Decentralized Crowdfunding Platform for Startup Initiatives: Utilizing Blockchain for Transparent Fund Allocation

Mehmet Bayram Alpay Yusuf Batuhan Kılıçarslan Kadir Gökdeniz

20290310 21290683 20290344

Ankara University Department of Computer Engineering

June 2024

1 Background

Blockchain is a revolutionary technology that enables secure, transparent, and decentralized transactions. It is widely used in various fields such as finance, supply chain management, and crowdfunding. This section provides a brief overview of the key concepts of blockchain technology, including nodes, transactions, consensus mechanisms, and smart contracts[1].

1.1 Nodes

Nodes are the fundamental components of a blockchain network. Each node holds a copy of the entire blockchain ledger and participates in the validation and propagation of transactions. Nodes can be classified into different types based on their functionality, such as full nodes, lightweight nodes, and mining nodes.

1.1.1 Full Nodes: These nodes store the complete blockchain and validate all transactions and blocks.

1.1.2 Lightweight Nodes: Also known as SPV (Simplified Payment Verification) nodes, they store only block headers and rely on full nodes for transaction verification.

1.1.3 Mining Nodes: Mining nodes are necessary nodes for creating new blocks and validating transactions. They are crucial for the security of the network in projects using consensus mechanisms such as Proof of Work (PoW). Mining nodes also play a significant role in influencing network

performance. With high transaction capacities and properly structured network infrastructures, mining nodes ensure the continuous security and resilience of the network. Additionally, they enhance decentralization, which positively impacts trust among participants.

1.2 Transactions

A transaction is a fundamental unit of operation in a blockchain network. It represents the transfer of value or data between participants. Each transaction is digitally signed by the sender and verified by the network nodes. Transactions are grouped into blocks and added to the blockchain in a linear, chronological order.

1.2.1 Structure of a Transaction: A typical blockchain transaction includes inputs (references to previous transactions), outputs (recipient addresses and amounts), and a digital signature.

1.2.2 Transaction Lifecycle: A transaction goes through various stages, including creation, broadcast, validation, and confirmation before it becomes part of the blockchain.

In our project, transactions fulfill the requirements for value transfers and voting processes that participants need. Each transaction is digitally signed by the sender and verified by nodes on the network, enhancing the security and transparency of the system. Transactions are grouped into blocks and added to the blockchain in a linear, chronological order, enabling the entire process to be traced. A typical blockchain transaction includes references to previous transactions (inputs), recipient addresses and amounts (outputs), and digital signatures. Each transaction goes through various stages such as creation, broadcast, validation, and confirmation before being included in the blockchain. These structures and processes enable our project to provide a transparent, reliable, and traceable funding and management mechanism.

1.3 Consensus Mechanisms

Consensus mechanisms are fundamental to the functioning of a blockchain network. They ensure that all nodes in the network agree on the validity of transactions and the order in which they are added to the blockchain. This section explores various consensus mechanisms, including Proof of Work (PoW), Proof of Stake (PoS), and other notable algorithms, detailing their operations, benefits, and drawbacks.

Proof of Work (PoW)

Proof of Work is the first consensus mechanism used in blockchain, famously implemented by Bitcoin. It involves solving complex mathematical puzzles to create new blocks, which is a process known as mining [3].

1.1.1 Operation:

- Miners compete to solve a cryptographic puzzle.
- The first miner to solve the puzzle gets to create a new block and receives a reward (usually in the form of cryptocurrency).
- The new block is then broadcasted to the network, and other nodes validate it by checking the solution.

1.1.2 Benefits:

- High security due to the computational difficulty of solving the puzzles.
- Proven track record with Bitcoin, the most secure blockchain to date.

1.1.3 Drawbacks:

- Energy-intensive, as it requires significant computational power.
- Slow transaction processing times due to the time it takes to solve puzzles.
- Centralization risks as mining becomes dominated by entities with substantial resources.

Proof of Stake (PoS)

Proof of Stake is a more energy-efficient alternative to PoW. Instead of miners, it uses validators who are chosen based on the number of tokens they hold and are willing to "stake" as collateral [4].

Operation:

- Validators are selected to propose and validate new blocks based on their stake.
- The likelihood of being chosen is proportional to the amount of cryptocurrency they hold and are willing to lock up as a stake.
- Validators receive transaction fees as rewards rather than new cryptocurrency.

Benefits:

- Energy-efficient, as it does not require solving complex puzzles.
- Faster transaction processing times compared to PoW.
- Reduces the risk of centralization, as it does not require expensive hardware.

Drawbacks:

- Potential for the "nothing at stake" problem, where validators might approve multiple competing chains without any cost.

- Requires a robust staking mechanism to ensure security and fairness.

Delegated Proof of Stake (DPoS)

Delegated Proof of Stake is a variant of PoS where stakeholders elect a small number of delegates to validate transactions and create new blocks [5].

Operation:

- Stakeholders vote for delegates by pooling their tokens.
- The elected delegates take turns producing blocks and validating transactions.
- Rewards are distributed among the delegates and their supporters.

Benefits:

- Highly scalable and capable of processing many transactions per second.
- More democratic as stakeholders have a direct influence on the selection of validators.
- Efficient and faster consensus due to a limited number of validating nodes.

- Drawbacks: - Centralization risks if a small number of delegates gain disproportionate control. - Potential for collusion among delegates to manipulate the network.

Practical Byzantine Fault Tolerance (PBFT)

PBFT is a consensus algorithm designed for permissioned blockchain networks. It aims to provide high throughput and low latency [6].

Operation:

- Nodes (or replicas) are divided into three categories: primary, backup, and client.
- The primary node proposes a block, and the backups verify and agree on the proposal.
- If a sufficient number of backups agree (usually 2/3), the block is added to the blockchain.

Benefits:

- High transaction throughput and low latency.
- Fault tolerance, as it can withstand a certain number of malicious nodes.
- Suitable for private and consortium blockchains with known participants.

Drawbacks:

- Limited scalability due to the communication overhead among nodes.
- Requires a majority of honest nodes to function correctly.

Other Consensus Mechanisms

There are several other consensus mechanisms designed to address specific use cases and improve upon the limitations of existing algorithms.

Proof of Authority (PoA): Validators are pre-approved and known entities, providing high efficiency and speed [7]. It is suitable for private blockchains but lacks decentralization.

Hybrid Consensus: Combines elements of different consensus mechanisms to balance security, efficiency, and decentralization. An example is Ethereum 2.0, which uses a combination of PoW and PoS. Also, other hybrid consensus mechanisms are available such as Tendermint, Casper, and EOSIO.

1.4 Smart Contracts

Smart contracts are self-executing contracts with the terms of the agreement directly written into code. They automatically enforce and execute the terms when predefined conditions are met, eliminating the need for intermediaries [8].

1.4.1 Characteristics of Smart Contracts: Automation, transparency, immutability, and efficiency.

1.4.2 Applications of Smart Contracts: Financial transactions, supply chain management, voting systems, and decentralized applications (DApps) [2].

1.4.3 Use of Smart Contracts in Crowdfunding:

- **Decentralized Crowdfunding Platforms:** Smart contracts enable the creation of decentralized crowdfunding platforms where the terms and conditions of the funding process are encoded in the contract [9]. These platforms opermissonedate without a central authority, ensuring that funds are only released when certain conditions are met, thereby protecting both the investors and the project creators.
- **Escrow Services:** In traditional crowdfunding, funds are held by the platform until the project reaches its funding goal. Smart contracts can act as automated escrow services, holding the funds and automatically releasing them when the project meets predefined milestones or refunding the contributors if the goal is not met within the specified timeframe.

- **Transparency and Trust:** The transparency provided by blockchain technology ensures that all transactions and fund movements are publicly verifiable. This transparency increases trust among contributors, as they can track how their funds are being used and ensure that the project creators are meeting their promises.
- **Reduced Costs:** By eliminating intermediaries, smart contracts reduce the costs associated with crowdfunding. This means that a larger portion of the funds raised can go directly to the project rather than being used to pay platform fees.
- **Global Accessibility:** Blockchain-based crowdfunding platforms are accessible to anyone with an internet connection, allowing projects to reach a global pool of potential backers. This inclusivity can lead to higher funding success rates and a more diverse group of contributors.

2 Construction

Our decentralized crowdfunding platform leverages blockchain technology to ensure transparent fund allocation and prevent fraud. This section details the architecture of our platform and answers key questions regarding its design and functionality.

2.1 Permissioned vs. Permissionless Blockchain

In traditional crowdfunding systems, companies manage funds with transaction confidentiality. This results in individuals participating in crowdfunding not being able to see the transaction flow and the campaign manager holding most of the authority. This situation raises various concerns among participants and results in their withdrawal from crowdfunding processes. The problem definition of our project stems from this point, aiming to provide the necessary transparency in created systems, enhance decision-making through a decentralized system, and minimize control manipulation. The goal is to establish a democratic process. Additionally, crowdfunding systems should inherently support a large number of participants. This leads us to the design of a permissionless system.

With a permissionless system design, we need to decide on some critical aspects of the system design. For example, democratic decision-making processes rely on voting systems. The successes and failures of individuals based on their roles are regulated with reputation scores, which helps monitor and prevent potential security issues. Voting systems can raise security concerns related to Sybil attacks. To mitigate the impact of this problem, maintaining DID (Decentralized Identity) records is targeted. Even if a Sybil attack occurs, monitoring the malicious actor with DID becomes easier, and reputation scores, along with particularly suspicious transaction voting, aim to address these attacks.

We chose Ethereum as our platform for the following reasons[10]:

Short Blockchain Creation Times: Ethereum has faster block creation times compared to Bitcoin, allowing transactions to be completed more quickly.
Lower Costs: It offers lower transaction costs than Bitcoin, making the system more economically efficient.
Ease of Implementation: The use of the Solidity programming language facilitates the development of Ethereum-based applications.
High Scalability for Crowdfunding Processes: Ethereum-based applications provide a suitable infrastructure for managing and implementing large-scale crowdfunding processes.
Widespread Usage and Security: Today, widely used Ethereum-based applications and coinbase systems demonstrate Ethereum's success in terms of security and reliability.

2.2 Entities and Their Roles

Our platform consists of the following entities:

2.2.1 Organizers: Organizers are responsible for campaigns, management, and promotion of projects. They ensure that the project follows the right path by taking on important responsibilities such as determining financial targets and managing funds. Publicly available reputation scores indicate the level of trustworthiness and activity. Behaviors such as unsuccessful campaign management or misuse of funds lead to a serious decrease in reputation points.

2.2.2 Suppliers: Suppliers provide the necessary materials and services and receive payment after delivery. Payment transactions are carried out using smart contracts to be fair and secure. Suppliers' participation in the fund is carried out through a reputation system based on reputation points. A high reputation score indicates that it is a safe supplier and these suppliers are allowed to join the system. Stricter and more cautious rules apply to suppliers with low reputation scores. Suppliers with low reputation scores may be included in the platform again based on participant voting. The supplier with a low reputation score can be included in the platform if it receives at least 70% of the votes of the participants. In this way, a reliable system that gives authority to the participants is created.

2.2.3 Participants: Participants play an important role by providing funds and participating in the voting process. In this way, participants take steps to meet the financial needs of the project and have a say in the progress of the project in line with their expectations. They may also have the opportunity to participate in important votes on the security side, such as system-initiated suspicious transaction investigations and statute of limitations decisions.

Each entity plays a crucial role in ensuring the transparency and success of crowdfunding campaigns.

2.3 Registration Process

The registration process for entities on our Decentralized Crowdfunding Platform for Startup Initiatives involves the following steps:

2.3.1 Decentralized Identity Verification (DID System): Participants use decentralized identifiers (DIDs) to establish their identities. Unlike traditional methods, this approach does not rely on a central authority. Instead, participants create their own DID documents which are securely stored on the blockchain. This ensures privacy and security, making the system permissionless and accessible to all.

2.3.2 Cryptographic Keys: Upon successful DID registration, participants are issued unique cryptographic keys that enable secure transactions and interactions on the platform. These keys are crucial for maintaining the integrity and security of user data and transactions.

2.3.3 Smart Contract Deployment: Organizers can deploy smart contracts to create new campaigns. These smart contracts specify the funding goals, deadlines, and other parameters, ensuring that all terms are transparent and automatically enforced without the need for intermediaries.

2.3.4 Fund Allocation and Management: The platform utilizes smart contracts to manage the allocation of funds. Funds are held in escrow by the smart contract and are released only when specific conditions are met, such as reaching funding milestones or deadlines. This ensures that funds are used transparently and according to the terms agreed upon.

2.3.5 Sybil Attack Mitigation: To prevent Sybil attacks and ensure the integrity of interactions, participants are required to undergo rigorous verification processes through the DID system. This mechanism deters malicious actors from creating multiple fake identities, thus safeguarding the platform's reliability[11].

This process ensures that only legitimate participants can join the platform and contribute to campaigns.

2.4 Use of Smart Contracts

Smart contracts are central to the operation of our crowdfunding platform. They automate various processes and enforce the rules of the platform, including:

2.4.1 Campaign Creation: Organizers deploy smart contracts to create new campaigns, specifying the funding goals, and deadlines.

2.4.2 Fund Allocation: Smart contracts manage the allocation of funds based on predefined rules and conditions, ensuring transparency and accountability.

2.4.3 Supplier Voting: To ensure a reliable system, it is important to mitigate risks. Verifying the authenticity of new suppliers joining the network is crucial due to its unauthorized structure. Therefore, suppliers maintain reputation scores to evaluate their acceptance into the network. For instance, a candidate with fewer than 20 reputation points undergoes a voting process for network inclusion. If the candidate receives over 70% positive votes, they are accepted; otherwise, the process is halted.

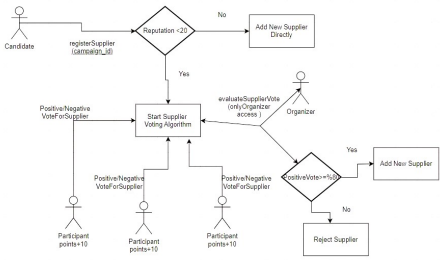

Figure 1: Supplier Voting Algorithm Diagram

2.4.4 Suspicious Transaction Voting: Suspicious Transaction Voting addresses cases where a manager initiates a vote by frequently sending funds to the same individuals or in high amounts. This voting assumes that suspicious transactions occur between the same parties involving significant amounts. For validation, a majority vote of over 50% is required for the transaction to be approved; otherwise, it is rejected.

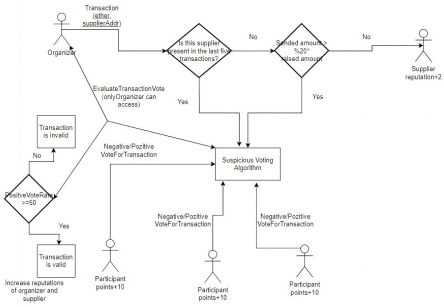

Figure 2: Suspicious Transaction Algorithm

2.4.5 Timeout Voting: Timeout Voting is related to the deadline set by organizers at the beginning of the campaign. After this deadline, any

participant can initiate a vote to reclaim the donated amount. If more than 70% of the votes are in favor, 80% of the donated amount is refunded. Failure to meet this threshold extends the timeframe by two weeks.

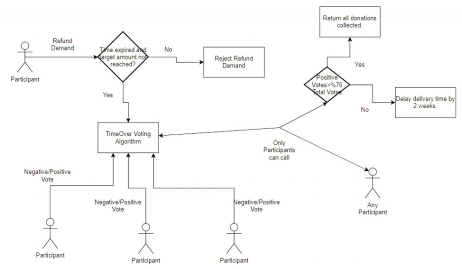

Figure 3: Time Over Voting and Refund Algorithm

2.4.6 DID registration: Considering the system's sensitivity to Sybil attacks due to voting processes, we have implemented DID registration to prevent such incidents. This approach enhances system security by verifying the identity of each node.

By leveraging smart contracts, our platform ensures transparency, security, and efficiency in crowdfunding operations.

2.5 Consensus Algorithm

Our platform utilizes a Proof of Stake (PoS) consensus algorithm, which is particularly well-suited for our permissionless blockchain network. PoS offers several key advantages:

2.5.1 Energy Efficiency

PoS is much more energy-efficient compared to Proof of Work (PoW) as it does not require solving complex cryptographic puzzles. Validators are chosen based on the amount of cryptocurrency they hold and are willing to "stake" as collateral [12].

2.5.2 High Throughput

PoS can handle a large number of transactions per second, making it ideal for high-performance applications. The absence of intensive computational requirements allows for faster transaction processing.

2.5.3 Decentralization and Security

PoS reduces the risk of centralization as it does not rely on expensive hardware for mining. Validators are incentivized to act honestly because they risk losing their staked cryptocurrency if they validate fraudulent transactions.

2.6 Security Solutions

Our platform addresses various security issues identified in the proposal

2.6.1 Against Malicious Organizers: In cases where a malicious organizer tries to gain unfair profits by creating fake campaigns, blockchain technology offers a solution: Each suspicious expenditure is subject to a vote by confirmators. Transactions need enough votes to be validated. If not, they are invalid. This process shifts decision-making from the organizer to the community, ensuring a more secure and democratic system.

2.6.2 Against Malicious Participants: To prevent a malicious participant from manipulating the voting system by creating multiple accounts, the following measures can be implemented: Ensuring each donor remains independent prevents manipulation. Participants register using unique cryptographic keys to protect personal information and maintain system integrity. Anyone can register independently without needing permission, ensuring each member is distinct.

2.6.3 Time-over Voting Problem and Identity Privacy: To address time-over voting and ensure identity privacy in campaigns:Each campaign development stage has a set time limit. Users can request to reclaim their donations after the time limit passes. Any participant can initiate a vote on reclaiming donations. If enough votes are collected, users can reclaim their funds. If not, the campaign continues to collect donations. Users' private information remains confidential, eliminating identity concerns. This blockchain-facilitated approach ensures that authority is not held by a single entity, promoting democratic decision-making.

2.6.4 Continuous Demand and Voting Problem:

Presenting all actions for voting and waiting for everyone to complete the voting process can be inefficient. To address this, we propose a two-pronged approach: Transactions involving cryptocurrency amounts above a certain threshold should be presented for voting. If transactions have been previously sent to similar individuals, they should also be presented for voting. Other transactions can be completed without requiring a voting process. By utilizing the transparency of blockchain transactions, this approach aims to: Focus voting on significant transactions to avoid delays and engage the community in crucial decisions while streamlining the overall process.

2.7 Security Issues in the Proposal and Solutions

In our proposed system, measures against malicious organizers focus on the management of collected funds. We aim to prevent the organizer from misusing the funds for personal gain. This can occur in two ways: by transferring the funds directly to their account or indirectly through another account.

In our system, collected donations do not go to the manager's account as in traditional crowdfunding processes. Donations are held at the campaign contract's

address. The organizer can still send large amounts of money to the supplier for personal gain. To prevent this, we implemented suspicious transaction voting, giving the process a democratic structure. Reputation scores help other users learn about the campaign manager's past successes.

Malicious participants may attempt Sybil attacks to manipulate democratic decision-making processes. We use a DID (Decentralized Identity) registration system to prevent this, which helps mitigate potential Sybil attacks.

Although not explicitly mentioned in our proposal, the participation of suppliers in the network also faces various security issues. There is no mechanism to verify if a new supplier is genuinely a supplier. Therefore, suppliers are given reputation scores, which are evaluated for network acceptance. A voting process is initiated to ensure democracy for candidate suppliers with low reputation scores.

We mentioned some problems related to non-innovative campaigns [13]. We conducted research suggesting that trend analysis and patent data tracking could solve this issue. However, we need more research to learn the use of hybrid and multi-hybrid systems with blockchain to effectively address this problem.

We addressed the problem of donations being held up due to project delays with a timeout voting mechanism. If the target amount is not reached after the specified time, participants can initiate a timeout vote. This allows participants to reclaim a portion of their donations at the end of the process.

While we aim for democratic processes and facilitate them through various votes, continuous voting can slow down crowdfunding processes and increase gas costs. To mitigate this, we should reduce the number of votes but maintain their functionality. For example, supplier network acceptance voting is only applied to those with low scores, higher thresholds can be set for suspicious transactions, and specific restrictions are placed on timeout voting.

All these measures ensure the stability of our system and address various problems from different angles.

3 Implementation

The implementation of our project is the result of the integration of five different smart contracts. These are named Campaign, Crowdfunding, DIDRegistry, SupplierManager, and VoteManager. These contracts handle the necessary features for campaign processes, fundraising, identity management, supplier verification, and community voting, ensuring the management and sustainability of the project.

Through smart contracts, we automate the needs related to fundraising and management, reduce the need for intermediaries, and minimize the risk of fraud.

Each contract is designed for a specific purpose, and their integration brings out the main functionality of our project.

Here is an overview of the explanations and applications of the five different contracts we have:

3.1 Campaign Contract

The campaign contract is designed to manage crowdfunding campaigns. Each campaign can be overseen by a manager who can create spending requests and monitor the payment and approval of funds.

3.1.1 Manager Role: The manager is the creator of the campaign, with the right to create spending requests and terminate campaigns based on various conditions.

3.1.2 Minimum Contribution: This is the minimum contribution amount each participant must meet to participate in the campaign and have voting rights.

3.1.3 Contribute: Accepts donations from participants who pay the minimum contribution or more, and holds the funds at the campaign address. This money is transferred to the supplier when the transfer function in the campaign contract is called.

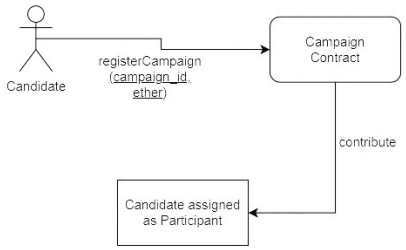

Figure 4: Registration and Contribution Diagram

3.1.4 Contribution Management: This allows users to participate in the campaign and grants them certain rights.

3.1.5 Request Creation: After the campaign funds are collected, the manager creates requests for spending the funds. Each request includes a description, the amount to be spent, the recipient, and an approval mechanism.

3.1.6 Request Finalization: The manager can transfer the desired amount to the recipient and mark the request as completed.

3.1.7 Time-Limited Voting and Refund: In the event of a timeout vote, if sufficient approval votes are not received within a certain period, participants can reclaim their donations at the end of the voting period. This mechanism ensures that donors can get their investments back if the campaign fails.

Security:

- **Restricted Access:** Restricted Access: Grants certain actions important for the project only to the project manager and restricts other users. Access to the funds in the campaign is managed by the system and access permission is restricted, thereby increasing transparency and reliability.
- **Refund Mechanism:** If sufficient votes are not obtained as a result of the timeout vote, there is a secure refund mechanism that allows participants to retrieve their contributions. This mechanism is initiated by the timeOver voting within the crowdfunding contract and then completed by the refunding within the campaign contract.

3.2 Crowdfunding Contract

This contract is designed to create and manage multiple crowdfunding campaigns. It keeps a list of all created campaigns and offers functions to create new campaigns and retrieve existing ones.

3.2.1 Campaign Creation: This function allows the creation of a campaign. It specifies the minimum amount required for the new campaign and distributes the instance of the created campaign and stores its address.

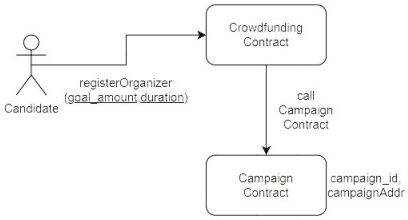

Figure 5: Organizer Registration and Campaign Creation Diagram

3.2.2 Campaign Retrieval: This function returns a list of all campaign addresses created through the transaction. It provides a way to access these campaigns.

3.2.3 Approval Process: Investors with voting rights vote to approve or disapprove the request. If the request receives more than 50% approvals, it can be finalized.

3.2.4 Vote Function: Participants can vote to approve or reject spending requests for the campaign. This function records participants' votes and initiates the voting process.

3.2.5 Evaluate Vote: This function determines the steps to be taken based on the voting results. If the request receives sufficient approval, it finalizes the spending request and transfers the funds to the appropriate recipient.

3.2.6 Suspicious Activity Detection: The detection of suspicious activities and the tracking of transaction movements end by transferring the appropriate ETH amount within this contract. Potentially suspicious activities are detected using specific rules and algorithms, and appropriate measures are taken.

3.2.7 Time-Limited Voting: The timeout voting is managed and initiated within this contract. If sufficient votes are not received within a certain period, participants can reclaim their donations at the end of the voting period.

Security:

- **Restricted Access:** Allows only authorized users to perform important actions and restricts other users.
- **Approval Checks:** Gives each participant only one vote right and distinguishes between participants with voting rights and others.
- **Refund Mechanism:** If sufficient votes are obtained as a result of the timeout vote, there is a secure refund mechanism that allows participants to retrieve their contributions. This mechanism is initiated by the timeOver voting within the crowdfunding contract and then completed by the refunding within the campaign contract.

This contract facilitates the organization and management of instances created by the Campaign Contracts. This way, investors can more easily access the campaigns they are interested in. It also includes mechanisms that allow investors to evaluate spending requests by voting and manage funds based on the voting results.

3.3 DIDRegistry Contract

This contract manages decentralized identifiers for users. It allows users to register and obtain their unique DIDs on the blockchain.

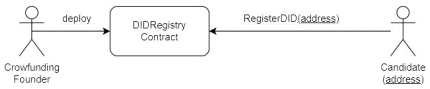

Figure 6: DID Initialization and Registration

3.3.1 registerDID: This function allows a user to register their DID (Decentralized Identifier). The user inputs the DID value as a parameter, and this value is associated with the user. Additionally, the function emits a message via DIDRegistered, indicating the completion of the registration process.

3.3.2 getDID: This function is used to retrieve a specific user's DID. When the user's address is entered as a parameter, the function returns the DID associated with that address, providing the user's unique identifier.

Security:

- Each address is allowed to register and update only its own DID information.
- Emits an informative call to ensure transparency and record information.
- Each user in the system has a unique public key associated with their address. This public key is used to verify the user's identity when registering or retrieving their DID, ensuring authenticity through Public Key infrastructure.
- Users are responsible for securely storing their private keys, which are used to sign transactions. This means that the security of the private key is crucial because it ensures that only the legitimate owner can register or update their DID, provided by Private Key infrastructure.
- When a user registers their DID, they sign the transaction with their private key. This digital signature guarantees the authenticity of the registration request, preventing unauthorized individuals from registering DIDs on behalf of others, ensured by Digital Signatures.
- Since the registration and management of DIDs are decentralized, there is no need for a central authority. This decentralization enhances security by distributing the control and verification processes across the blockchain network.

3.4 SupplierManager Contract

This contract manages supplier information by evaluating reputation scores and voting statuses. It ensures the storage of supplier information, updates votes, and supports supplier approval.

3.4.1 registerSupplier: This function registers a supplier for a created campaign. This function registers a supplier for a created campaign. The function updates the supplier's information and, if the supplier's reputation is below a certain threshold (20), marks the supplier as a candidate. Successful voting ensures the supplier's acceptance into the network.

3.4.2 updateVote: This function updates the vote status of a supplier. The function verifies that the voting process has been initiated and then marks the supplier as voted once the voting process is completed.

3.4.3 getSupplier: This function retrieves supplier information for a specific campaign. The user inputs campaignId and supplierAddr parameters. The function returns the information related to the specified supplier.

3.4.4 confirmSupplier: This function allows for supplier approval. The function removes the candidate tag from the supplier, thereby approving them. After the approval process, the supplier is automatically accepted by the system through smart contracts and can perform their duties.

3.4.5 incrementReputationSupplier: This function increases a supplier's reputation.

Security:

- Checks are performed to ensure that only valid actions are executed; for example, voting is only allowed if the voting process has been initiated.
- Updates are allowed only for existing suppliers.

3.5 VoteManager Contract

This contract manages the voting processes required for campaigns. It includes functionalities for voting, detecting suspicious activities, and handling timeout issues.

3.5.1 startVote: This function initiates the voting process for a specific election (e.g., Supplier, Transaction, Timeout).

3.5.2 castVote: This function allows users to vote for a specific election. If the vote is positive, it increases the count of positive votes, and if negative, it increases the count of negative votes.

3.5.3 getVote: This function retrieves vote details for a specific election. The function returns the Vote structure for the specified election.

3.5.4 updateVote: This function updates the vote status for a specific election. The function marks the candidacy status for the given election as false.

Security:

- Ensures only valid votes are allowed and voting is permitted only for initiated elections.
- Uses keccak256 method to securely manage string comparisons.
- Properly manages state changes and access control to prevent unauthorized updates.

These contracts collectively manage different aspects of a decentralized crowdfunding platform, ensuring transparency and security through blockchain technology. DIDRegistry handles identity management, SupplierManager oversees supplier-related activities, and VoteManager supervises various voting processes essential for maintaining trust and accountability in the system.

References

[1] Cole, Rosanna, Mark Stevenson, and James Aitken. "Blockchain technology: implications for operations and supply chain management." Supply chain management: An international journal 24.4 (2019): 469-483.

[2] Hartmann, Felix, et al. "Alternative fundraising: success factors for blockchain-based vs. conventional crowdfunding." 2019 IEEE international workshop on blockchain oriented software engineering (IWBOSE). IEEE, 2019.

[3] Sarmah, Simanta Shekhar. "Understanding blockchain technology." Computer Science and Engineering 8.2 (2018): 23-29.

[4] Saleh, Fahad. "Blockchain without waste: Proof-of-stake." The Review of financial studies 34.3 (2021): 1156-1190.

[5] kh Saad, S. M., & Raja Mohd Radzi, R. Z. (2020). Comparative Review of the Blockchain Consensus Algorithm Between Proof of Stake (POS) and Delegated Proof of Stake (DPOS). International Journal of Innovative Computing, 10(2). https://doi.org/10.11113/ijic.v10n2.272

[6] O. Onireti, L. Zhang and M. A. Imran, "On the Viable Area of Wireless Practical Byzantine Fault Tolerance (PBFT) Blockchain Networks," 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 2019, pp. 1-6, doi: 10.1109/GLOBECOM38437.2019.9013778.

[7] Manuel Adelin Manolache, Sergiu Manolache, Nicolae Tapus, Decision Making using the Blockchain Proof of Authority Consensus,Procedia Computer Science, Volume 199, 2022, Pages 580-588, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.01.071.

[8] Taherdoost, H. Smart Contracts in Blockchain Technology: A Critical Review. Information 2023, 14, 117. https://doi.org/10.3390/info14020117

[9] N. Yadav and S. V., "Venturing Crowdfunding using Smart Contracts in Blockchain," 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2020, pp. 192-197, doi: 10.1109/ICSSIT48917.2020.9214295.

[10] Mukhopadhyay, Mayukh. Ethereum Smart Contract Development: Build blockchain-based decentralized applications using solidity. Packt Publishing Ltd, 2018.

[11] Maram, Deepak, et al. "Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability." 2021 IEEE Symposium on Security and Privacy (SP). IEEE, 2021.

[12] Nguyen, Cong T., et al. "Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities." IEEE access 7 (2019): 85727-85745.

[13] Wustmans,Michael; Haubold,Thomas; Bruens,Bennet. "Bridging Trends and Patents: Combining Different Data Sources for the Evaluation of Innovation Fields in Blockchain Technology". June 2022. 10.1109/TEM.2020.3043478. 18. 04. 2024.