Charity and Donations

21z262 - Sushanth S
21z265 - V S Aswin Sailesh
22z434 - Abai Kumar I

19ZO03 - INTRODUCTION TO BLOCKCHAIN TECHNOLOGY

report submitted in partial fulfillment of the requirement for the award of degree of

BACHELOR OF ENGINEERING

Branch: COMPUTER SCIENCE AND ENGINEERING

Of Anna University

Oct 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PSG COLLEGE OF TECHNOLOGY

(Autonomous Institution)

COIMBATORE – 641 004

CONTENTS

ABSTRACT

The charity and donations sector is often undermined by a lack of transparency, accountability, and efficiency. Traditional charity systems rely heavily on intermediaries for fund distribution, increasing operational costs and reducing the percentage of donations that reach those in need. Additionally, donors often face uncertainty about how their contributions are being utilized, leading to a lack of trust in charitable organizations. Mismanagement or misuse of funds further exacerbates these issues, deterring potential donors and affecting the overall credibility of charitable institutions.

This project proposes a decentralized platform for charity and donations, leveraging the Ethereum blockchain to create a trust less, transparent, and efficient system. By utilizing smart contracts, the platform automates donation handling and fund distribution, ensuring that funds are released only when predefined conditions are met, such as achieving fundraising goals or reaching specific milestones. This removes the need for intermediaries and reduces associated operational costs, while providing real-time transparency for donors to track their contributions. Additionally, the platform uses decentralized storage (IPFS) for campaign details, further enhancing security and reliability.

The decentralized nature of the platform addresses critical challenges in the traditional charity model by ensuring transparency through immutable blockchain records, enhancing donor trust, and eliminating the risk of fund mismanagement. This report outlines the system's architecture, implementation details, results, and observations, demonstrating the potential of blockchain technology to revolutionize the charity sector by making it more transparent, accountable, and efficient.

CHAPTER 1

INTRODUCTION

1. Problem Statement:

Traditional charity systems face numerous challenges that hinder their efficiency and effectiveness. Among these challenges are a lack of transparency, high operational costs due to reliance on intermediaries, and the potential for mismanagement or misuse of funds. As a result, donors often feel insecure about how their contributions are used and whether they are making a genuine impact.

- **Lack of transparency**: Donors cannot track how funds are spent once donated, leading to a lack of trust in charities.
- **High transaction costs**: The reliance on intermediaries for fund distribution adds unnecessary costs, reducing the amount of donations reaching the end beneficiaries.
- **Mismanagement and misuse of funds**: Without proper oversight, funds may be allocated inefficiently or unethically.
- **Lack of donor trust**: These issues, coupled with insufficient accountability, discourage potential donors from contributing.

The aim of this project is to solve these issues by creating a decentralized platform using Ethereum blockchain technology. Blockchain enables trustless transactions and immutable records, allowing full transparency from donation to fund allocation. Smart contracts automate fund distribution, reducing operational costs and ensuring that funds are released only when predefined conditions are met.

CHAPTER 2

Requirement Analysis

2.1 Software Requirements:

- **Blockchain Platform**: Ethereum
- **Development Environment**: Remix IDE, MetaMask
- **Programming Languages**: Solidity for smart contract development, HTML, CSS, JavaScript for the frontend, Python for backend integration.
- **Libraries**: Web3.js for Ethereum interactions, Hard Hat for smart contract testing and deployment.
- **Testing and Deployment Tools**: Ganache (local blockchain environment).

2.2 Functional Requirements:

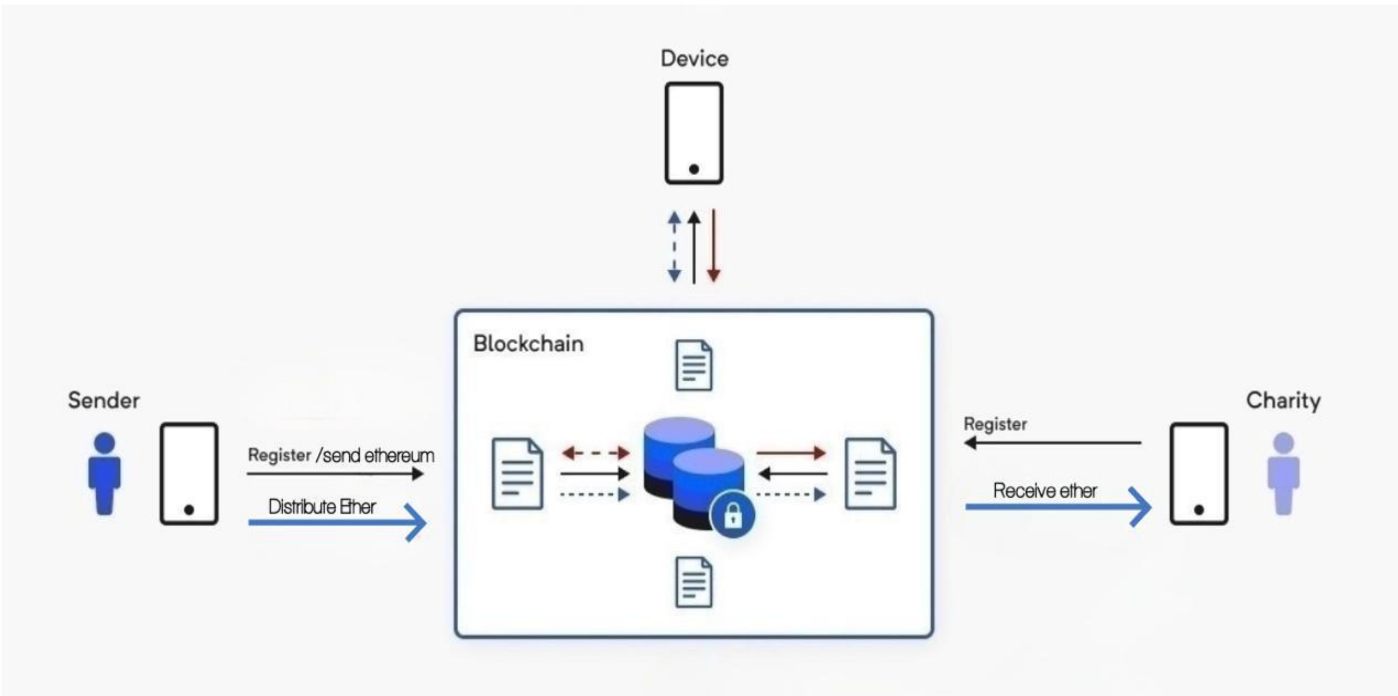
- **Donor Portal**: Donors should be able to contribute to various charity campaigns and track the flow of their funds.
- **Charity Campaign Management**: Organizations should be able to create and manage campaigns with predefined goals and conditions.
- **Smart Contracts**: Smart contracts will handle fund release based on predefined conditions, such as achieving a fundraising target or reaching a specific milestone.
- **Decentralized Record Keeping**: Transactions and fund allocation records should be stored in a decentralized and immutable manner using blockchain and IPFS.

2.3 Non-functional Requirements:

- **Security**: The platform must ensure data integrity and protection against unauthorized access.
- **Scalability**: The platform should handle multiple campaigns and donors simultaneously without performance degradation.
- **Usability**: The system should be easy to navigate for both donors and charity organizations.

CHAPTER 3

System Architecture



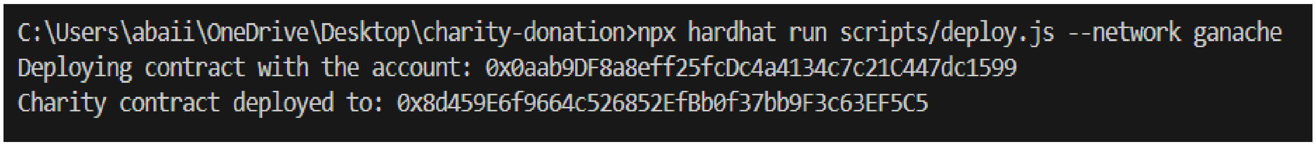
CHAPTER 4

Tools and Technologies

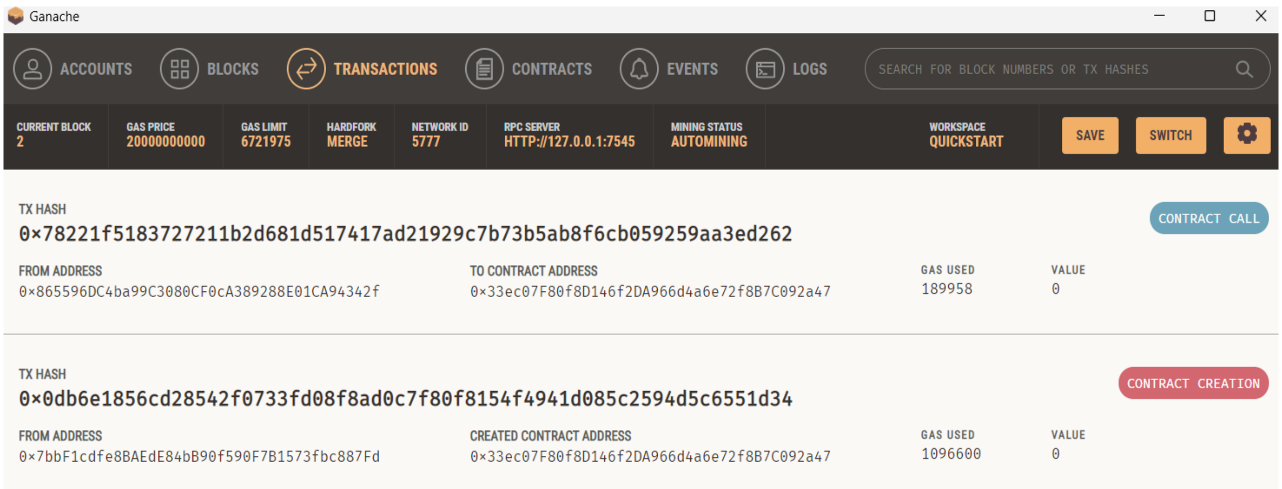
- **Ethereum Blockchain**: A decentralized platform that enables the execution of smart contracts and the creation of decentralized applications.
- **Solidity**: The primary programming language used for writing Ethereum smart contracts.
- **Remix IDE**: An online integrated development environment used for developing, deploying, and testing smart contracts on Ethereum.
- **MetaMask**: A cryptocurrency wallet and gateway to Ethereum-based DApps, integrated into the platform for donor transactions.
- **Web3.js**: A JavaScript library that enables interaction with the Ethereum blockchain from the frontend.
- **Ganache**: A personal Ethereum blockchain used for testing smart contracts locally.
- **Hard Hat**: Frameworks for testing and deploying smart contracts.

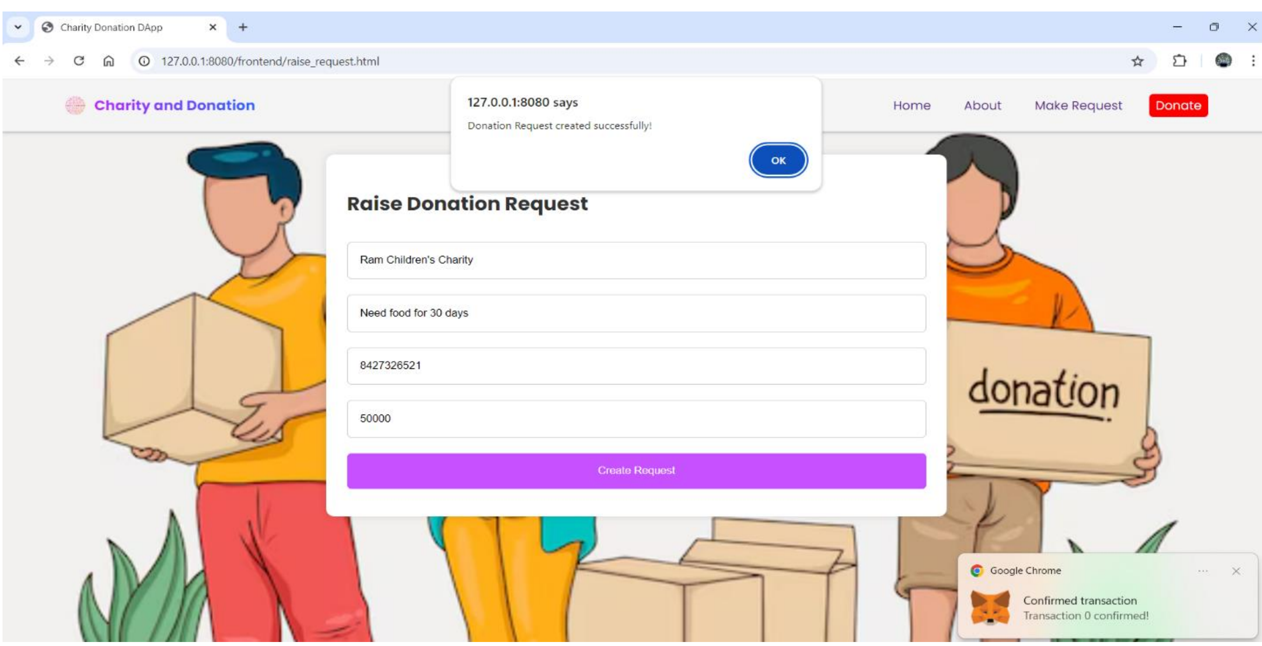
CHAPTER 5
IMPLEMENTATION

5.1 Screenshots

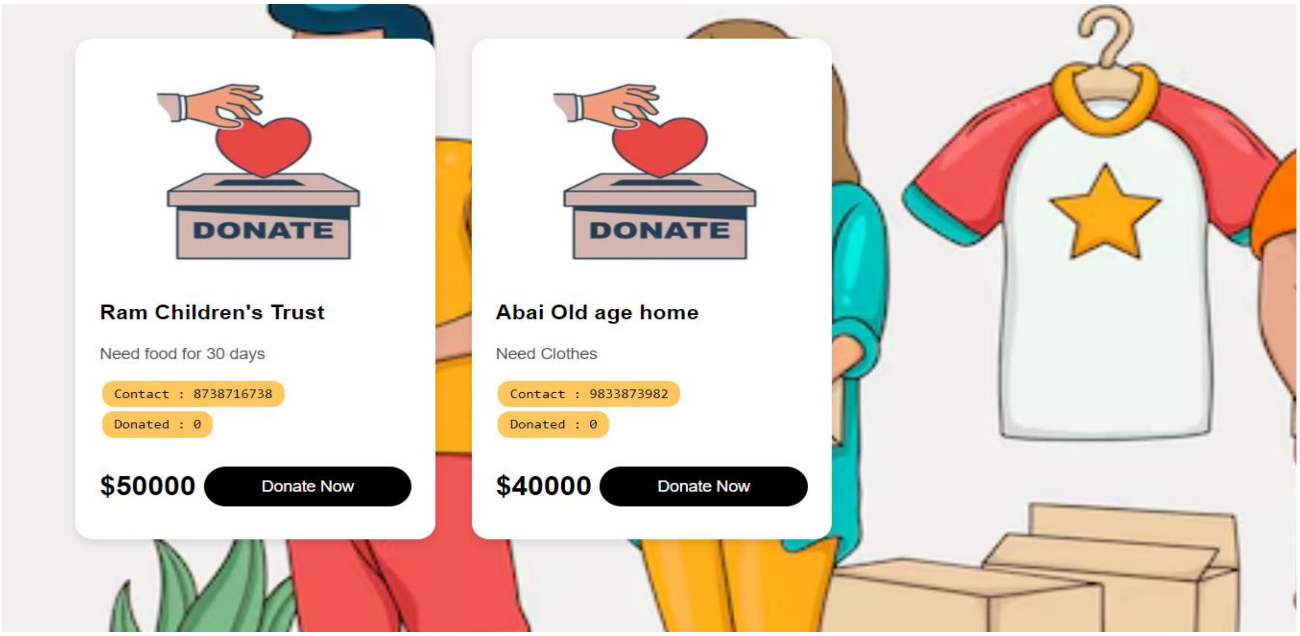


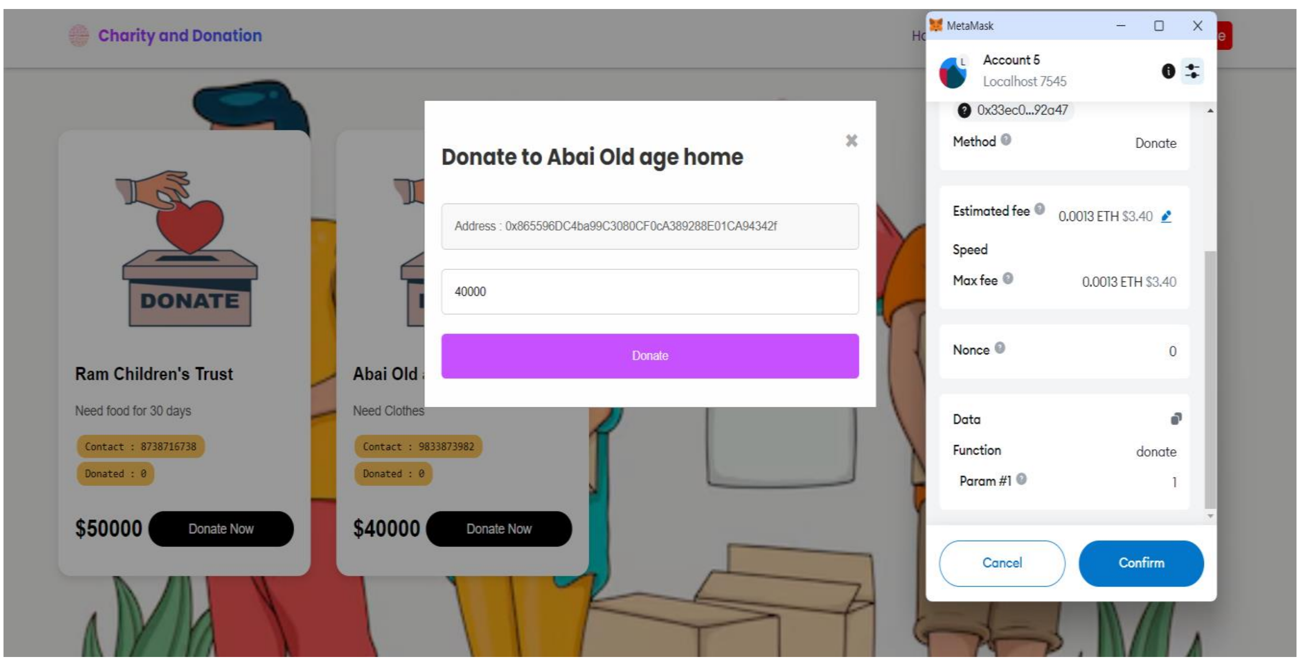
Contract Deployment



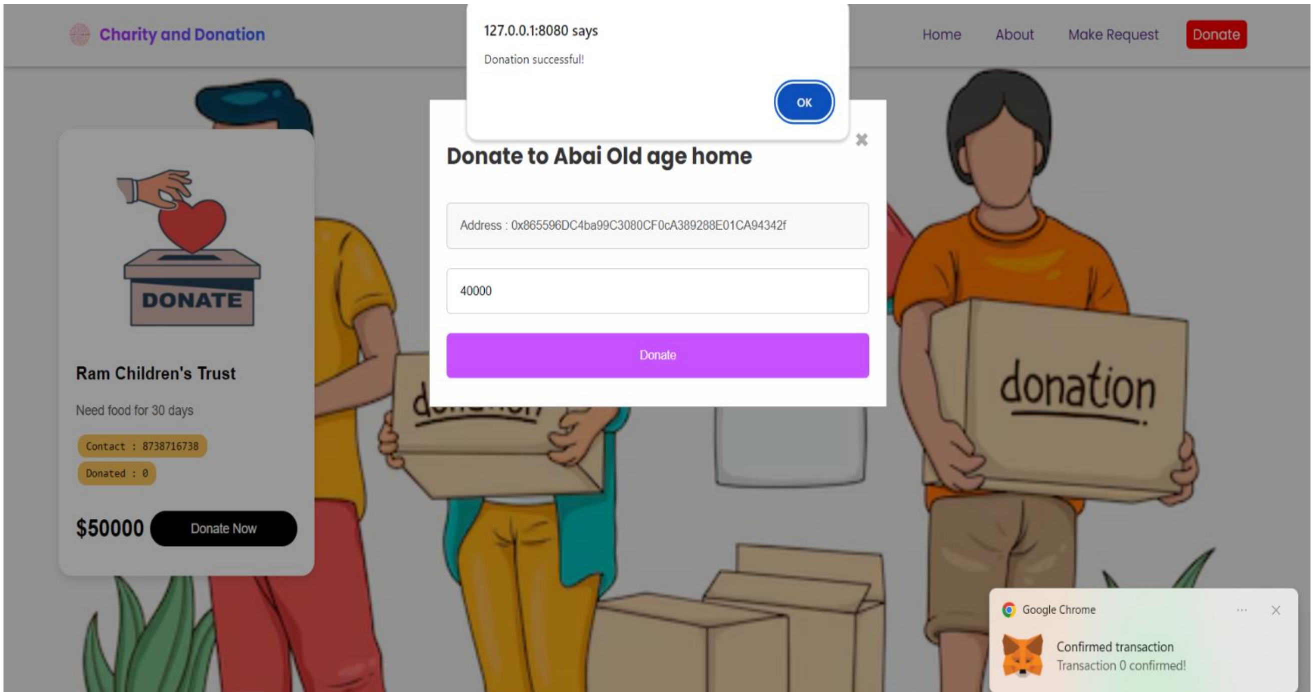


Make charity request





Donate request



ADDRESS	BALANCE	TX COUNT	INDEX
0x865596DC4ba99C3080CF0cA389288E01CA94342f	100.18 ETH	2	1
0xBa540f4f971023459a833f116E2757dc184AF9Ed	99.82 ETH	1	2

Ether after transaction

5.2 Donation Request Creation

Users and charitable organizations can raise donation requests by filling out a form that captures essential information such as:

- **Trust Name:** The organization or individual requesting donations.
- **Purpose of Donation:** A clear explanation of the cause or need for which the funds are being requested (e.g., "Donate for food").
- **Contact Information:** A phone number or email address for further inquiries.
- **Requested Amount:** The total amount being sought through donations.

Upon submission, the form triggers a smart contract, which records the donation request on the Ethereum blockchain. This process ensures that the request is securely stored and verifiable by anyone, safeguarding against any unauthorized alterations.

5.3 Donation Process

Donors can browse active donation requests and contribute to the cause of their choice. The donation process involves:

- **Ethereum Address:** The smart contract address to which the funds are transferred.
- **Amount to Donate:** The sum the donor wishes to contribute.

Upon initiating the donation, the platform interacts with the MetaMask wallet, allowing donors to confirm and authorize the transaction. Once confirmed, the donation is securely recorded on the blockchain, with all transaction details transparently available for audit.

5.4 Transaction Confirmation via MetaMask

The platform integrates with MetaMask for handling cryptocurrency transactions. Donors confirm transactions through MetaMask, which securely processes the donation on the Ethereum network. A notification confirms successful completion of the transaction, and the blockchain guarantees the traceability and immutability of each donation.

5.5 Smart Contract Mechanism

The underlying Ethereum smart contract manages all donation-related processes, ensuring automatic execution of transactions and adherence to predefined rules without the need for intermediaries. This decentralized approach enhances trust between donors and beneficiaries, as all records are publicly accessible and verifiable.

CHAPTER 6

Results and Observations

6.1 Platform Performance

The platform successfully demonstrated the advantages of using blockchain for charitable donations:

- **Transparency:** Each donation request and contribution is stored on the blockchain, allowing all participants to verify the status of donations at any time.
- **Security:** The Ethereum smart contracts eliminate the possibility of tampering with donation data, ensuring that funds reach the intended recipients securely.
- **Decentralization:** By eliminating intermediaries, the platform reduces the overhead associated with managing donations, making the process more efficient and trustworthy.

6.2 Functional Testing and Validation

The platform underwent extensive functional testing, covering all aspects of the donation process:

- **Donation Requests:** New requests were successfully created and stored on the blockchain, with accurate reflection of all submitted data.
- **Transactions:** Donations were successfully processed via MetaMask, with prompt transaction confirmation and transparency regarding the transaction details.

- **Blockchain Auditability:** Donors and beneficiaries could track the flow of donations and verify the total amount raised for specific causes, reinforcing the system’s trustworthiness.

6.3 User Experience

The platform delivered a smooth and intuitive user experience:

- **Donation Request Form:** The form for creating donation requests was streamlined, ensuring a quick and efficient process for organizations seeking funding.
- **Donation Process:** The integration with MetaMask made it simple for donors to contribute to causes using their Ethereum wallets, with clear instructions guiding them through the transaction process.

6.4 Observations

Several important observations were made during the implementation and testing phases:

- **Smart Contract Efficiency:** The smart contract executed as expected, ensuring that donations were transferred to the correct trust or organization immediately upon confirmation of the transaction.
- **Transaction Costs:** A small gas fee was required for each transaction, which was handled smoothly through MetaMask.
- **Scalability:** The platform demonstrated potential for scalability, with the ability to manage multiple donation requests and concurrent transactions without performance degradation.

CHAPTER 7

Conclusion

The blockchain-based charity and donation platform successfully integrates decentralized technology to address the common challenges associated with trust, transparency, and security in charitable giving. By leveraging Ethereum smart contracts, the platform ensures that all transactions are secure, immutable, and verifiable, providing donors with confidence that their contributions are being used as intended. The use of smart contracts automates the fund management process, ensuring that donations are only released when specific predefined conditions are met, thereby reducing the risk of fund mismanagement and eliminating the need for costly intermediaries.

Furthermore, the platform offers a scalable solution for managing charitable donations in a transparent, efficient, and tamper-proof manner. Decentralized storage through IPFS ensures that campaign details and other critical data are securely stored and easily accessible without the reliance on centralized systems. This enhances data integrity and accountability, making the system highly reliable for both donors and charitable organizations. The platform's design also makes it flexible enough to be expanded for use across multiple charitable organizations and larger-scale applications, allowing for greater collaboration within the nonprofit sector. Overall, the platform represents a significant step forward in modernizing charitable donations, bringing greater transparency and trust to the process, and making it more efficient, secure, and accessible for all stakeholders involved.

CHAPTER 8

APPENDIX

Solidity Program:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;

contract Charity {
    struct Request {
        string name;
        string description;
        int256 contact;  // Changed int to int256
        uint256 amount;
        uint256 received;
        address payable recipient;
        bool complete;
        uint256 approvalCount;
    }

    Request[] public requests;
    uint256 public numRequests;
    // Event for new request creation
    event RequestCreated(uint256 requestId, string name, uint256 amount, address recipient);

    // Function to create a new charity request
    function createRequest(
        string memory _name,
        string memory _description,
        int256 _contact,  // Changed int to int256
        uint256 _amount,
        address payable _recipient
    ) public {
        Request memory newRequest = Request({
            name: _name, description: _description,  contact: _contact, amount: _amount, received: 0,
recipient: _recipient, complete: false, approvalCount: 0
        });

        requests.push(newRequest);
        numRequests++;
        emit RequestCreated(numRequests - 1, _name, _amount, _recipient); // Emit event
    }
    // Retrieve information about a specific request
```

```
    function getRequest(uint256 _requestId) public view returns (
        string memory name, string memory description,
        int256 contact, uint256 amount,
        uint256 received, address recipient,
        bool complete, uint256 approvalCount
    ) {
        require(_requestId < requests.length, "Request does not exist"); // Validate request ID
        Request memory req = requests[_requestId];
        return (req.name, req.description, req.contact, req.amount, req.received, req.recipient,
req.complete, req.approvalCount);
    }

    // Donate to a specific request
    function donate(uint256 _requestId) public payable {
        require(_requestId < requests.length, "Request does not exist"); // Validate request ID
        Request storage request = requests[_requestId];
        require(!request.complete, "Donation Request already completed");
        require(msg.value > 0, "No donation amount provided");

        // Update the received amount
        request.received += msg.value;

        // Check if the request is fully funded
        if (request.received >= request.amount) {
            request.complete = true;
        }

        // Transfer Ether to the recipient
        request.recipient.transfer(msg.value);
    }

    // Fallback function to receive donations directly to the contract
    receive() external payable {}

    // Function to check the contract's balance
    function getContractBalance() public view returns (uint256) {
        return address(this).balance;
    }

    // Function to get the number of requests
    function requestLength() public view returns (uint256) {
        return requests.length;
    }
}
```

CHAPTER 9
REFERENCES

1. https://www.researchgate.net/publication/343026256_Aid_Charity_and_Donation_Tracking_System_Using_Blockchain

2. https://www.sciencedirect.com/science/article/pii/S1319157822003512#:~:text=Features%20of%20blockchain%2Dbased%20charity%20donation%20systems.&text=The%20proposed%20system%20focuses%20on,by%20the%20relevant%20needy%20party.

3. https://iopscience.iop.org/article/10.1088/1757-899X/768/7/072020

4. https://ieeexplore.ieee.org/document/9480649