IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Blockchain Based Fund Management System

Kona Rajesh kumar[1], Krishna Vamsi Pulipati[2], S Sameer Rithwik[3], S V Pavan Sree Vathsav[4]

[1]Student CSE(IOT, Cyber security incl. Blockchain Technology, Vasireddy Venkatadri Institute of Technology , Nambur , Guntur
[2]Student CSE(IOT, Cyber security incl. Blockchain Technology, Vasireddy Venkatadri Institute of Technology , Nambur , Guntur
[3]Student CSE(IOT, Cyber security incl. Blockchain Technology, Vasireddy Venkatadri Institute of Technology , Nambur , Guntur
[4]Student CSE(IOT, Cyber security incl. Blockchain Technology, Vasireddy Venkatadri Institute of Technology , Nambur , Guntur

Abstract: The Blockchain Based Fund Management System is an innovative online platform that utilizes blockchain technology to transfigure the operation of finances. This system introduces translucency and fairness by furnishing public access to project progress for all stakeholders. Client contribute funds in the form of cryptocurrencies, which are securely stored in smart contract.. These smart contracts insure the automatic allocation of funds to project directors grounded on predefined milestones. As design milestones are achieved and vindicated, separate finances are released automatically to the project manager or director's wallet. This ensures an indifferent distribution process and eliminates the need for centralized banks and third-party intervention. likewise, each milestone's data submitted by the manager is recorded on the blockchain, creating an inflexible transaction history. This real- time visibility enables guests, directors, and other stakeholders to singly corroborate the application of finances and the completion of milestone. The system's translucency and responsibility significantly reduce information imbalances and implicit controversies, fostering a cooperative and harmonious terrain among all stakeholders

Key Words: Blockchain, Fund Management, Transparency, Smart Contracts, Cryptocurrencies, Milestones, Accountability, Stakeholders

1.Introduction

Effective and transparent operation of fund management is essential for achieving project success. still, conventional approaches frequently face obstacles like detainments, uncertainty, and conflicts due to unstable access to information. To overcome these hurdles, the" Blockchain- Based Fund Management System" harnesses blockchain technology, famed for its decentralized and inflexible nature, to transfigure fund operation methodologies. Within this system, directors submit project milestones and accompanying attestation, which suffer rigorous confirmation upon completion. administrators strictly corroborate the authenticity of these documents to insure their integrity and delicacy. Once validated, the documents are digitized and securely stored within Inter Planetary File System(IPFS) and the hash generated from the IPFS is stored in Blockchain, staking on blockchain's essential traits of invariability and resistance to tampering. This guarantees that the information remains unaltered and empirical indefinitely.

By integrating the invariability of blockchain technology with the automation of smart contracts, the system facilitates flawless transfer of finances upon document verification. Smart contracts, being tone-executing agreements decoded with predefined rules on the blockchain, automate the fund transfer process, dwindling the need for centralized banks intervention and boosting functional effectiveness. In summary, the Blockchain- Based Fund Management System optimizes fund operation procedures, abridging time lags,

enhancing translucency, and mitigating conflicts by furnishing a secure and transparent frame for administering design finances.

2.Literature Review

Blockchain technology is poised to revise the future of project fund management by introducing transparency, effectiveness, security, and collaboration. At its core, blockchain is a decentralized and inflexible ledger that records transactions across a network of computers[1]. Each transaction is stored in a block , generated block is cryptographically linked to the former one, forming a chain of blocks that can not be altered or tampered with. This essential limpidity and stability make blockchain an ideal result for fund management operations, where trust and responsibility are consummate[1]. One of the vital ways in which blockchain will revise project fund management is through limpidity.
Traditionally, fund management processes have been opaque, with limited visibility into how finances are allocated and employed [2]. Blockchain changes this paradigm by furnishing real- time access to transaction data .Stakeholders, including guests, design directors, and investors, can track fund movements and project progress with limpidity. This limpidity not only builds trust among stakeholders but also helps to prevent fraud and ensure compliance with regulations[2]. effectiveness is another area where blockchain will drive significant change in project fund management. Centralized banking and third-party payment processes for fund allocation and distribution are time- consuming and prone to security issues. Blockchain streamlines these processes through the use of smart contracts, which are tone- executing contracts with predefined rules deciphered on the blockchain. Smart contracts automate fund transfers rested on predefined milestones, reducing the need for third-party intervention and minimizing detainments.[3] This robotization not only saves time and coffers but also ensures that finances are expended directly and efficiently. Security is a critical concern in fund management, where sensitive fiscal data is at trouble of theft or manipulation[3]. Blockchain technology addresses these security enterprises through its cryptographic features and decentralized armature. Each transaction is paraphrased and vindicated by network nodes, making it nearly insolvable for unauthorized parties to tamper with the data. also, the decentralized nature of blockchain means that there's no single point of failure, reducing the trouble of hacking or data breaches. Blockchain's transparent and decentralized nature also facilitates collaboration. By furnishing a sharing and accessible platform for fund management exertion, blockchain fosters collaboration and invention among stakeholders. Stake holders, directors can unite more effectively, participating in information in real- time. also, blockchain eliminates the need for intermediaries , similar to banks or brokers, reducing costs and streamlining communication between stakeholders[4].

3. Theoretical Approach

3.1 Introducing Blockchain for payments

Introducing blockchain for payments revolutionizes the traditional fiscal geography by offering a decentralized and transparent approach to transaction processing. At its core, blockchain technology serves as a distributed ledger that records all transactions across a network of decentralized nodes. Each transaction is cryptographically secured and vindicated by network nodes, barring the need for intermediaries like banks or payment processors. This decentralized nature ensures lesser security threats, as transactions are inflexible formerly recorded, reducing the threat of fraud or manipulation. also, blockchain payments enable briskly agreement times compared to traditional styles, as payments can be reused and verified within twinkles, anyhow of geographical position or banking hours.

This speed and effectiveness streamline the payment process. likewise, blockchain payments foster fiscal addition by furnishing access to banking services for the unbanked and underbanked populations. With blockchain- grounded payment results, individualities without access to traditional banking structure can securely make payments and admit finances using only a smartphone and an internet connection. This empowers individualities in underserved regions to share in the global frugality, easing cross-border payments and remittances without the need for third-party banks and payment processors. also, blockchain payments charge lower rate compared to traditional banking services, making fiscal payments more affordable and accessible to a broader range of people.

3.1.2 Impact of using Blockchain for Project Fund Management

The adoption of blockchain for project fund management introduces transformative changes to the traditional fund allocation and operation processes. One significant impact lies in the enhanced translucency and responsibility swung by blockchain technology. Through the use of decentralized, checks every stage and allocation within the project fund management system is recorded immutably and transparently. This translucency not only fosters trust among stakeholders but also enables real- time shadowing and auditing of fund allocations. also, the cryptographic security mechanisms essential in blockchain technology safeguard the integrity of the fund management system, reducing the threat of fraud or manipulation. Another notable impact of exercising blockchain for design fund operation is the streamlining of executive processes and the reduction of central costs. By automating fund allocation and distribution through smart contracts, the need for intermediaries similar as banks or fiscal institutions is minimized. Smart contracts execute predefined rules and conditions autonomously, barring the need for homemade intervention and reducing the associated executive outflow. This wisdom not only enhances the overall effectiveness of fund management processes but also reduces transaction costs and detainments, eventually optimizing resource application within the project ecosystem.

3.2 Stages involved in the field of Blockchain based funds management system for Projects.

3.2.1 Project Assignment.

Supervisor / Client assigns project (includes project details like project name, maximum allocated funds, maximum time allocated) to a Project Manager. On project assignment, manager submits blueprint of project (includes number of stages , each stage maximum fund allocation , each stage time allocation) and sends a request for verification. Supervisor verifies the data submitted by manager .

3.2.2 Stage Details Submission.

After completion of each stage, manager shall request for funds allocated to respective stage by submitting required documents (includes proof of completion files and other documents). On submitting the data a request will be sent to supervisor to release funds.

3.2.3 Supervisor Validation

After receiving a request from manager, Supervisor will verify files submitted by manager and will verify or reject the request. Before a successful verification of a stage ,Supervisor Should submit a digitalized document of proof of verification in which he/she shall state the documents submitted are legit and links to the files submitted by manager and funds allocated . This data is accessible to the public which makes the entire process transparent.

3.2.4 Smart Contract Role

Smart Contract is responsible for interacting, storing records submitted by both parties and fund allocation and transfer . Smart contract also has functions that can be called upon by any person in-order to view the data submitted by any party and to know the stage details or project details for free of cost which are publicly accessible.

4.Tranditional System and its Problems

4.1 Types of Traditional system for funds management

Manual Systems In traditional systems, finances operation relies on paper- grounded records, spreadsheets, and homemade computations. This system is labor- ferocious, prone to mortal error, and lacks real- time visibility into fund movements. Banking Systems numerous associations use banking systems handed by fiscal institutions to manage their finances. These systems offer introductory features similar as account operation, transfers, and introductory reporting. still, they may warrant advanced functionalities and customization options. Enterprise Resource Planning(ERP) Systems ERP systems integrate colourful business processes, including finances operation, into a single platform. While ERP systems offer comprehensive functionalities, they can be complex and expensive to apply, making them more suitable for large enterprises. Accounting Software Accounting software packages, similar as QuickBooks or Xero, frequently include features for managing finances, similar as tracking income and charges, generating fiscal

reports, and coordinating bank accounts. These systems are popular among small and medium- sized businesses due to their ease of use and affordability. Investment Management Systems Financial institutions and investment enterprises use technical investment operation systems to oversee investment portfolios, dissect request trends, and execute trades. These systems frequently incorporate advanced analytics and threat operation tools acclimatized to the requirements of investment professionals. Cash Management Systems Cash operation systems concentrate on optimizing cash overflows, liquidity, and working capital within an association. They generally include features similar as cash soothsaying, liquidity operation, and storeroom operations to insure effective use of finances. Donation Management Systems Nonprofit associations and charities use donation operation systems to track and manage incoming donations and induce patron reports. These systems frequently include features for patron relationship operation and fundraising crusade shadowing.

4.2 Problems with traditional funds management system

4.2.1. Time-consuming Fund Allocation Process

In traditional fund management systems, the process of allocating funds to different stages of a project can be time-consuming and inefficient. Each stage of fund allocation typically involves multiple intermediary institutions, including project managers, financial officers, and accounting departments. These intermediary institutions may operate with independent accounting systems, requiring manual reconciliation and clearance of funds. As a result, the fund allocation process can be prolonged, leading to delays in project execution and hindering overall project efficiency.

4.2.2. Costly Fund Allocation Fees

The fund allocation process in traditional systems often incurs significant fees and expenses, adding to the overall cost of project management. Project managers may encounter high transaction fees when allocating funds through intermediary institutions, such as banks or financial service providers. These fees can vary depending on the size and complexity of the project, as well as the number of intermediary institutions involved. Additionally, foreign currency remittances may incur additional charges, such as foreign currency spreads and telegraphic fees, further increasing the cost of fund allocation. These expenses can impact project budgets and reduce the amount of available funds for project execution.

4.2.3. Inefficient Capital Utilization

Traditional fund management systems may result in inefficient utilization of capital due to the large amount of funds held in transit during the allocation process. Project funds may be tied up in margin accounts across different intermediary institutions, reducing the efficiency of fund utilization. Moreover, banks may need to hold currencies of multiple countries in their accounts to facilitate cross-border transactions, increasing hedging and opportunity costs. This inefficient capital utilization can limit the availability of funds for project execution and hinder project progress.

4.2.4. Security Risks in Fund Allocation

The traditional fund allocation process is vulnerable to security risks, particularly in centralized payment systems where customers must provide sensitive account information to intermediaries. This information, including account details and transaction records, may be susceptible to hacking or unauthorized access, posing a threat to data security and privacy. Furthermore, third-party payment platforms involved in fund allocation may also be at risk of information leakage, exposing personal and financial data to potential exploitation. These security risks can undermine trust in the fund allocation process and jeopardize the integrity of project finances.

4.2.5. Lack of Transparency in Fund Allocation

In traditional fund management systems, there is often a lack of transparency in the fund allocation process, leading to uncertainty and ambiguity for project stakeholders. The intricate network of intermediary institutions and opaque accounting systems may obscure the flow of funds and associated transaction fees. As a result, project managers and stakeholders may have limited visibility into how funds are allocated across

different stages of the project. This lack of transparency can impede accountability and make it difficult to track the movement of funds, potentially resulting in disputes or discrepancies.

4.2.6. Compliance Challenges in Fund Allocation

Traditional fund allocation systems are susceptible to compliance challenges, particularly in cross-border transactions where regulatory requirements vary across jurisdictions. Project managers may encounter difficulties ensuring compliance with anti-money laundering (AML) and know-your-customer (KYC) regulations, as well as other legal and regulatory obligations. The complexity of regulatory frameworks and the need to navigate multiple jurisdictions can pose operational challenges and increase the risk of non-compliance. Moreover, regulatory changes or updates may further complicate compliance efforts, requiring ongoing monitoring and adaptation of fund allocation processes to remain compliant with evolving regulations. Addressing these compliance challenges requires robust systems and processes to ensure adherence to regulatory standards and mitigate associated risks.

5.Blockchain Based Fund Management System Application Model

5.1 Smart Contract Code

(each subset of code is explained under “Using and Approach Section [5.2])

5.1.1 Pre-requisites (import necessary packages and state license)

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
contract ProjectManagement is ERC721, Ownable {
    // Define struct to store project details
    struct Project {
        string projectName;
        uint256 maxAllocation;
        address builder;
        bool assigned;
        bool startproject;
        bool stageFundsSubmitted;
        bool detailsSubmitted;
        uint256 numStages;
        mapping(uint256 => Stage) stages;
        uint256 totalAllocatedFunds;
    }
    // Define struct to store stage details
    struct Stage {
        string description;
        uint256 amount;
        bool submitted;
        bool verified;
    }
    // Mapping to store projects
    mapping(address => Project) public projects;
      address[] public assignedBuilders;
    // Event emitted when a project is assigned
    event ProjectAssigned(address indexed supervisor, address indexed builder);
    // Event emitted when project details are submitted
    event DetailsSubmitted(address indexed builder);
    // Event emitted when stage details are submitted
    event StageSubmitted(address indexed builder, uint256 indexed stage);
    // Event emitted when stage is verified
    event StageVerified(address indexed builder, uint256 indexed stage);
    // Event emitted when stage is rejected
```

```
    event StageRejected(address indexed builder, uint256 indexed stage);
    // Event emitted when project details are rejected
    event ProjectDeleted(address indexed builder);
    // Event emitted when project starts
    event ProjectStarted(address indexed builder);
    constructor(address initialOwner) ERC721("Project NFT", "PNFT") Ownable(initialOwner) {
_buildertokenId=1; _supervisortokenId=1; }
```

5.1.2 Function for supervisor to assign a project to a builder.

```
function assignProject(address builder, string memory projectName, uint256 maxAllocation) external
onlyOwner {
        require(!projects[builder].assigned, "Project already assigned to builder");
        Project storage newProject = projects[builder];
        newProject.projectName = projectName;
        newProject.maxAllocation = maxAllocation;
        newProject.builder = builder;
        newProject.assigned = true;
        newProject.detailsSubmitted = false;
        newProject.startproject = false;
        newProject.stageFundsSubmitted = false;
        newProject.numStages = 0;
        newProject.totalAllocatedFunds = 0;
          assignedBuilders.push(builder);
        emit ProjectAssigned(msg.sender, builder);
    }
```

5.1.3 Function for builder to submit project details

```
function submitProjectDetails(uint256 numStages) external {
        Project storage project = projects[msg.sender];
        require(project.assigned && !project.detailsSubmitted, "Project not assigned or details already
submitted");
        project.numStages = numStages;
        project.detailsSubmitted = true;
        emit DetailsSubmitted(msg.sender);
    }
```

5.1.4 Function for Builder to submit stage funds allocation

```
function stageFunds(uint256 stage, uint256 amount) external {
        Project storage project = projects[msg.sender];
        require(project.assigned && project.detailsSubmitted, "Project not assigned or details not submitted");
        require(stage > 0 && stage <= project.numStages, "Invalid stage number");
        require(project.totalAllocatedFunds + amount <= project.maxAllocation, "Total allocated funds exceed
max allocation");
        project.totalAllocatedFunds += amount;
        project.stages[stage].amount = amount;
        // Check if all stages have their amounts provided
        bool allStagesFunded = true;
        for (uint256 i = 1; i <= project.numStages; i++) {
            if (project.stages[i].amount == 0) {
                allStagesFunded = false;
                break;
            }
        }
        if (allStagesFunded) {
            project.stageFundsSubmitted = true;
```

```
    }
}
```

5.1.5 Function for Supervisor to verify funds allocation for each stage and project approval

```
function startProject(address builder) external onlyOwner payable {
    Project storage project = projects[builder];
    require(project.assigned, "Invalid project ");
    require(!project.startproject, "Project already started");
    require(project.stageFundsSubmitted, "Stage Fund details are not submitted yet!");
    // Ensure the amount sent by the supervisor matches the total allocated funds
    uint256 totalFunds = project.totalAllocatedFunds;
    require(msg.value == totalFunds, "Incorrect total funds sent");
    project.startproject = true;
    emit ProjectStarted(builder);
}
```

5.1.6 Function for builder to submit stage details

```
function submitStage(uint256 stage, string memory description,string calldata _uri) external {
    Project storage project = projects[msg.sender];
    require(project.assigned && project.detailsSubmitted, "Project not assigned or details not submitted");
    require(stage > 0 && stage <= project.numStages, "Invalid stage number");
    require(project.startproject, "Project is not yet set to start! Please wait for supervisor confirmation");
    require(project.stages[stage].amount > 0, "Stage funds allocation not provided");
    // Retrieve stage funds allocation
    uint256 amount = project.stages[stage].amount;
    // Set stage details
    project.stages[stage] = Stage(description, amount, true, false);
     _mint(msg.sender, _buildertokenId);
    _setTokenURI(_buildertokenId, _uri);
    emit StageSubmitted(msg.sender, stage);
}
```

5.1.7 Function to retrieve stage details for a given project

```
function getStageDetails(address Builder,uint256 stage) external view returns (string memory description,
uint256 amount, bool submitted, bool verified) {
    Project storage project = projects[Builder];
    require(project.assigned && project.stages[stage].submitted, "Invalid project or stage");
    Stage storage stageDetails = project.stages[stage];
    return (stageDetails.description, stageDetails.amount, stageDetails.submitted, stageDetails.verified);
}
```

5.1.8 Function for supervisor to verify stage details and transfer funds to builder

```
function verifyStage(address payable builder, uint256 stage,string calldata _uri) external onlyOwner payable
{
    Project storage project = projects[builder];
    require(project.assigned && project.stages[stage].submitted && !project.stages[stage].verified, "Invalid
project or stage");
    // Retrieve the amount allocated for this stage
    uint256 amount = project.stages[stage].amount;
    // Transfer the funds to the builder
    require(address(this).balance >= amount, "Contract balance is insufficient");
    builder.transfer(amount);
    // Mark the stage as verified
    project.stages[stage].verified = true;
```

```
        _mint(msg.sender, _supervisortokenId);
        _setTokenURI(_supervisortokenId, _uri);
        emit StageVerified(builder, stage);
        // Check if all stages are verified
        bool allVerified = true;
        for (uint256 i = 1; i <= project.numStages; i++) {
            if (!project.stages[i].verified) {
                allVerified = false;
                break;
            }
        }
        // Reset builder status if all stages are verified
        if (allVerified) {
            project.assigned = false;
            delete projects[builder];
        }
    }
```

5.1.9 Function to reject stage details submitted by builder.

```
function rejectStage(address Builder, uint256 stage) external onlyOwner {
        Project storage project = projects[Builder];
require(project.assigned&& project.stages[stage].submitted&& !project.stages[stage].verified, "Invalid project or stage");
        delete project.stages[stage];
        emit StageRejected(Builder, stage);
    }
```

5.2 Model usage and Approach.

At first Supervisor will assign a project to a contractor. Upon selection of contractor, the supervisor communicates the project details including project name, maximum budget to the contractor. [function reference 5.1.2]

After being assigned a project, the contractor begins by submitting project details to the supervisor. This could include blueprints, project timelines, and resource requirements. [function reference 5.1.3]

Once the project is approved, the contractor breaks down the project into stages and allocates funds accordingly. Contractor submits these stage fund allocations to supervisor for review.[function reference 5.1.4]

After receiving the stage fund allocation details from contractor, supervisor reviews them to ensure they align with the project budget and requirements. Once satisfied, supervisor formally approves the project to commence and send the allocated funds to smart contract.[function reference 5.1.5]

With the project officially started, contractor begins work on each stage. On completion of each stage, contractor submits the stage details for review.[function reference 5.1.6]

Upon receiving stage details from contractor, supervisor reviews. If satisfied, supervisor releases the funds allocated for the stage to the contractor.[function reference 5.1.8]

In cases where the stage details submitted by the contractor do not meet the requirements, supervisor may reject them.[function reference 5.1.9]

5.3 Results

A user has two options, Supervisor, or builder/contractor.
User should have a MetaMask wallet in order to login.

Figure 1 [Contractor Login]

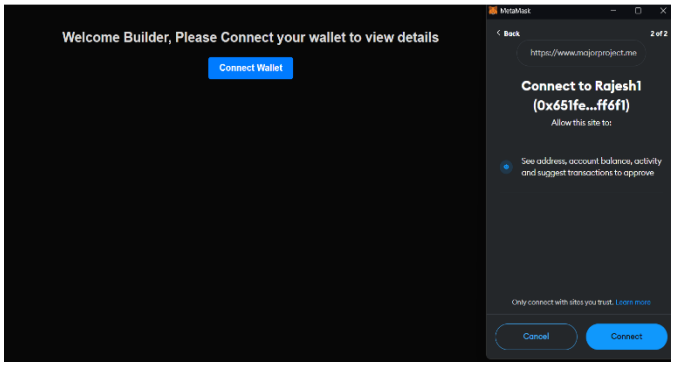

Figure 2 [Contractor accepts wallet connection]

After succesful login, contractor can request for funds from supervisor. To do so , contractor needs to submit required documents and proof of completion in a form. After successfully requesting for funds from supervisor, an NFT is sent to contractor address as a proof for requesting funds .

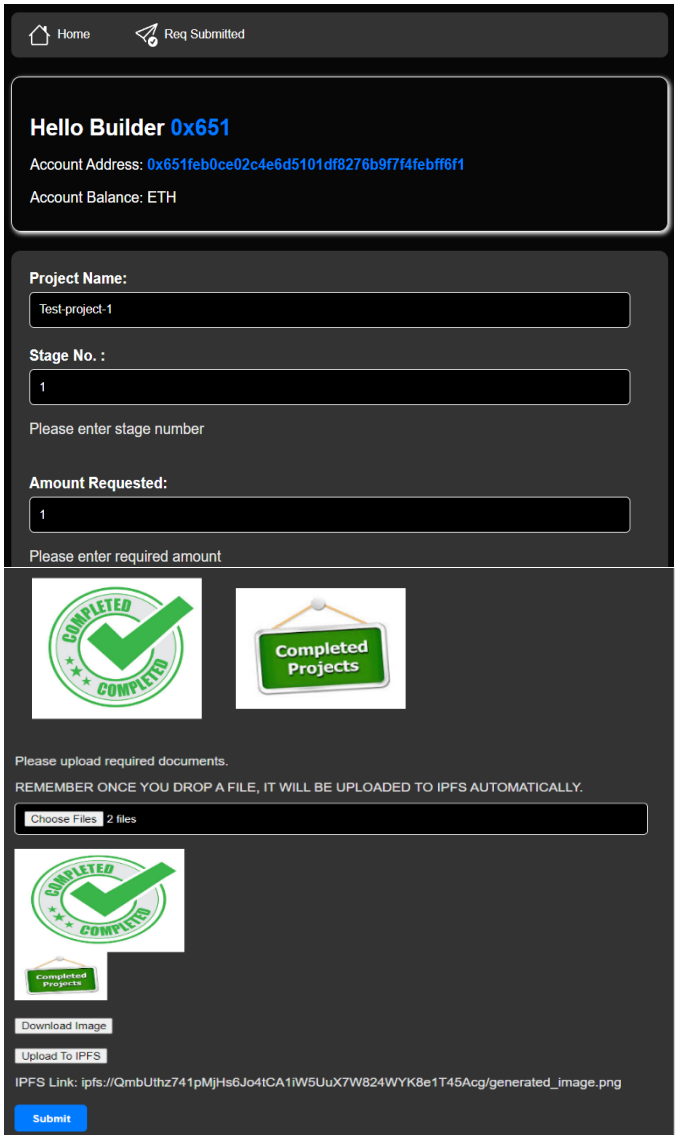

Figure 3 [Contractor submits stage details with proof]

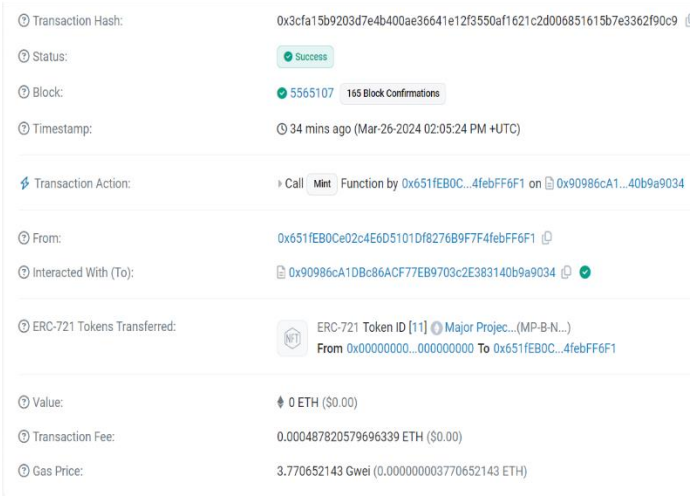

Figure 4 [successful transaction]

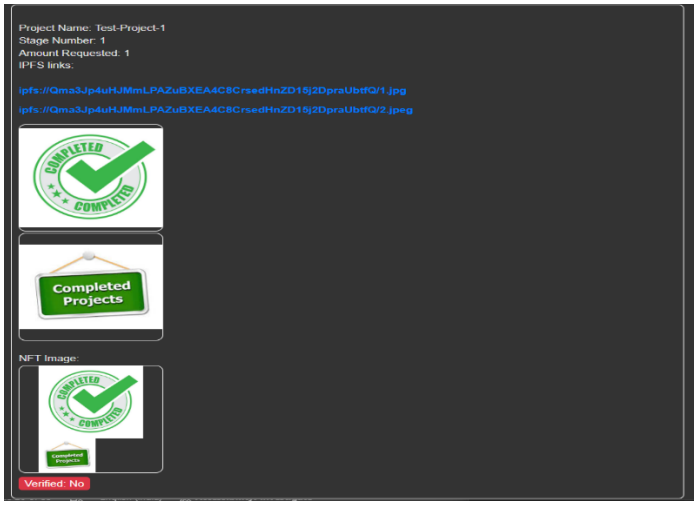

Figure 5 [Details submitted by Contractor]

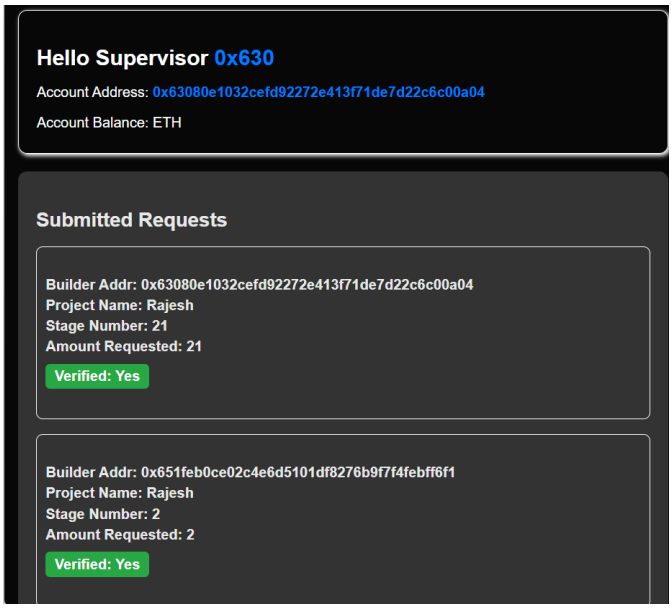

Figure 6 [Supervisor dashboard]

Now supervisor can view the submitted project details and check the status of each project , details submitted. Supervisor should verify the details.

Figure 7 [Supervisor verifies stage details]

Once the transaction is successful, an NFT is sent to supervisor address as a proof of fund allocation and funds are transferred from smart contract to contractor address.

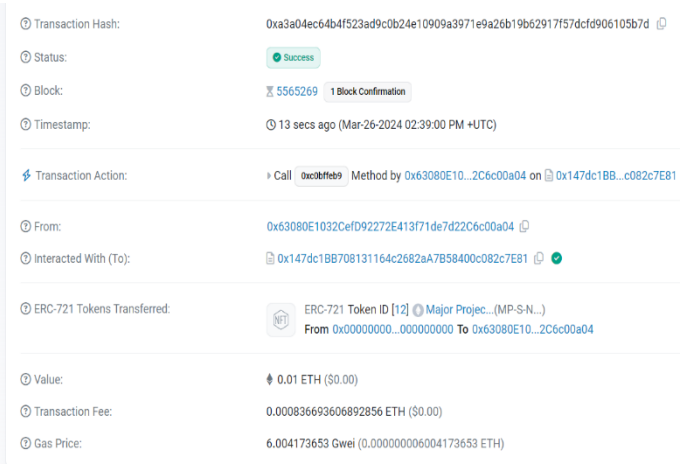

Figure 8 [successful transaction]

6. Constraints Faced by Blockchain Technology in Fund Management

Limitations in Blockchain Technology, presently, blockchain technology is still in its initial stage of development, facing challenges in handling large- scale data volumes generated by the fund operations. The system's capacity to support high- concurrency reclamation and queries is limited, making it unfortunate for large- scale fund operation results. also, there exists a trade- off between the degree of decentralization and the performance of the agreement medium. Advanced decentralization frequently leads to lower effectiveness in the agreement medium, performing in longer sale detainments and reduced outturn. Balancing decentralization and agreement medium effectiveness is pivotal for the development of blockchain technology. also, while blockchain offers traceability and tamper resistance, it also poses limitations. Adding sale information requires agreement from the entire system, and latecomers to the blockchain network must bear substantial costs in terms of time and storehouse space to gain a complete dupe of the blockchain data.

7. CONCLUSIONS

Although blockchain technology faces constraints and challenges in fund operation, its disruptive eventuality in different payments and other sectors is inarguable. Unborn development sweats should concentrate on addressing being limitations and advancing blockchain technology. Collaboration among request actors and transactional exchanges is essential to explore results and promote the relinquishment of blockchain technology. at the same time, measures must be taken to alleviate blockchain pitfalls and enhance the security of fund operation processes.

REFERENCES

[1] Jayanth Rama Varma, "Blockchain in Finance".
[2] Pingquan Wang, "Application of Blockchain in Financial Management ".
[3] Katharina Sigalov , Xuling Ye , Markus König and Philipp Hagedorn, "Automated Payment and Contract Management in the Construction Industry by Integrating Building Information Modeling and Blockchain-Based Smart Contracts"
[4] Hewavitharana T, Nanayakkara S and Perera S,"Blockchain as a project management platform",2019.
[5] Jansi Rani SELLA VELUSWAMI, Yamini LAKSHMI NARSIMHAN, Shivaani KRISHNAKUMAR, "Blockchain based system for transfer of funds through an e-Governance application", Vol. 32, No. 4, 93-102, 2022.
[6] Qing Deng,"Application Analysis on Blockchain Technology in Cross-border Payment
[7] Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.
[8] Li Chunting, Lian Zhigang, & Li Guoming. (2019). Financial management system based on automatic identification of bills. Journal of Shanghai Dianji University, 022 (004), 227-232.
[9] S. Aggarwal, R. Chaudhary, G. S. Aujla, N. Kumar, K. R. Choo, and A. Y. Zomaya, Blockchain for smart communities: Applications, challenges and opportunities, J. Netw. Comput. Appl., vol. 144, pp. 13-48, 2019.