Visvesvaraya Technological University, Belagavi – 590018

MINI-PROJECT REPORT
ON

BLOCK-FUND : Blockchain based crowdfunding application

Submitted in partial fulfillment for the award of degree of

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE & ENGINEERING

Submitted by

Akhil Shetty M	**4SO21CS013**
Allan D'Souza	**4SO21CS017**
Kruthag Rai	**4SO20CS077**
Arvin D'Silva	**4SO21CS025**

Under the Guidance of

Ms. L Hema
Assistant Professor, Department of CSE

DEPT. OF COMPUTER SCIENCE AND ENGINEERING
ST JOSEPH ENGINEERING COLLEGE
An Autonomous Institution
(Affiliated to VTU Belagavi, Recognized by AICTE, Accredited by NBA)
Vamanjoor, Mangaluru - 575028, Karnataka

2023-24

ST JOSEPH ENGINEERING COLLEGE

An Autonomous Institution

(Affiliated to VTU Belagavi, Recognized by AICTE, Accredited by NBA)

Vamanjoor, Mangaluru - 575028, Karnataka

DEPT. OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Mini-project work entitled **"BLOCK-FUND : Blockchain based crowdfunding application"** carried out by

Akhil Shetty M	**4SO21CS013**
Allan D'Souza	**4SO21CS017**
Kruthag Rai	**4SO20CS077**
Arvin D'Silva	**4SO21CS025**

the bonafide students of VI semester Computer Science & Engineering in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2023-2024. It is certified that all suggestions indicated during internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Ms. L Hema
Mini-Project Coordinator

Dr Sridevi Saralaya
HOD-CSE

Abstract

A blockchain based BLOCK-FUND system is an innovative blockchain-based crowdfunding platform that leverages Ethereum technology to provide a secure and transparent environment for campaign creation and funding. Users can seamlessly interact with the platform by connecting their MetaMask wallet, creating new projects, and funding existing ones. The integration of blockchain technology ensures the integrity and transparency of transactions, making it a reliable tool for both project creators and contributors.

This report outlines the technical implementation of BLOCK-FUND, from the development of smart contracts using Solidity and Truffle to the creation of a user-friendly frontend with React. By incorporating a comprehensive approach to both backend and frontend development, BLOCK-FUND aims to deliver a robust platform that not only meets current crowdfunding needs but also sets a foundation for future enhancements in blockchain-based financial solutions.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Background

BLOCK-FUND is designed to address the growing need for transparent and secure crowdfunding solutions. The platform utilizes Ethereum's blockchain to manage and record transactions, ensuring that all contributions and project details are immutable and publicly verifiable. The frontend, built with React, offers an intuitive interface for users to interact with the platform, whether they are project creators or backers. By connecting their MetaMask wallets, users can effortlessly engage in the crowdfunding process, from launching new campaigns to funding existing ones.

The platform offers a seamless experience from landing on the homepage to creating and funding projects. Upon entering the site, users are greeted with a clean and navigable interface. They can quickly log in, connect their MetaMask account, and access various features including viewing all projects, creating new campaigns, and tracking funded projects. This streamlined process ensures that both project creators and backers have a smooth and efficient experience.

1.2 Problem statement

Traditional crowdfunding platforms often face issues related to transparency, security, and the trustworthiness of financial transactions. Many platforms do not offer verifiable proof of contributions or the authenticity of project details, leading to potential disputes and lack of trust. BLOCK-FUND aims to address these issues by leveraging blockchain technology to ensure that all transactions and project data are immutable and transparent, thereby enhancing the overall reliability of the crowdfunding process.

1.3 Scope

BLOCK-FUND aims to revolutionize the crowdfunding landscape by offering a blockchain-based solution that provides transparency and security. The project's scope includes the development of a user-friendly frontend in React, the implementation of smart contracts in Solidity for campaign management, and integration with MetaMask for secure transactions. Achievements will include a functional platform that supports campaign creation, funding, and management, along with the ability to verify transactions on the blockchain, ensuring both transparency and trust.

In addition to these core features, BLOCK-FUND will focus on enhancing user engagement through a responsive and intuitive interface that simplifies the process of discovering and supporting projects. The platform will also incorporate advanced features such as real-time updates and automated notifications to keep users informed about campaign progress and funding milestones. By leveraging blockchain technology, BLOCK-FUND will not only address the inherent limitations of traditional crowdfunding models but also set a new standard for security, efficiency, and user empowerment in the digital fundraising space.

Chapter 2

Software Requirements Specification

2.1 Introduction

This Software Requirements Specification (SRS) document outlines the requirements for the blockchain-based BLOCK-FUND crowdfunding platform. The system is designed to enhance the crowdfunding experience by providing a secure, transparent, and decentralized environment for campaign creation and funding using blockchain technology. By integrating smart contracts and leveraging Ethereum's capabilities, BLOCK-FUND ensures the integrity and transparency of all transactions, offering a reliable platform for both project creators and contributors.

The BLOCK-FUND platform aims to address the limitations of traditional crowdfunding models by enabling users to create, manage, and fund campaigns while maintaining a high level of security and trust through decentralized verification [**antonopoulos2018mastering**]. This document details the functional and non-functional requirements, user interface design, and performance criteria essential for the project's success. It is intended for use by developers and stakeholders to ensure the system aligns with its objectives and meets its constraints.

Key resources for this project include official documentation and best practices for Ethereum development [**ethereumdocs**], React [**reactdocs**], and MetaMask [**metamaskdocs**], as well as related technologies.

2.2 Functional Requirements

Functional requirements define the system's expected behaviors and interactions. They outline the essential functions and features that the system must support.

1. **User Interactions:**
    - **Login/Logout:** Users can log in or out of their accounts using MetaMask.
    - **Management:** Users can create, view, and manage projects.

- **Funding:** Users can fund existing projects by specifying the amount of ETH.

2. **System Operations:**
   - **Campaign Creation:** Ability to set up new campaigns with title, description, duration, and funding goals.
   - **Transaction Processing:** Handling of ETH transfers and project funding.
3. **Error Handling**:
   - **Transaction Errors:**: Display appropriate error messages for failed transactions.
   - **Connection Issues**: Notify users if there are issues connecting to MetaMask.
4. **Integration Points**:
   - **MetaMask Integration**: Seamless connection and interaction with MetaMask for wallet management.
   - **Blockchain Integration**: Interaction with Ethereum blockchain for storing and retrieving project data.

2.3 Non-Functional Requirements

Non-functional requirements describe the quality attributes, performance standards, and other system constraints that are not related to specific functionalities but are crucial for the system's overall effectiveness and user satisfaction.

1. **Performance:** The platform should handle multiple concurrent users and transactions without significant delays.
2. **Scalability:** The system must be able to scale with increasing numbers of users and projects.
3. **Security:** Transactions and user data must be secured through encryption and blockchain technology.
4. **Reliability:** The platform should maintain high availability and minimal downtime.
5. **Compatibility:** The BLOCK-FUND must be compatible with all major web browsers, including Chrome, Firefox, and Edge.

2.4 User Interface Requirements

The user interface must be intuitive and responsive, providing a seamless experience across different devices. Key aspects include clear navigation, accessible features, and a visually appealing design that enhances user engagement.

2.4.1 Design Principles

The user interface should adhere to modern design principles to enhance usability and user satisfaction.

- **Consistency:** The UI should maintain consistency in design elements across all screens, including fonts, colors, and button styles.
- **Simplicity:** The design should be clean and clutter-free, emphasizing essential functions to avoid overwhelming the user.
- **Clarity:** Information should be presented clearly, avoiding ambiguity and confusion.

2.4.2 Accessibility

The application should be accessible to all users, including those with disabilities, by adhering to accessibility standards.

- **Consistent Navigation:** Design the application's navigation to be consistent across all pages, allowing users to predictably move through the application without confusion.
- **Flexible Input Methods:** Support multiple input methods, such as touch, keyboard, and mouse, to cater to different user preferences and abilities.
- **Color Contrast:** Ensure sufficient contrast between text and background colors to improve readability for users with visual impairments.

2.4.3 User Feedback

Feedback mechanisms should be implemented to gather user input and improve the application over time.

- **Error Messages:** Provide clear and informative error messages to guide users in resolving issues.
- **User Surveys:** Periodic surveys can be conducted to collect user feedback on the UI and identify areas for improvement.

2.5 Software Requirements

This section outlines the technical requirements for the BLOCK-FUND system, addressing hardware, software, network, and operational needs to ensure efficient and reliable performance.

2.5.1 Hardware Requirements

The BLOCK-FUND system requires specific hardware configurations to deliver optimal performance and support user needs.

- **Server Specifications:** Requires a minimum of 16 GB RAM, a 4-core CPU, and a 500 GB SSD.
- **User Devices:** Devices must have at least 4 GB RAM and a modern web browser for optimal functionality.

2.5.2 Software Dependencies

The system relies on several software components and libraries for its functionality.

- **Operating Systems:** Compatible with Windows Server 2019 and macOS Mojave (10.14).
- **Database Systems:** Utilizes Node.js for data management.
- **Tech Stack:** Developed with HTML, CSS, JavaScript, and Solidity, utilizing the React library for building user interfaces.

Chapter 3

System Design

3.1 Architecture Design

Figure 3.1 The high-level architecture of the BLOCK-FUND crowdfunding platform integrates several key components to deliver a seamless user experience. The user interface, developed with React, allows users to create and fund campaigns interactively. Blockchain smart contracts, deployed on the Ethereum network, manage campaign metadata and validate funding transactions, ensuring transparency and security. The storage backend, powered by Firebase, handles the storage and retrieval of campaign-related data, including project details and contributor information. This architecture combines a user-friendly interface with robust blockchain technology and reliable data storage, providing an efficient and trustworthy crowdfunding experience.

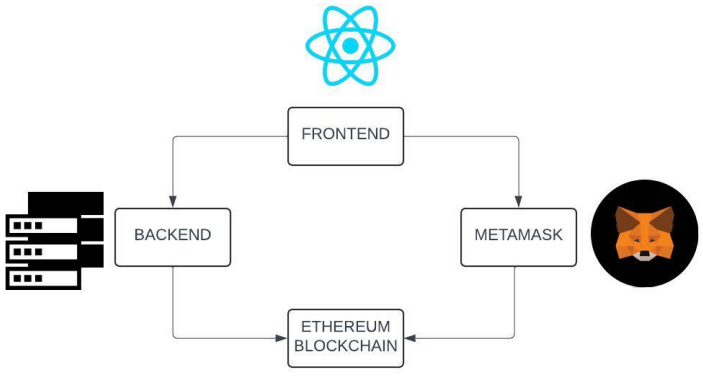

Figure 3.1: System Architecture Diagram of BLOCK-FUND

3.2 Decomposition Description

Figure 3.2 presents a decomposition diagram of the BLOCK-FUND crowdfunding platform, detailing its core modules: user interface, user authentication, campaign man-

agement, blockchain integration, and data retrieval. Each module is essential for the platform's operation:

- **User Interface:** The web-based interface where users perform activities such as registration, campaign creation, funding, and browsing. It provides an intuitive and responsive experience.
- **User Authentication:** Manages user accounts, including registration, login, and verification, ensuring secure and authorized access.
- **Campaign Management:** Oversees the creation, management, and status of campaigns, including project details and funding goals.
- **Blockchain Integration:** Interfaces with the Ethereum blockchain to manage smart contracts, ensuring secure and transparent transactions.
- **Data Retrieval Service:** Retrieves and displays campaign data and funding information, keeping users informed with up-to-date details.

This structure ensures an organized and efficient platform, combining user-friendly design, secure authentication, effective campaign management, and reliable blockchain integration.

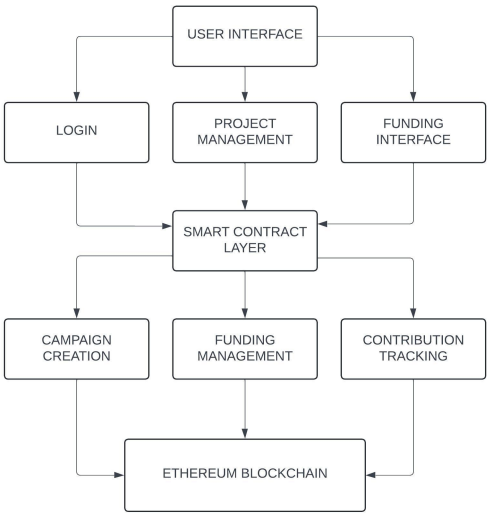

Figure 3.2: Decomposition Diagram of BLOCK-FUND

3.3 Data Flow Design

Figure 3.3 depicts the data flow within the BLOCK-FUND crowdfunding platform. The diagram illustrates the process from campaign creation to funding and retrieval. When a user creates a campaign, the campaign details are first submitted through the user interface and then stored in the platform's backend. The campaign data, including its details and funding goals, is simultaneously sent to the Ethereum smart contract for secure storage and verification on the blockchain. Upon successful creation, the platform updates the user interface to display the new campaign and its funding status. During funding and retrieval, the platform queries both the backend and the blockchain to provide users with up-to-date information on campaign progress and contributions.

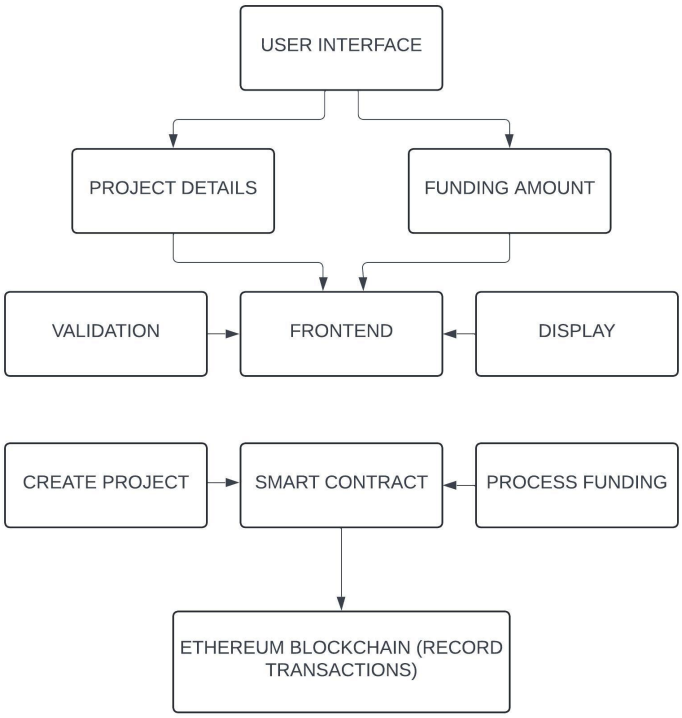

Figure 3.3: Dataflow Diagram of BLOCK-FUND

Chapter 4

Implementation

4.1 System Overview

BLOCK-FUND is a blockchain-based crowdfunding platform designed to facilitate project creation and funding securely. The core components include:

1. **Frontend:** A React application[**reactdocs**] for user interaction created using Node Package Manager(npm)[**npmdocs**].
2. **Backend:** Managed by Node.js to handle API requests and interactions with the Ethereum blockchain.
3. **Blockchain:** Ethereum is used for secure transaction management and smart contract execution.
4. **Smart Contract:** Smart contract written using Solidity language[**soliditydocs**] and migrated to the Ethereum network[**ethereumdocs**] using Truffle[**truffledocs**].

4.2 Algorithms

4.2.1 Campaign Creation Algorithm

1. **Input Data:** Collect title, description, duration, and funding goal from the user.
2. **Validation:** Ensure all fields are filled and the funding goal is positive.
3. **Transaction Creation:** Create a transaction to deploy the new campaign on the Ethereum blockchain.
4. **Confirmation:** Prompt the user to confirm the creation of the campaign.
5. **Completion:** Upon confirmation, finalize the campaign creation and store details on the blockchain.
6. **Error Handling:** Display appropriate error messages if the upload fails.

4.2.2 Project Funding Algorithm

1. **Input Data:** Collect the amount of ETH to be contributed from the user.
2. **Validation:** Ensure the amount is positive and sufficient funds are available.
3. **Transaction Creation:** Create a transaction to fund the selected project on the Ethereum blockchain.
4. **Confirmation:** Prompt the user to confirm the funding transaction.
5. **Completion:** Upon confirmation, process the transaction and update the project's funding status.

4.3 Key Components Implementation

4.3.1 Smart Contract

The BLOCK-FUND smart contract[**antonopoulos2018mastering**], written in Solidity and compiled and migrated using Truffle[**truffledocs**], is integral to the BLOCK-FUND application. The Crowdfunding.sol smart contract manages project creation, funding, and retrieval. It includes functions for creating new projects, funding existing ones, and retrieving project details. The contract ensures all data is securely stored and immutable on the Ethereum blockchain.

```
    // SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract CrowdFunding {
    // List of existing projects
    Project[] private projects;

    // Event that will be emitted whenever a new project is started
    event ProjectStarted(
        address contractAddress,
        address projectStarter,
        string projectTitle,
        string projectDesc,
        uint256 deadline,
        uint256 goalAmount
    );

  .
  .
  .
```

```
    .
    .
    }
    function getInfo()
        public
        view
        returns (
            address payable projectStarter,
            string memory projectTitle,
            string memory projectDesc,
            uint256 deadline,
            State currentState,
            uint256 currentAmount,
            uint256 goalAmount
        )
    {
        projectStarter = creator;
        projectTitle = title;
        projectDesc = description;
        deadline = raiseBy;
        currentState = state;
        currentAmount = currentBalance;
        goalAmount = amountGoal;
    }
}
```

Listing 4.1: Solidity Smart Contract (Crowdfunding.sol)

4.3.2 Frontend (React)

The frontend, built with React, includes components for user registration, project creation, and funding. Key components include:

1. **Login Component:** Handles user login.
2. **Display Campaigns:** Shows a list of available campaigns with options to view details or fund them.
3. **Campaign Details:** Displays detailed information about a specific campaign, including funding status and backers.
4. **Custom Button:** A reusable button component for various actions like creating campaigns and funding projects.

4.3.3 Backend (Blockchain)

1. **Database:** Saves metadata about the files.
2. **Ethereum Blockchain:** Stores the file hashes for verification.

4.4 Integration

In the BLOCK-FUND crowdfunding platform, the frontend components interact seamlessly with the Ethereum blockchain to offer a secure and intuitive user experience. User authentication and interactions are managed through MetaMask, ensuring that users are securely signed in and their transactions are handled safely. Campaign creation and funding processes are directly executed on the Ethereum blockchain via smart contracts, which store and manage all campaign data, including project details and funding status. The integration with Ethereum guarantees that all campaign data is immutable and verifiable, maintaining the integrity and transparency of each transaction. This setup ensures that the BLOCK-FUND platform is both secure and scalable, providing a reliable solution for managing crowdfunding projects and contributions.

Chapter 5

Results and Discussion

5.1 Results

After successfully implementing BLOCK-FUND, our blockchain-based crowdfunding application, we conducted a series of tests to evaluate its performance and functionality. Below are the key results from these tests:

1. **User Authentication and Verification:**
    - **Result:** The application successfully authenticated users via MetaMask.
    - **Discussion:** The integration of MetaMask ensured a smooth and secure sign-in process. Users could only interact with the platform once authenticated, ensuring the integrity of user interactions.
2. **Campaign Creation and Management:**
    - **Result:** Campaigns were created and managed effectively on the Ethereum blockchain.
    - **Discussion:** The smart contracts handled all campaign data securely. Campaign details, including funding goals and contributions, were accurately recorded on the blockchain, providing transparency and immutability.
3. **Blockchain Integration:**
    - **Result:** Campaign data and transactions were successfully recorded on the Ethereum blockchain.
    - **Discussion:** Storing campaign information on the blockchain provided a high level of security and allowed users to verify the authenticity and progress of campaigns through blockchain transactions.
4. **Performance Metrics:**

- **Result:** The system handled concurrent campaign creations and funding transactions efficiently.
- **Discussion:** The application maintained stable performance under various load conditions. The integration of blockchain technology did not significantly impact the responsiveness of the application.

5.2 Discussion

The development of BLOCK-FUND has demonstrated significant advancements in creating a secure and efficient crowdfunding platform using blockchain technology. Several key aspects were highlighted during the development and testing phases:

- **Security:** The integration of MetaMask for user authentication provided a robust mechanism for verifying user identities. Combining this with blockchain for storing campaign data and transactions enhanced security by ensuring the integrity and ownership of campaign information, making it difficult for unauthorized users to alter or manipulate the data.
- **Scalability:** The architecture of BLOCK-FUND is designed for scalability. The Ethereum blockchain handles an increasing number of campaigns and transactions effectively. While blockchain transactions involve some delays due to network confirmations, the system scales well with growing user demand for transparent and immutable campaign data.
- **User Experience:** User feedback indicated a positive experience with the platform's interface and functionality. Users found the process of creating and funding campaigns intuitive and straightforward. The responsive design and clear instructions contributed to a smooth user journey.
- **Integration Challenges:** Integrating blockchain with frontend technologies posed some challenges, particularly in ensuring seamless interactions between the smart contracts and the user interface. These challenges were addressed through comprehensive testing and iterative development.

5.3 Screenshots

Below are screenshots of the BLOCK-FUND application demonstrating key features and user interface elements.

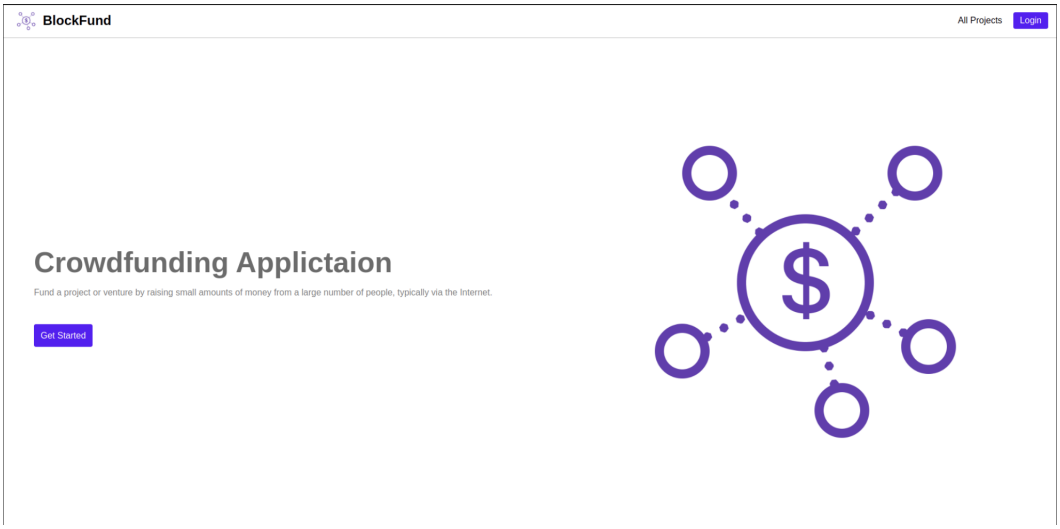

Figure 5.1: Home Page of BLOCK-FUND application

5.3.1 Home Page

Description: This screenshot displays the home page of the BLOCK-FUND application, showcasing the main dashboard where users can navigate to various sections of the platform.

5.3.2 Login Page

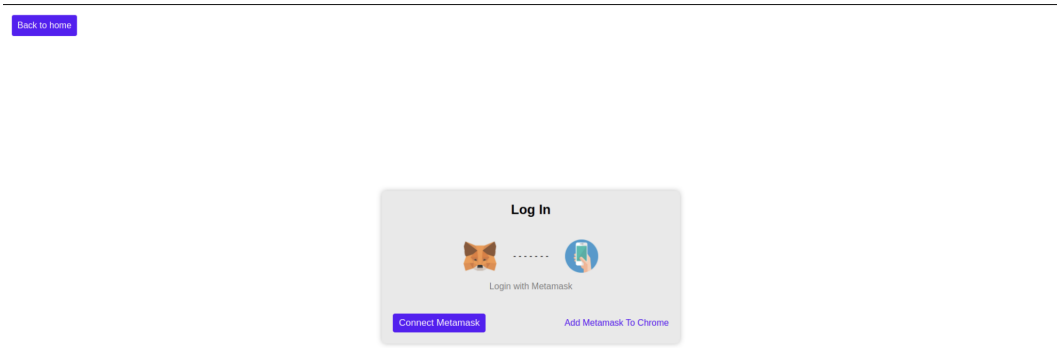

Figure 5.2: Login Page for existing users

Description: This screenshot shows the login page where registered users can enter their credentials to access their accounts.

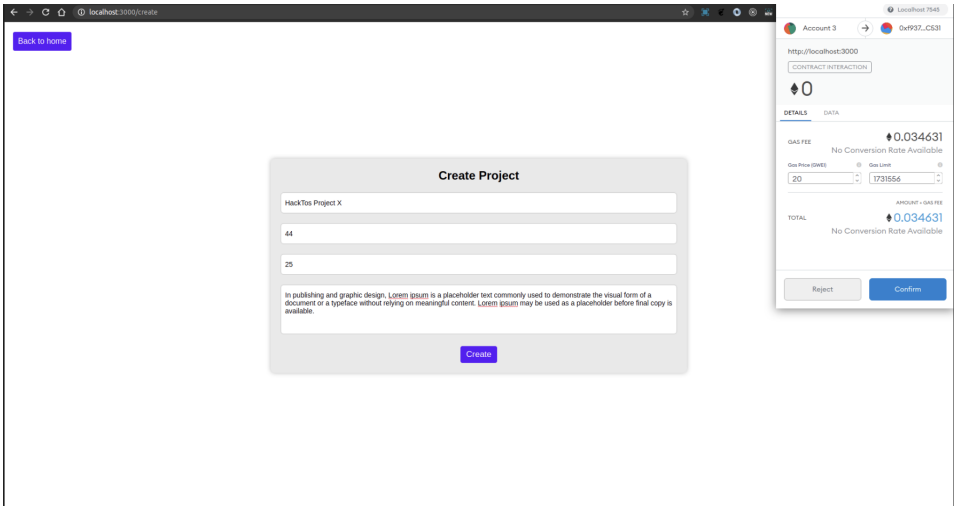

Figure 5.3: Creation of a new campaign

5.3.3 Create Campaign Page

Description: This screenshot demonstrates the page where users can create new crowdfunding campaigns by entering the required details.

5.3.4 Fund Campaigns Page

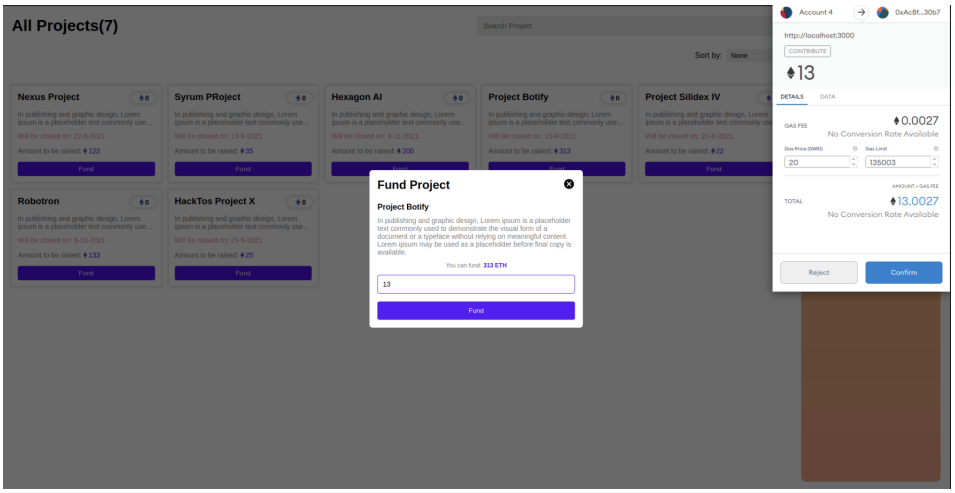

Figure 5.4: Campaigns available for funding

Description: This screenshot displays the page listing various campaigns that users can fund.

5.3.5 Funded Campaigns Page

Description: This screenshot shows the page where users can view campaigns they have funded.

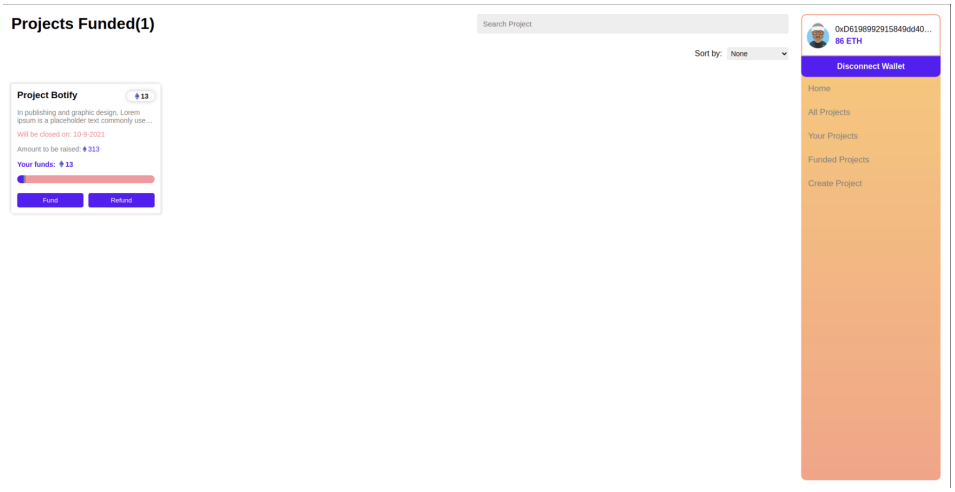

Figure 5.5: Campaigns that have been funded

5.3.6 All Projects Page

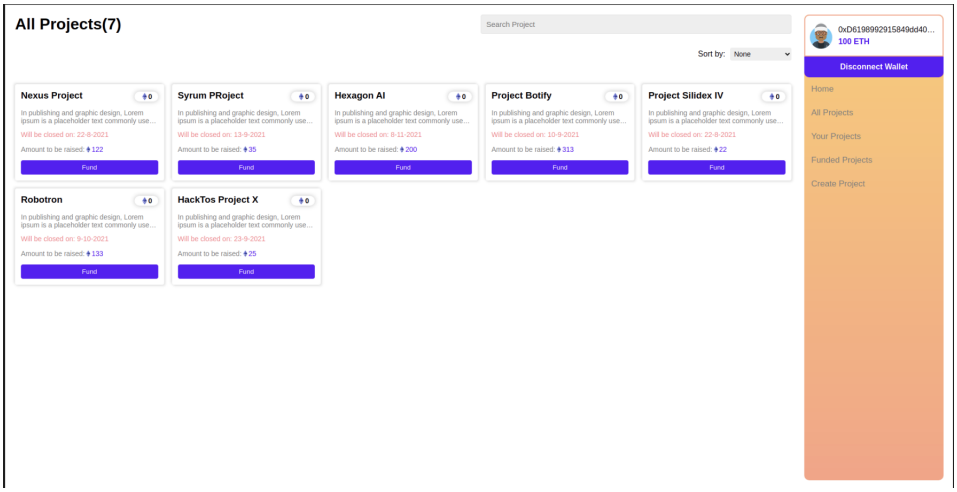

Figure 5.6: All created projects

Description: This screenshot presents the page listing all projects that have been created on the platform.

5.3.7 File Details Stored on Blockchain

Description: This screenshot shows how file details, including hashes, are stored as blocks on the Ethereum blockchain using Ganache.

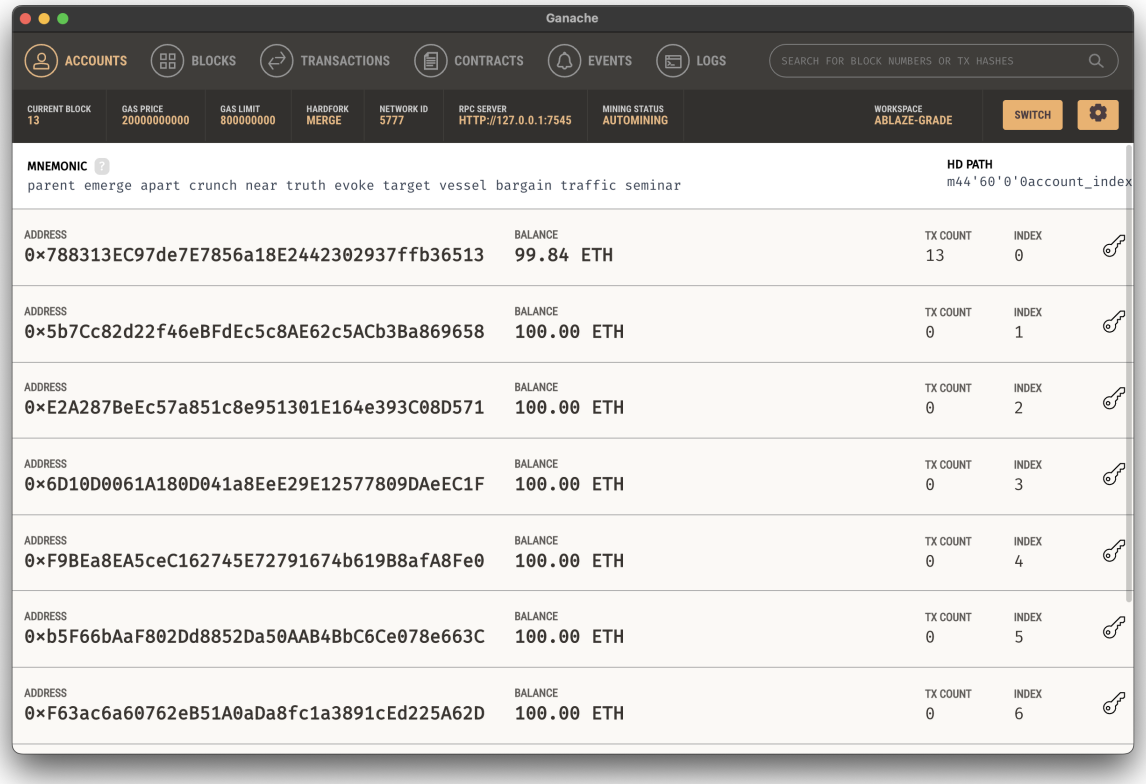

Figure 5.7: File details stored on blockchain in Ganache

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The BLOCK-FUND project successfully illustrates the integration of blockchain technology with crowdfunding solutions to create a secure and transparent platform for fundraising. Throughout the development process, we focused on key aspects such as campaign security, user authentication, and data integrity. By leveraging Ethereum's blockchain, we ensured that all campaign data, including transaction records and campaign details, are securely hashed and verifiable, preventing unauthorized alterations or data tampering. This approach not only enhances the security and transparency of the crowdfunding process but also fosters user trust by providing a decentralized and immutable system. The project's success highlights the effective synergy between blockchain technology and crowdfunding, establishing a solid foundation for future innovations in fundraising platforms.

6.2 Future Work

Although BLOCK-FUND meets its core objectives, there are several areas where further development and enhancement could be pursued. Future work may focus on the following aspects:

- **Enhanced Scalability:** As the user base and campaign volume grow, optimizing the system for better scalability will be essential. This may involve exploring more efficient blockchain solutions or layer-2 scaling techniques to handle increased traffic and transaction loads.
- **Mobile Application:** Developing a mobile version of BLOCK-FUND to extend accessibility and usability across various devices.
- **Advanced Security Features:** Integrating additional security measures such as multi-factor authentication (MFA), biometric verification.

- **Decentralized Storage Integration:** Exploring decentralized storage solutions such as IPFS (InterPlanetary File System) or Filecoin to further enhance data availability and redundancy.

By addressing these areas, BLOCK-FUND can evolve into a more comprehensive and versatile CrowdFunding solution, offering enhanced security, scalability, and user experience. The project's success serves as a testament to the potential of blockchain technology in revolutionizing secure data storage and management systems.

References

1. Andreas M. Antonopoulos and Gavin Wood. *Mastering Ethereum: Building Smart Contracts and DApps*. O'Reilly Media, 2018. ISBN: 978-1491971949.

2. Ethereum Foundation. *Ethereum Official Documentation*. 2024. Available online: `https://ethereum.org/en/developers/docs/`.

3. Firebase Core Team. *Firebase Documentation*. Google LLC. 2024. Available online: `https://firebase.google.com/docs`.

4. Node.js Core Team. *Node.js Official Documentation*. OpenJS Foundation. 2024. Available online: `https://nodejs.org/`.

5. npm, Inc. *npm Documentation*. 2024. Available online: `https://docs.npmjs.com/`.

6. React Core Team. *React Official Documentation*. Meta Platforms, Inc. 2024. Available online: `https://react.dev/`.

7. Solidity Contributors. *Solidity Documentation*. Ethereum Foundation. 2024. Available online: `https://soliditylang.org/docs/`.

8. Truffle Team. *Ganache Documentation*. Truffle Suite. 2024. Available online: `https://trufflesuite.com/ganache/`.

9. Truffle Team. *Truffle Suite Documentation*. Truffle Suite. 2024. Available online: `https://trufflesuite.com/docs/`.