have 4 main functions, which would be accessible with the front end. To make the process intuitive and easy the design should be minimalistic, clean, and easy to navigate. This project focuses on having the least possible amount of information the user needs to deal with. This makes it very important for the front-end to have less contents and pages on the website.

The website would have 3 pages, within which one page will be only for campaign creation, another page for viewing all campaigns, and the last page for viewing a particular campaign, donating to campaign, and seeing the list of backers and backed amounts. Minimalism and simplicity in design is the key. The Wire Frame for the UX design (3 pages) was made using FigmaJam.



3.7 (UI Wire Frame)

The application begins at the **Landing Page** which consists of the **Connect Wallet Button** on the top right, clickable **Campaign Cards** in the middle, and a **Navigation Bar** on the left. The **Connect Wallet Button** will prompt the user to connect their Metamask wallets with the application. The **Campaign Cards** on the **Landing Page** will contain some of the basic information about the campaign, such as title, thumbnail, target, current funds raised, time left to deadline. When the wallet has been connected, the

**Connect Wallet Button** will become a **Create Campaign Button**. The **Navigation Bar** will contain links to view all campaigns (**Landing Page**), View current user's campaigns (**Landing Page** with only current user created campaigns), and **Create Campaign Page**.

The **Create Campaign Page**, also accessible with the **Create Campaign Button**, is a form which takes all the inputs required to create a campaign in the contract. After filling in the details the user should hit the **Submit Button** which will initiate a Metamask transaction after which the campaign will be created.

When a **Campaign Card** is clicked, it will lead to the **Campaign Details Page** which will contain all the information about the campaign such as, title, thumbnail, target, amount collected, deadline, donators, and the donated amounts. This page will also contain the **Fund Button** which will initiate a transaction on Metamask to the campaign. The transacted amount should be input by the user in the blank given.

The aim of UI design is to create a visually appealing and user-friendly interface that enables users to easily navigate and interact with the product. As mentioned before a crowdfunding platform could be used for important causes so making it appealing and friendly increases the chances of donations and campaigns being created which brings the chance for positive change.

The colour pallet has an important role in the user experience. Colour psychology is an important consideration when building a UI, as much as it is in the logo. Below is the base colour pallet used for the components of EthRaiser.



- White (ffffff) Grey (545454) and Black (000000) are standard colors used.
- The Ethereum Blue (a2bad2) is a touch to symbolize the usage of the Ethereum blockchain in this application.
- The Green, and super dark shade of Green (008f4a, 1e3d35) are to symbolize growth and increment.

Below is the Lo-Fi UI Design of the web application.

Landing Page:



Create Campaign Page:





**Campaign Details Page:**

## FRONT-END DEVELOPMENT

The ThirdWeb application template is created within the 'Client' directory. The application template is created with the Vite framework (selected with prompts), and JavaScript as the language. Along with the configuration and dependencies that come with the application template, there is also a react router dom dependency installed. 'npm run dev' will run the development script from Vite, which compiles the application and hosts on a localhost port. Visiting this port on the browser will allow the developer to test the application.

The existing source folder is replaced with the developer's own 'src' folder. The 'src' folder contains the assets, components, constants, context, pages, styles, utils folders; App.jsx, index.css, main.jsx, and index.jsx files.

The 'assets' folder contains all the required elements that are placed into the UI as themselves such as logos, and icons.

The 'components' folder will contain all the custom-made UI component templates built by the developers, they are styled and customized as needed based on where it is being used in the application. Components included are CustomButton.jsx, FormField.jsx, and CountBox.jsx. The CustomButton.jsx will be used for the create campaign button from the landing page, as well as the submit campaign button from the create campaign page. The form fields are used in the Form in the create campaign page. The count box is used in the campaign details page.

The 'constants' folder, contains an index.js file that has the components of the navigation bar. It is responsible to import the assets needed as icons, linking them to a page, and naming the navigation component.

The 'context' folder will contain the interfacing between the contract and the web application. All of that code is contained in the 'index.jsx' folder within. The code imports the contract, creates functions that interface with the functions of the contract using hooks such as useContractWrite(), useAddress,



useMetamask, which are provided by ThirdWeb. It then creates more functions that include more logic including the functions that interface with the contract. Suppose a function called publishCampaign is made which initiates the createCampaign function (interfaced from web application to contract) corresponds the inputs of the campaign creation form set as variables. Similarly, all the other functions of the contract are interfaced as well and provided with the necessary input and logic the functions require to run in the contract.

The 'pages' folder contains the user interfaces, and logic for the three pages: CampaignDetails page, CreateCampaign page, home (landing) page, view my campaigns (profile) page. The pages will use all the logic and UI components, assets, constants, and utils (described later).

The styles folder consists of the pre-defined styles that comes from TailWindCSS, which will be used in building the pages and UI components.

The 'utils' folder, contains of a single file, which has the logic and calculations for days remaining till deadline, a calculation for how much the campaign progress bar should be filled, verifying thumbnail validity.

Under the main 'Client' directory there exists scripts 'App.jsx', 'main.jsx'. The App.jsx file consists of the navigation bar, side bar, and view of all campaigns. There are routes specified in the navigation bar, route to the URL (of page) from the component. To summarize, the 'App.jsx' file is to set up the routes for the components to their destination URL. The 'main.jsx' file is where the root element for the application is created, application is set to Goerli chain, wrap router component around 'stateProvider' and 'App' component, and render the web application to the DOM within the root element created earlier.

---

## INCENTIVIZATION PROPOSAL

As mentioned earlier, a crypto currency can be created by anyone and liquidated against Ether or be minted as a governance token for an application/platform. To incentivize the users of the crowdfunding platform, a governing token can be minted, with unlimited supply. The token could be made to have functions, such as act as a price to be paid to be able to create a campaign on the platform. As the token has a value of function, it would be a desirable asset. The value of the asset can fluctuate based on the supply & demand of the asset. This will create a decent leverage for the crowdfunding platform, provided the platform has marketed itself well. The leverage is as so, as the popularity of the platform increases, the token price gets higher, thus creating more popularity for the platform which makes it more effective for the campaign owners. If the popularity or demand of the platform decreases, the price of the token decreases, which might make the platform less effective however, makes it less expensive to use for the campaign owners.

To keep the prices of the token fluctuating healthily, there can be several mechanisms used.

Token burning can be used to burn a percentage of the tokens thus removing it from the circulating supply. As supply decreases, demand increases while driving the price upwards.

Types of Token Staking can be implemented such as staking tokens for the probability of success of the campaign funding goal, staking the amount as a loan to the campaign (debt-based implementation), amongst other rewards.

Airdrop mechanisms can be implemented, such as giving an individual or group of individual users a number of tokens as reward for their usage of the platform. This mechanism can also be used in conjunction with Token Staking.

The crowdfunding platform can be integrated or partnered with other platforms where the token can gain value by having access to features on the other platforms.

The token can be circulated, exchanged, and traded further increasing its popularity and potential as a financial asset or investment.

Such techniques of tokenomics can greatly increase the awareness, popularity, and demand of the crowdfunding platform. This will encourage more users onto the platform which helps bring in more productivity and innovation. It also solves various limitations in web3 crowdfunding platforms and traditional crowdfunding platforms.

## TESTING

This section will include descriptions on two of all the tests conducted during the development of EthRaiser. The two tests are the final contract functional testing, final front-end functional testing.

### CONTRACT TESTING

#### EXPECTED RESULTS

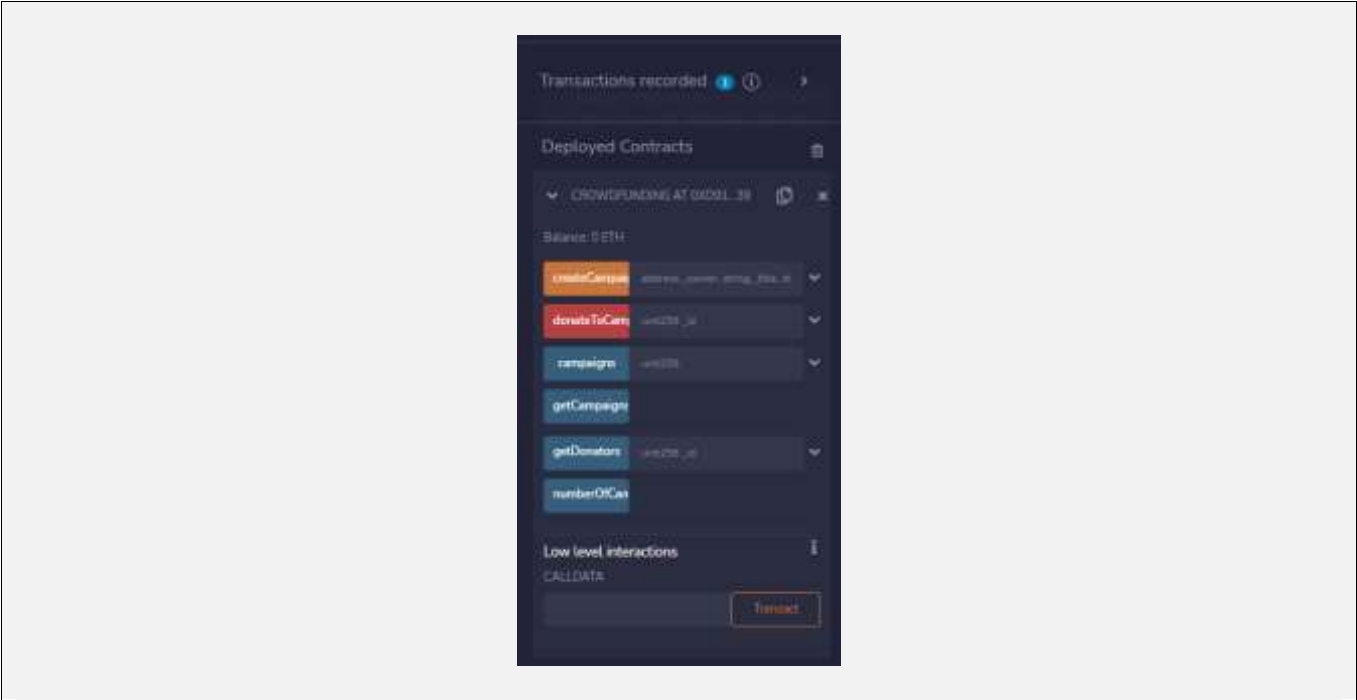
- Validation and verification of smart contract to be free of errors.
- Deployment of Smart Contract to Goerli chain.
- Create campaign function working.
- Donate to campaign function working.
- Get all campaigns function working.
- Get all donators and donations function working.

#### TEST SETUP

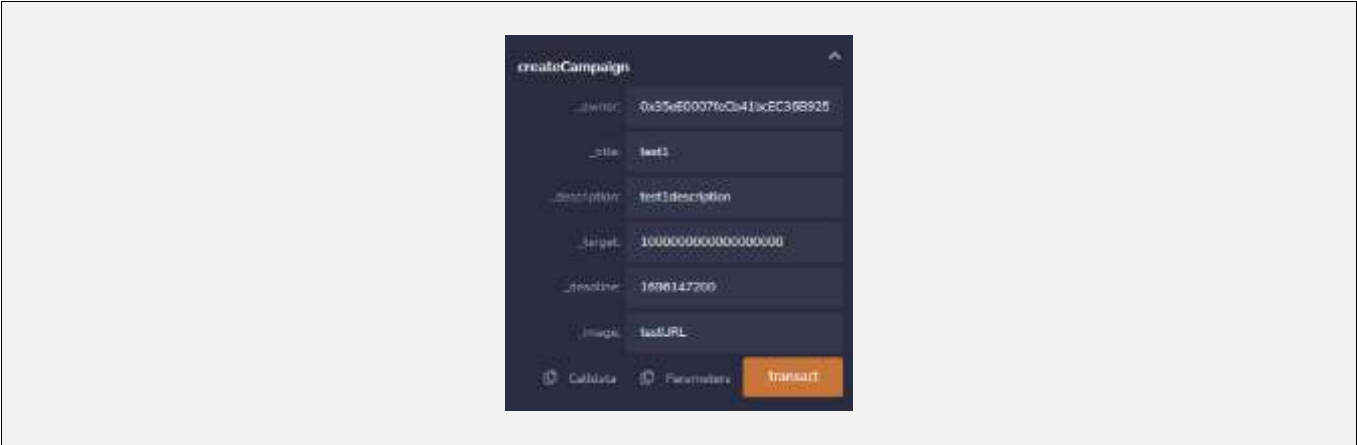
- [Remix IDE](#) is used for testing as it allows for easy interfacing with the contract functions. Remix IDE can alternatively be used in conjunction with Metamask to upload the contract to the Goerli chain.
- Within the IDE a test file is created in the IDE within the 'contracts' directory, to which the contract code is added.
- After compilation, in deployments, environment set to 'Injected Provider-Metamask'
- Metamask Wallet connected to Remix IDE.
- Smart contract deployed.

#### PROCEDURE

Once the contract is deployed, the functions and variables in the contract will be shown as seen below:



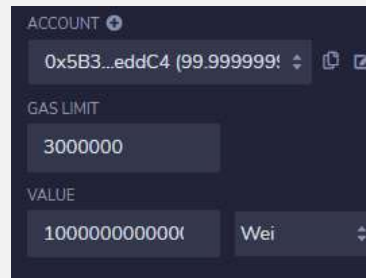
- To create a campaign, the details are added, and the 'createCampaign' button is pressed. The target amount is input in Wei (1 Ether = 100000000000000000 Wei), and deadline is in Unix Epoch time (October 1, 2023, 12 PM, GMT+4 = 1696147200).



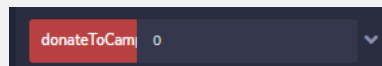
- The campaign was created with id 0.
- The get all campaigns function was tested to see the existence of the campaign created.



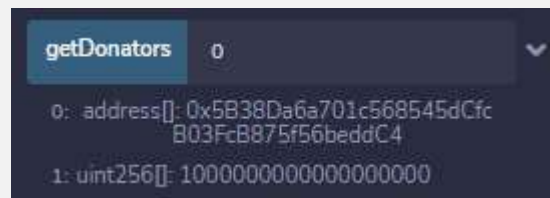
- Transaction amount is set as below.



- Value is sent to campaign ID 0.

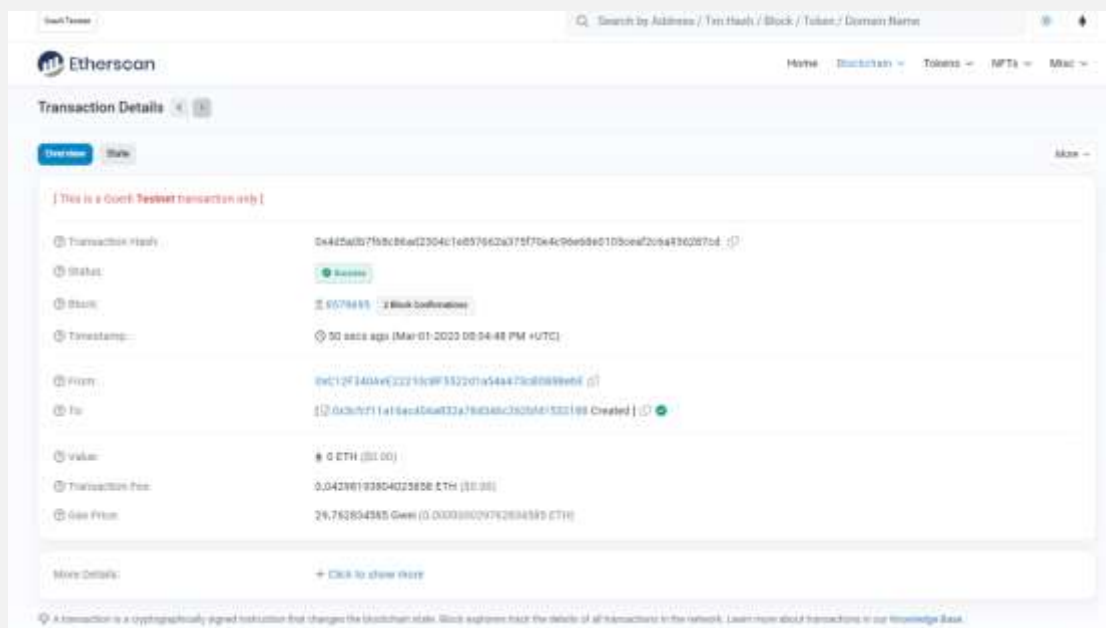


- The result of donation and functioning of the get donors and donations is tested by giving the function a campaign ID of 0.



## RESULTS

- No errors in smart contract code.
- Contract successfully deployed on Goerli chain.



- All functions of the smart contract are working as expected.
- Contract is ready to be used for development of web application.

## ETHRAISER FINAL TESTING

### EXPECTED RESULTS

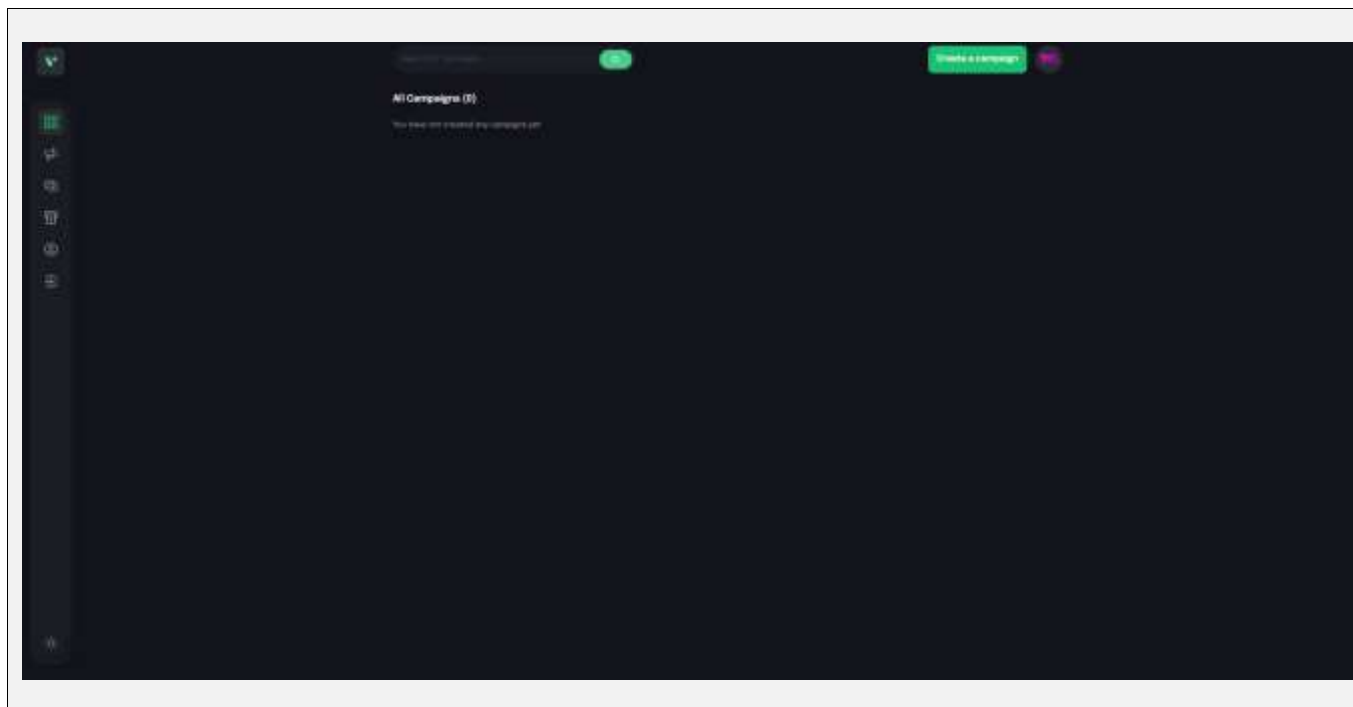
- Responsiveness
- Home page working (viewing campaigns)
- Create campaign working.
- Donate to campaign working.
- Campaign details page (viewing campaign before and after donation)
- Able to view backers.

### TEST SETUP

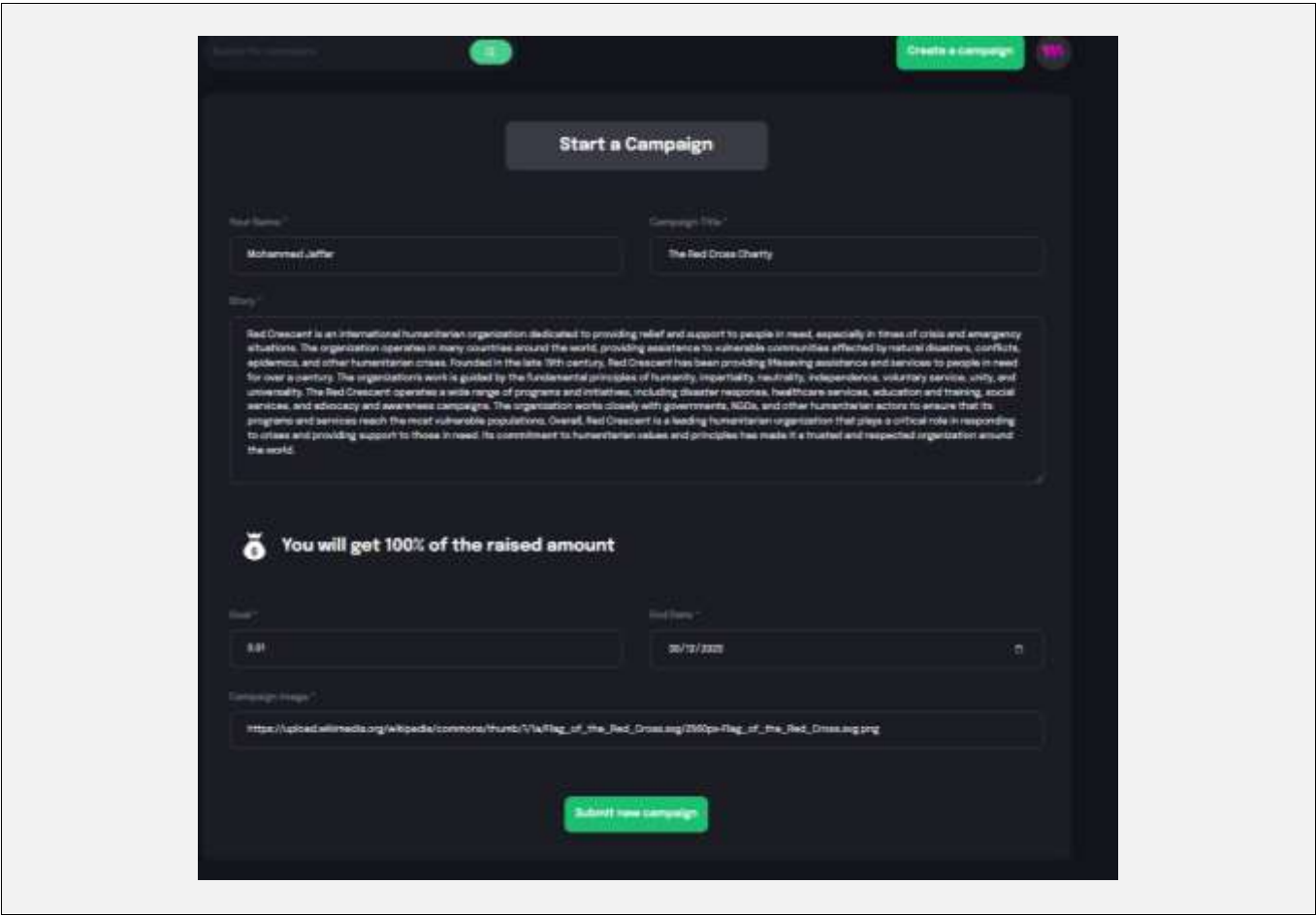
- Vite 'dev' script is run after which the web application is hosted on 'http://127.0.0.1:5174/'.
- The IP is put into the browser.
- Wallet connected to application.

### PROCEDURE

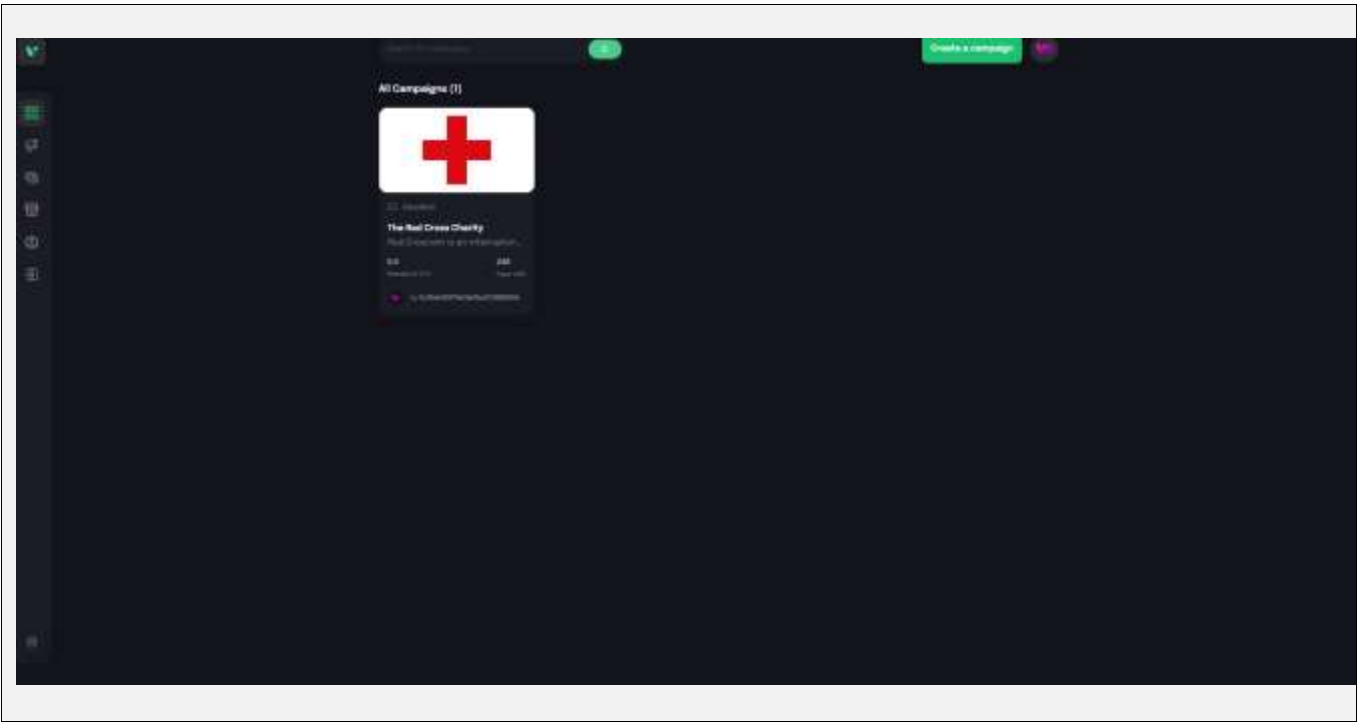
- Below is the landing page.



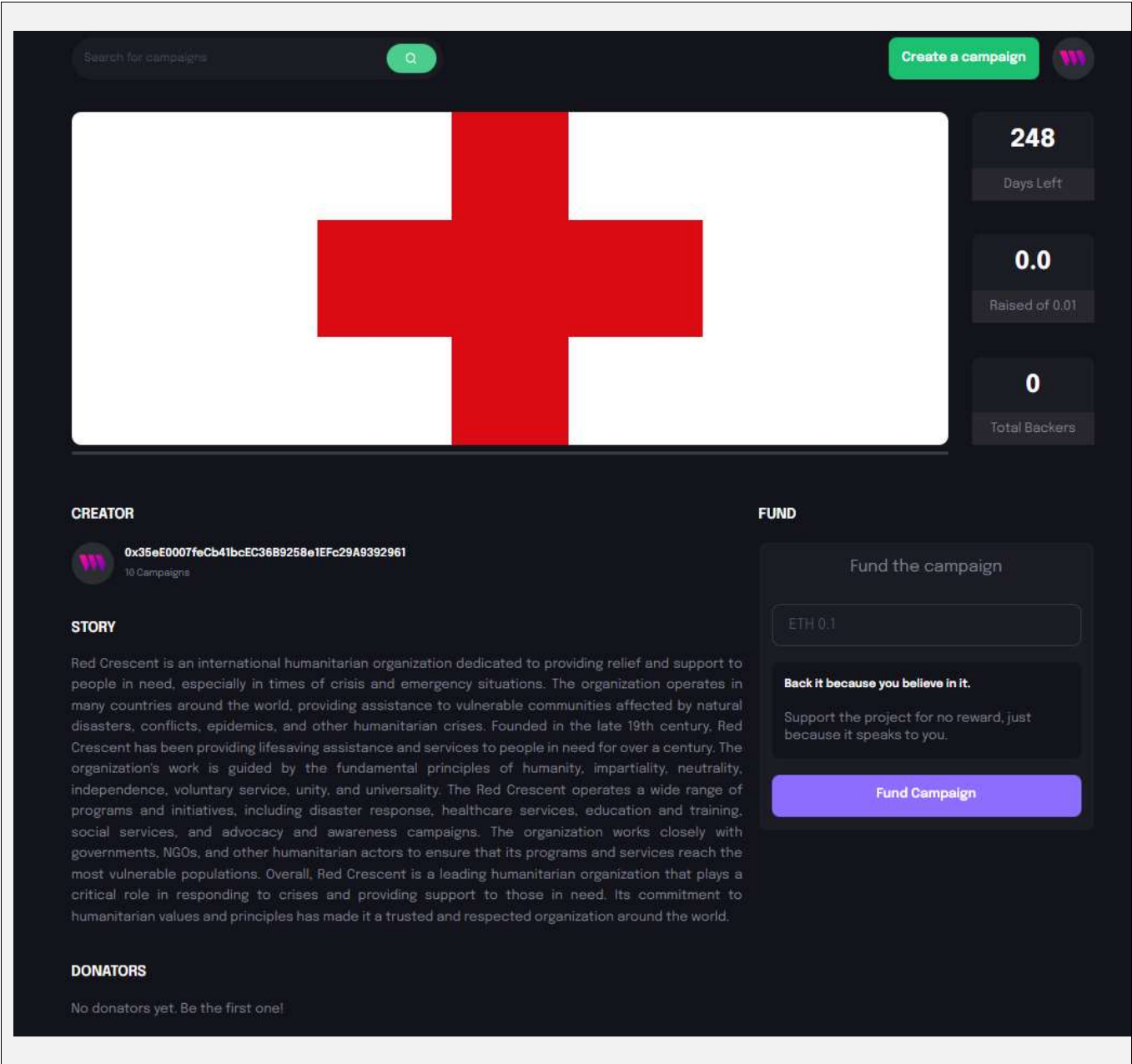
- A campaign called 'The Red Cross Charity' will be created as a test campaign, by clicking the 'Create Campaign' button and filling the form.



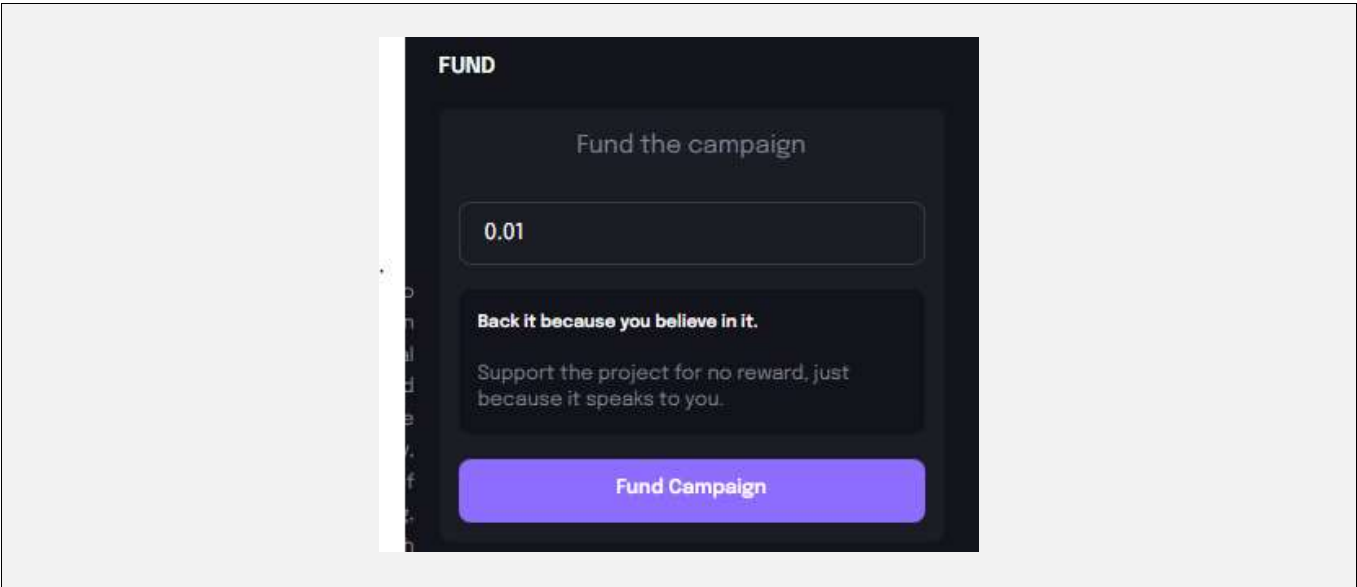
- The campaign is now visible on the front page.



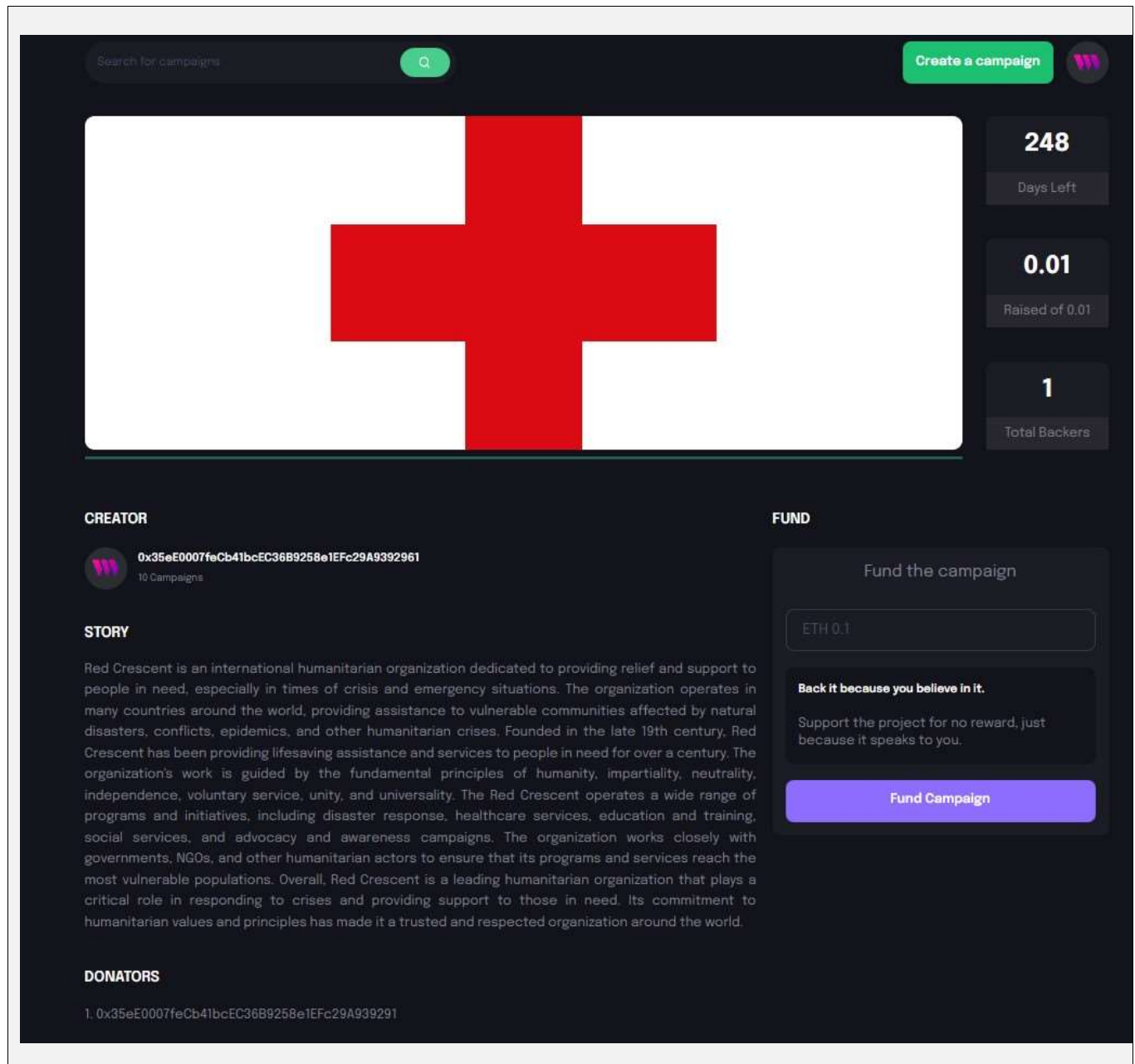
- The campaign details page is accessed by clicking the campaign card on the home/landing page.



- 0.01 GoETH is donated to the campaign.



- After donation the campaign details page is viewed again.



## RESULTS

- A responsive front-end.
- Campaign 'The Red Cross Charity' was created successfully
- Functions for viewing campaign and campaign details page working successfully.
- Donate to campaign function working successfully as 0.01 GoETH donation has been made to the campaign.

## DISCUSSION

## RELATED WORKS

### KICKSTARTER

Kickstarter is a donation-based and reward-based crowdfunding platform. Kickstarter is based in America and takes payments through online payments. The reward-based model is dominant in the platform. There



are many different categories given on the website on which anyone can sign up, and create their campaign.

---

## IN CONTRAST TO ETHRAISER

- Web2, requires users to make an account and collects user information.
- Takes online payments in fiat currency.
- Follows a deposit pattern, if campaign does not reach funding goal before deadline, the funds are directed back to the backers.
- The deposit is directed to the campaign owner only if the funding goal is reached before the deadline.
- 5% fee from the funds collected along with payment processing fees.
- Has separate categories.
- Comprises of a large employee base and ecosystem.

---

## STARTJOIN

StartJOIN is a web3 crowdfunding platform launched in 2014. It also is designed like EthRaiser to provide an alternative to traditional crowdfunding systems.

---

## IN CONTRAST TO ETHRAISER

- Supports a range of cryptocurrencies such as bitcoin, Litecoin, and dogecoin.
- Provides various project management tools.
- Provides useful analytics and helps creators understand how their campaigns are performing.
- It connects to social media platforms to make sharing campaigns easier.
- Like EthRaiser, the campaign creators and backers are only charged the gas fee.

StartJOIN and Kickstarter have been the main two inspirations for EthRaiser's developer. The ambition of EthRaiser in this light is to pick the best of both worlds and integrate as many useful features as possible. There are many mechanisms and aspects that can help solve the problem definition in a well-informed manner. Some of which will be included in the Future Developments section.

## LIMITATIONS OF ETHRAISER

Here the limitations of blockchain technology will be summarised along with EthRaiser specific limitations which are limitations of EthRaiser particularly.

- **Ethical limitation:** On one hand where automation of various processes benefits each stakeholder in terms of efficiency, costs, and time. The benefits of the increased probability in the campaigns. On the other hand, the loss of jobs and livelihoods can potentially cause financial setbacks to a large group of people. Especially when most of these people have worked in the same field their entire careers. This can also be taken as an advantage as it could lead to creation of newer careers and jobs, however, it is uncertain. EthRaiser's move towards the blockchain agenda is aware of this.
- **Energy Usage:** The PoW mechanism, consumes high amount of energy and computational power. If Bitcoin was a country and its energy consumption was metered, it would consume more than 5,500 TW of energy, which is 0.6% of the world's energy consumption. EthRaiser is aware of the fact and pushes towards supporting the PoS consensus mechanism, however, the blockchain agenda cannot ensure all of its users move towards PoS.
- **Security:** Although arbitrary, it is not impossible for a 51% attack even on the PoS consensus mechanism, and irreversible hacks. Smart contract invulnerability also comes into light here. As

EthRaiser was developed by a single developer and a limited budget, the code hasn't been entirely audited and verified.

- **Regulation:** EthRaiser is not yet regulated by any financial authority or government, this includes lack of licensing. This could be a bottleneck to investors particular about the regulatory aspect and investor protection.
- **Authority:** Since EthRaiser is an open application currently, it could lead to overcrowding, adding to this is the lack of categories in the campaign views. This could lead to difficulty traversing and finding interesting campaigns to users.
- **Lack of features:** Currently EthRaiser only operates on the donation-based model which is simply put, funds flowing back & forth, and providing visibility to campaigns.
- **Limited Flexibility:** As EthRaiser only accepts donations in Ether, it could be a limiting factor for those unwilling to convert their cryptocurrency usually due to transaction charges or time constraint.

## FUTURE IMPROVEMENTS & DEVELOPMENTS

EthRaiser is currently working on ways to increase visibility and improve user retention. Various techniques such as governing cryptocurrency mechanism, development of campaign categories, and a stand-alone mobile application for IOS and android are being looked at. The current techniques in scopes are described about below:

- **Governance Cryptocurrency:** EthRaiser plans to mint governing tokens, RaiseCoin (RAZE). RAZE will be used to access various (coming soon) features of EthRaiser, which will be described below.
- **Campaign Categorisation:** Campaigns on the website will be categorised firstly, based on the crowdfunding model, then based on the subject of the campaign.
- **Staking:** EthRaiser plans to bring a staking feature which will extend EthRaiser to Debt-based crowdfunding. Debt-based crowdfunding can be rewarding to users, which brings more incentive and popularity to EthRaiser.
- **Equity-based:** A feature where users can back a campaign in return of equity of the company or business hosting the campaign. Since smart contracts will be used it would be easier to transfer the asset tokens. This feature would require EthRaiser to be able to accept other tokens on the platform as its principal is for the campaign owner, creating their own token which would be recognized as the equity of the company.
- **NFT Auction:** EthRaiser plans to create a mini marketplace where, NFTs (Non-Fungible Tokens), which could be used by campaign owners and users. When campaigns receive NFT donations, the campaign owners can decide to auction it off within the website itself, or hold it to auction on any other platform such as OpenSea.io
- **NFT Airdrops:** As NFT support will be added when NFT auction feature is added, campaign owners can mint or select existing NFTs as rewards either in form of an airdrop or a sure reward for a backer. This extends EthRaiser further into reward-based crowdfunding.
- **UI Design:** EthRaiser is fond of its dark and subtle website design, however, to please more users, a light mode for the website is under development. A new page with a donator/backer leader board will be added where donators can choose to list them self, based on the amount they have donated and to which. This could attract users who want a certain social status.
- **RAZE Airdrops:** Select users or campaign owners and token owners will be chosen at random to receive an airdrop of minted token supply.

## CONCLUSION

### REFLECTION

#### FINDINGS AND CONTRIBUTIONS

Project EthRaiser had provided a push towards to blockchain agenda, raising awareness of the technology and the benefits of it. The technical report of EthRaiser focused on the background of the technology in detail, provided a review of literatures in the topic. Views are provided both, for and against blockchain technology. Analysis was done to understand how traditional methods of crowdfunding are affected by the problem definition. The stakeholders were shed light on. The need for a web3 crowdfunding platform was assessed after which EthRaiser was born. The planning, development, and testing of EthRaiser was described in detail. Providing current limitations, future developments, and improvements to come.

#### DIFFICULTIES FACED

The development of EthRaiser was not a uniform journey, mainly since the goerli testnet was going to be deprecated, the developer was forced to move to Sepolia. However, moving the project entirely built for goerli was a slight challenge. Amongst other challenges as mentioned below.

- Several instruments such as TailWindCSS and ThirdWeb were updated during the development of the project resulting in reiterations, and updates being made to keep EthRaiser running.
- Tools that were used in making the logos, icons, and SVGs were paid, hence, adding a cost to the developer's budget.
- Solidity language is not quite widely adopted yet, so it is hard to find proper training in the writing of the language.
- Developer had to deal with unclear documentations for the development of the context for the front-end web application, the toughest of which was getting to import the contract, resolve contract metadata, and interface with contract functions.
- At the start, goerli Test ETH token faucets required the developer to run a Proof-of-Work node due to high demand, which led to keeping the computer on for a long time and energy consumption.

#### AUTHOR'S CONCLUSION

The project EthRaiser began with the interest of promoting the agenda of blockchain, adoption and implementation of blockchain technology, and transformed in the eyes of EthRaiser's developer into a much more significant and impactful journey.

EthRaiser pushes a technology that could positively impact countless lives, make the best dreams of the numerous stakeholders come true. A powerful technology showing humankind how humankind itself can engineer itself away from their problems. A technology that can stop potential frauds and take away unfair power from large singular entities. A technology bringing efficiency and automation into play so that humankind can focus on what's really important. A solution so that humankind can move to solving a new problem.

Looking far ahead, EthRaiser's developer is filled with visions for EthRaiser's bright future. A moment to witness, a better future for all.

## APPENDIX

### 1.1 Hashing

Hashing is known as mapping the given information to an alternate format after which the original information is now encrypted. The encrypted information can only be decrypted with a key if included in the hashing algorithm. There are various methods of hashing and pre-made algorithms such as SHA-256. SHA-256 hashing is used in Blockchain. SHA stands for Secure Hash Algorithm which produces a 256-bit hash out of the given information. As the hashing function is the same, the output for the same given information will also be the same, whereas if there is even little change between two given informations, the output could be vastly different.

### 1.2 Double Spending

Ledgers are distributed. Double Spending can be understood through this scenario. Imagine an entity A with a balance of 5 dollars in his/her account, and currently this data is in everyone's ledger. If entity A decides to send his/her 5 dollars to an entity B, the transaction needs to be accounted for in everyone's ledger. Failure to do so will result in entity A still having 5 dollars in other ledgers which he/she can again spend.

### 1.3 Gas Fees & Gas Limit

Gas in Ethereum is defined as the amount of computation power, energy and network resources that was required in order to validate a transaction on the blockchain. A Gas Fee is the fees associated with the amount of computational power or energy that was expended that is faced by the user which initiates a transaction. A Gas Limit is the limit of Gas Fee the user is committed to spend to make their transaction go through. Users can vary their Gas Limit in order to make their transaction go through faster. Gas Fees keep fluctuating based on Supply & Demand (Appendix 1.6) of the computational power, energy and network resources. Gas is the cryptocurrency Ether. Gas Fees is usually a small amount so it is calculated in Wei or Gwei which are the smallest units of Ethereum.

$$1 \text{ Wei} = 10^{-18} \text{ Ether}$$

$$\text{GWei} = 10^{-9} \text{ Ether}$$

### 1.4 Hash Value

A hash value is a randomly generated hash which is unique for each node in the network. Each node has its own hash value.

### 1.5 Supply & Demand

The value of most commodities fluctuates on the basis of Supply & Demand. When demand increases or supply decreases the price of the commodity increases. When demand decreases or supply increases, prices go down. Supply & Demand are inversely proportional to each other, as in if supply increases demand decreases and vice versa. The price of Ether follows the same concept.

### 1.6 Minting

Minting in other words is the issuance of cryptocurrency. Similar to how cash is printed in real life. Ether is minted when through validator/miner nodes as incentives. When other cryptocurrencies are created on top of the Ethereum blockchain, they are minted against the value of Ether. For example, if a company wants a cryptocurrency of their own and they want it to have some value they should mint their token

amount against a value of Ether. So, if they mint 100 tokens against 1 Ether's price (suppose 1000 dollars), each token is now worth 0.01 ether which means 10 dollars.

### **1.7 Whale Movement**

In the cryptocurrency world, users that play with more than a million dollars worth of cryptocurrency are known as whales. Since Ethereum is a transparent network, it is easier to track such transactions. These transactions are known as Whale Movement. The movements and transactions of Whales create impact in the prices of the cryptocurrency as they have significant ability to increase or decrease supply or demand. Due to this, many traders around the world track Whale Movements and make investments similar to Whales in order to gain profit or minimize loss while trading.

### **1.8 Statically Typed Programming Language**

A programming language where data types have to be declared before compilation and runtime. The data types do not and can not change during run time. Any attempt to do so will cast an error.

### **1.9 Object Oriented Programming**

A programming paradigm based on the creation and usage of objects and classes. Objects can be defined as the developer's own data type which has its code data and functions.

### **1.10 Private-Key and Public-Key**

The Private-Key only known to the owner of the wallet, is for signing transactions. The Public-Key is generated from the Private-Key and it can be used by other entities to ensure the transaction has come from the sender. This is done by providing the transaction the public-key of the sender.

### **1.11 Equity**

When a company decides to take itself public or offer shares of the company to an audience or investors, those shares, or the amount owned of the company by the entity is called equity. For example, entity A owns 25% equity of company B. The price of the company shares can fluctuate based on the company's success and supply & demand.

### **1.12 Fiat**

Fiat currency in terms of money is known as currency which is declared a legal tender by the government or controlling body and isn't backed against a tangible asset such as gold or silver so it has no fixed value. Examples of Fiat currency are USD (United States Dollar), INR (Indian Rupee), and AED (Emirati Dirham).

### **1.13 Edge Computing**

Edge computing refers to the model where processing in the network takes place at the 'edge' of the network. For Example, A group of vital sensors at a remotely located gas plant give out data which needs to be processed and analysed in order to control the gas plant preventing any failures or danger. In such an application there is very little room for failure. Reasons for failure could be network outages, latency, transmission errors, etc. So such an application would require the processing power to be close to the remote located gas plant which logically becomes now the edge of the network hence the name Edge Computing.

### **1.14 Nonce**

A nonce is a value which is incremented with every transaction initiated from a particular Ethereum Account. Nonces are used to prevent duplicate transactions occurring. This will ensure no suspicious actor can send a transaction once more even if it was validated before. Whenever a transaction is sent

from a particular account, Ethereum checks if the nonce of the current transaction is higher than the nonce of the previous transaction from that particular account.

### 3.1 DOM

The Document Object Model (DOM) is a cross-platform interface that can be used with various programming languages and platforms. It is designed to assist developers in interacting with HTML or XML documents as if they were tree structures, with each component of the document represented as an object or node. This logical tree enables developers to access and manipulate various parts of the document in a structured manner, making dynamic and interactive web applications easier to build. As a result, the DOM is a must-have tool for web developers who want to build robust web applications that can run on multiple platforms and be accessed by users all over the world.

## REFERENCES

1. Sornn-Friese, H. (2018). *Initial Coin Offerings and Smart Contracts: A Regulatory Overview*. Journal of Financial Regulation and Compliance, 26(3), 373-385.
2. Liu, X., Wang, C., & Zhu, Z. (2018). *Blockchain-Based Decentralized Crowdfunding: A Decentralized Automated Mechanism for Transferring Funds*. IEEE Access, 6, 13059-13069.
3. Li, H., Zhang, J., & Huang, S. (2021). *A Survey of Crowdfunding in Web3.0 Era: Architecture, Applications and Challenges*. IEEE Access, 9, 59889-59905.
4. Böhme, R., & Christin, N. (2018). *Blockchain and the Future of Crowdfund*. Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems (CryBlock 2018), Munich, Germany, June 25, 2018, 1-6.
5. Mao, S. (2021). *Ethical and social implications of blockchain technology in the crowdfunding industry*. Journal of Business Research, 133, 643-652.
6. Pilkington, M. (2019). *An exploration of blockchain-based crowdfunding: Is the technology a viable option for the future of crowdfunding?* Journal of Business Research, 98, 365-380.
7. Liu, Y., Li, X., & Liang, X. (2020). *Equity crowdfunding on blockchain: A comprehensive review and future research agenda*. Journal of Business Research, 110, 229-240.
8. Chen, H., & Zhang, C. (2021). *The emergence of digital debt-based crowdfunding and its potential implications for traditional banks*. Journal of Banking & Finance, 124, 106118.
9. Kokkinaki, A. and Papadopoulos, A., 2019. *Blockchain Technology Applications in Real Estate Crowdfunding*. Journal of Real Estate Literature, 27(2), pp.197-222.
10. Böhme, R., Christin, N., Edelman, B., & Moore, T. (2015). *Bitcoin: Economics, technology, and governance*. Journal of Economic Perspectives, 29(2), 213-238.
11. Swan, M. (2015). *Blockchain: blueprint for a new economy*. O'Reilly Media, Inc.
12. Buterin, V. (2014). *A next-generation smart contract and decentralized application platform*. Ethereum White Paper.
13. Gravino, C., Guagliardo, I., Milani, A., & Zanero, S. (2021). *A systematic review of smart contract vulnerabilities on Ethereum*. Computers & Security, 107, 102215.
14. Grossman, H., & Reitblat, E. (2018). *Crowdfunding on the blockchain: An introduction to decentralized crowdfunding*. Journal of Alternative Investments, 20(2), 24-34.
15. Hsieh, J. J., Liu, C., & Fang, S. C. (2021). *Blockchain-based crowdfunding for small and medium-sized enterprises: A review of benefits and challenges*. Sustainability, 13(5), 2795.
16. Atzei, N., Bartoletti, M., & Cimoli, T. (2018). *A survey of attacks on Ethereum smart contracts (SoK)*. In Proceedings of the 6th International Conference on Principles of Security and Trust (POST 2017) (pp. 164-186). Springer.
17. Brody, E. (2021). *DeFi has a usability problem. Here's what's being done to solve it*. Coindesk. Retrieved from <https://www.coindesk.com/defi-has-a-usability-problem-heres-whats-being-done-to-solve-it>



18. Nebbia, P. (2021). *DeFi and the regulatory challenges ahead*. Coin Center. Retrieved from <https://www.coincenter.org/defi-and-the-regulatory-challenges-ahead/>
19. Schär, F. (2021). *DeFi and the future of finance*. European Central Bank. Retrieved from <https://www.ecb.europa.eu/press/key/date/2021/html/ecb.sp210421~7566bd8b62.en.html>
20. Zohar, A., & Eyal, I. (2015). *Bitcoin: Under the hood*. Communications of the ACM, 58(9), 104-113.
21. Bhargavan, K., Delignat-Lavaud, A., Fournet, C., Gollamudi, A., Gonthier, G., Kobeissi, N., ... & Protzenko, J. (2016, October). *Formal verification of smart contracts: Short paper*. In Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security (pp. 91-96).
22. Baron, N. (2021). *DeFi flash loan attacks are rising, but can anything be done?*. Decrypt. Retrieved from <https://decrypt.co/76672/defi-flash-loan-attacks-are-rising-but-can-anything-be-done>
23. Chohan, U. W. (2021). *DeFi yield farming: Risks, rewards, and regulation*. Journal of Risk and Financial Management, 14(5), 202.
24. Casey, M. J., & Vigna, P. (2018). *The truth machine: The blockchain and the future of everything*. St. Martin's Press.
25. Makoto, S., & Oshiro, Y. (2021). *Towards decentralized stablecoins*. arXiv preprint arXiv:2101.01541.
26. Deloitte. (2018). *Smart contracts: A tool for speeding up business processes*. Retrieved from <https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/Innovation/deloitte-uk-blockchain-smart-contracts.pdf>
27. Wood, G. (2014). *Ethereum: A secure decentralised generalised transaction ledger*. Ethereum Project.
28. Xu, Y., Liu, L., & Du, X. (2020). *A systematic survey of smart contract development tools*. Journal of Systems and Software, 171, 110791.
29. Zhang, J., & Liu, X. (2018). *Challenges and opportunities of blockchain smart contracts*. In 2018 IEEE International Conference on Big Data (Big Data) (pp. 227-232). IEEE.
30. Hou, S., Zhang, X., Li, J., & Shi, W. (2018). *Smart contract-based financial markets: A framework for future design*. IEEE Communications Magazine, 56(10), 160-165.
31. Pilkington, M. (2016). *Blockchain technology: Principles and applications*. Research Handbook on Digital Transformations, 225-253.
32. Zheng, Z., Xie, S., Dai, H. N., Chen, W., & Wang, H. (2017). *An overview of blockchain technology: Architecture, consensus, and future trends*. In IEEE International Congress on Big Data (pp. 557-564). IEEE.
33. Crosby, M., Pattanayak, P., Verma, S., & Kalyanaraman, V. (2016). *Blockchain technology: Beyond bitcoin*. Applied Innovation, 2(6-10), 71-81.
34. Beck, R., & Müller-Bloch, C. (2017). *Blockchain as radical innovation: A framework for engaging with distributed ledgers as Incumbent organization*. In Proceedings of the 50th Hawaii International Conference on System Sciences.
35. Tapscott, D., & Tapscott, A. (2016). *Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world*. Penguin.
36. Antonopoulos, A. M. (2014). *Mastering Bitcoin: Unlocking Digital Cryptocurrencies*. O'Reilly Media.
37. Antonopoulos, A. M. (2018). *Mastering Ethereum: Building Smart Contracts and DApps*. O'Reilly Media.
38. Reynolds, P. (2018). *Ethereum energy consumption index*. Digiconomist. Retrieved from <https://digiconomist.net/ethereum-energy-consumption>
39. Christensen, C., Forkmann, R., & Malone, J. (2019). *Decentralized blockchain governance*. Communications of the ACM, 62(11), 38-43.
40. Kshetri, N. (2018). *Blockchain's roles in meeting key supply chain management objectives*. International Journal of Information Management, 39, 80-89.
41. Wright, A., & De Filippi, P. (2015). *Decentralized blockchain technology and the rise of lex cryptographia*. Available at SSRN 2580664.

42. Agrawal, A., Catalini, C., & Goldfarb, A. (2015). *Crowdfunding: Motivations and deterrents for participation*. Harvard Business School Working Paper, (15-097).
43. Belleflamme, P., Lambert, T., & Schwienbacher, A. (2014). *Crowdfunding: Tapping the right crowd*. Journal of Business Venturing, 29(5), 585-609.
44. Gerber, E. M., Hui, J. S., & Kuo, P. Y. (2012). *Crowdfunding: Why people are motivated to participate*. ACM Transactions on Computer-Human Interaction (TOCHI), 20(6), 1-32.
45. Block, J. H., Colombo, M. G., Cumming, D. J., & Vismara, S. (2018). *New players in entrepreneurial finance and why they are there*. Small Business Economics, 50(2), 239-250.
46. Belleflamme, P., Lambert, T., & Schwienbacher, A. (2014). *Crowdfunding: An industrial organization perspective*. In Handbook of research on innovation and entrepreneurship (pp. 225-260). Edward Elgar Publishing.
47. Zhang, Y., Song, M., & Liu, Y. (2019). *The crowdfunding success factors: A comparative study of reward- and equity-based crowdfunding in China*. Asia Pacific Journal of Management, 36(4), 951-976.
48. Kemp, S. (2021). Digital 2021: *Global overview report*. We Are Social & Hootsuite. Retrieved from <https://wearesocial.com/digital-2021>