الجمـــهوريـــة الجـــزائريـــة الديـمقـراطيـــــــة الشـــعبيـــــة
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزارة التعـلـيـم الـعـالـــي و الـبـحـث الـعـلـمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جـامـعـة عمّـــار ثـلـيـــجـــي بالأغـــواط
AMAR TELIDJI UNIVERSITY OF LAGHOUAT

كـلـيـــــة الـعـلـــــوم
FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Master's Thesis

Field : Mathematics and Computer Science
Specialty : Computer Science
Option : Information System and Decision-Making

By:

BENNAOUI Imad Eddine **LALMI Mohammed Abderahmane**

TOPIC

Integrating ERC-20 Tokens into a Blockchain-Based Charity System
A Decentralized Approach to Transparent Donations

Defended publicly on June 28,2025, before a jury composed of:

Pr. Mohamed Lahcen Bensaad	Prof	President
Dr. Messaoud Babaghayou	M.C.A	Examiner
Dr. Tarek Bouzid	M.C.A	Examiner
Pr. Noureddine Chaib	Prof	Supervisor

Thesis No. …… – Academic Year 2024/2025


بسم الله الرحمن الرحيم
١٤٣٨

DEDICATIONS

To my dear parents,
for all their sacrifices, their love, tenderness, support, and prayers throughout my studies.
To my beloved brothers,
for their constant encouragement and moral support.
To my entire family,
for their unwavering support throughout my academic journey.
May this work be the fulfillment of your deeply cherished wishes, and the result of your continued support. Thank you for always being there for me.
To my dear friend, brother and colleague,
for sharing this journey with me, for the collaboration, mutual support, and commitment. This accomplishment is as much yours as it is mine.
To my loyal friends,
who have stood by my side like brothers with their support, laughter, and presence through both the easy and difficult moments. Your friendship has been a source of strength, and I am deeply grateful to have shared this chapter of life with you.
To my Palestinian brothers,
to those who have been martyred and to those who continue to live with dignity and resistance this work and its idea are dedicated to you. May Allah grant you justice, peace, and liberation in the years to come. May better days come soon,
In shaa Allah.
" FREE PALESTINE "

BENNAOUI Imad Eddine

DEDICATIONS

To my dear parents,
for their endless love, sacrifices, and prayers that have shaped my path and given me the strength to persevere.
To my mother,
who was always after me to push forward and never give up—your constant encouragement and belief in me made this day possible.
To my father,
who guided us on the straight path with wisdom, faith, and patience. Your strength and values are forever my compass.
To my brothers and sister,
for the support, smiles, and motivation you gave me when I needed it the most. To my entire family,
for being the solid foundation upon which I have built this academic journey.
To my dear colleague and thesis partner,
thank you for walking beside me through this journey, for your dedication, your commitment, and your belief in the shared vision of our work. This achievement is as much yours as it is mine.
To my loyal friends,
for your unwavering support, your humor in times of stress, and your uplifting words. Your presence made the road less heavy and far more meaningful.
to our brothers and sisters in Gaza
hose who stand firm despite injustice, oppression, and pain—this work is also dedicated to you. May Allah protect you, elevate you, and bring you peace, dignity, and justice. May liberation be near, and may your sacrifices never be forgotten.
" FREE PALESTINE "

LALMI Mohamed Abderahmane

ACKNOWLEDGMENT

First and foremost, all praise and thanks are due to Allah, the Most Gracious, the Most Merciful, for granting us the strength, patience, and guidance to complete this thesis. Without His divine help, none of this would have been possible.

We express our deepest gratitude to our esteemed supervisor, Pr. Chaib Noureddine, for his valuable support, constructive advice, and continuous encouragement throughout this work. His guidance and trust played a major role in helping us grow both academically and personally.

Our sincere thanks also go to the jury members, who will evaluate our work, and who have also been our teachers and mentors throughout our academic journey. Each one of you has contributed, in one way or another, to shaping our understanding and knowledge. Thank you for your time, feedback, and dedication to your students.

We are also grateful to all the teachers of our department, whose efforts and commitment over the years have built the foundation of our education. A heartfelt thanks to each one of you, by name and memory, for every lesson shared and every piece of advice given.

A special mention to Dr. Badra Kerrouche, who, beyond her role as a teacher, provided us with additional help and support during the development of this thesis. Your kindness and availability meant a great deal to us.

We also extend our appreciation to the head of the department, whose leadership has ensured the smooth running of our academic experience, and to the department's secretaries, who always welcomed us with a smile and supported us with efficiency and patience.

Finally, our sincere thanks go to all the staff and workers in the department, especially the cleaning and maintenance staff, whose efforts behind the scenes help create a clean, safe, and functional environment for all students.

May Allah reward everyone who contributed, directly or indirectly, to the completion of this work.

Abstract

Traditional charity donation systems often suffer from a lack of transparency, high transaction fees, and donor distrust. Many donors are unable to verify how their funds are utilized, leading to concerns about fraud, inefficiency, and mismanagement. To address these challenges, this research explores the integration of ERC-20 tokens into a blockchain-based charity system, leveraging smart contracts to ensure secure, transparent, and auditable transactions.

The proposed system utilizes Ethereum-based smart contracts to automate donation processing, eliminating the need for intermediaries and reducing transaction costs. By implementing a decentralized application (dApp, a blockchain-powered application) with Web3 (a set of JavaScript libraries enabling interaction with the Ethereum blockchain), donors can track their contributions in real-time, ensuring that funds are only accessible to verified charities. The platform is developed using Solidity for smart contracts, Hardhat for blockchain testing, and React.js with Ethers.js for the user interface.

To evaluate the system's efficiency, we analyze gas fees, transaction speed, security risks, and user adoption challenges. Experimental results demonstrate that blockchain-based donations can significantly improve trust, transparency, and operational efficiency compared to traditional systems. However, challenges such as scalability, regulatory concerns, and smart contract vulnerabilities remain areas for further improvement.

This research contributes to the growing field of decentralized philanthropy, showcasing how blockchain and ERC-20 tokens can revolutionize the way charitable donations are managed. Future work may explore Layer 2 scaling solutions to reduce fees and DAO (Decentralized Autonomous Organization)-based governance to enhance community-driven charity initiatives.

Keywords: Blockchain, ERC-20 Tokens, Smart Contracts, Decentralized Donations, Ethereum, Web3

الملخص

تعاني أنظمة التبرع التقليدية من نقص في الشفافية، وارتفاع رسوم المعاملات، وانعدام الثقة من قبل المتبرعين. إذ غالبًا ما يعجز المتبرعون عن التحقق من كيفية استخدام أموالهم، مما يؤدي إلى مخاوف تتعلق بالاحتيال، وعدم الكفاءة، وسوء الإدارة. لمعالجة هذه التحديات، تستكشف هذه الدراسة دمج رموز يغث٢٠ في نظام خيري قائم على تقنية البلوكشين، مستفيدة من العقود الذكية لضمان معاملات آمنة، شفافة، وقابلة للتدقيق.

يستخدم النظام المقترح عقودًا ذكية قائمة على شبكة الإيثيريوم لأتمتة معالجة التبرعات، مما يلغي الحاجة إلى الوسطاء ويقلل من تكاليف المعاملات. ومن خلال تنفيذ تطبيق لامركزي **Dapp** باستخدام تقنيات **Web3** ، يمكن للمتبرعين تتبع مساهماتهم في الوقت الفعلي، مما يضمن أن الأموال متاحة فقط للجهات الخيرية المعتمدة. تم تطوير المنصة باستخدام لغة **Solidity** لبرمجة العقود الذكية، وأداة **Hardhat** لاختبار البلوكشين، و **React.js** مع مكتبة **Ethers.js** لواجهة المستخدم.

لتقييم كفاءة النظام، نقوم بتحليل رسوم الغاز، وسرعة المعاملات، ومخاطر الأمان، وتحديات تبني المستخدمين. تُظهر النتائج التجريبية أن التبرعات المعتمدة على البلوكشين يمكن أن تُحسّن بشكل كبير من الثقة، والشفافية، والكفاءة التشغيلية مقارنة بالأنظمة التقليدية. ومع ذلك، لا تزال هناك تحديات مثل قابلية التوسع، والمخاوف التنظيمية، وثغرات العقود الذكية، تمثل مجالات بحاجة إلى تحسين مستقبلي.

تُسهم هذه الدراسة في مجال العمل الخيري اللامركزي المتنامي، من خلال تسليط الضوء على كيفية إمكانية أن تُحدث تقنية البلوكشين ورموز **ERC-20** ثورة في طريقة إدارة التبرعات الخيرية. وقد يتناول العمل المستقبلي استكشاف حلول التحجيم من الطبقة الثانية **Layer 2** لتقليل الرسوم، وحوكمة مبنية على المنظمات اللامركزية المستقلة **DOA** لتعزيز المبادرات الخيرية القائمة على المجتمع.

الكلمات المفتاحية : البلوكشين، رموز **ERC-20** ، العقود الذكية، التبرعات اللامركزية، إيثريوم، **Web3**

Contents

List of Figures

List of Tables

Glossary

Dapp : Decentralized Application

DAOs : Decentralized Autonomous Organizations

DeFi : Decentralized Finance

DPoS : Delegated Proof of Stake

ERC-20 : Ethereum Request for Comment 20

ETH : Ethereum

ICOs : Initial Coin Offerings

IPFS : InterPlanetary File System

PoW : Proof Of Work

PoS : Proof Of Stake

PBFT : Practical Byzantine Fault Tolerance

P2P : Peer to Peer

RAFT : Reliable, Available, and Fault-Tolerant

SC : Smart Contract

TPS : Transactions Per Second

Chapter 1

Introduction

1.1 Context

Charitable organizations play a crucial role in addressing social, economic, and humanitarian challenges worldwide. However, traditional donation systems face persistent issues such as lack of transparency, high administrative costs, and potential fraud. Donors often have limited visibility into how their contributions are utilized, leading to distrust and decreased engagement in philanthropic activities.

Blockchain technology offers a decentralized, transparent, and tamper-proof solution for donation tracking. By integrating ERC-20 tokens, a standard for fungible digital assets on the Ethereum blockchain, it becomes possible to facilitate traceable, efficient, and automated donations. This approach ensures that funds reach the intended beneficiaries without intermediaries, reducing costs and increasing accountability.

The rise of decentralized applications (dApps) has demonstrated the potential of smart contracts in automating financial transactions with minimal human intervention. This research explores the development of a blockchain-based charity donation platform that leverages ERC-20 tokens to improve transparency, reduce transaction fees, and enhance donor trust.

1.2 Problematic and Motivation

The traditional charity system faces several critical issues that hinder its effectiveness and erode donor trust. One of the most significant challenges is the lack of transparency in how funds are allocated and utilized. Donors often have no visibility into whether their contributions are being used as intended, leading to skepticism and reduced willingness to donate. Additionally, high transaction fees associated with intermediaries such as banks and payment processors further reduce the amount of funds that reach beneficiaries. These inefficiencies, combined with the potential for mismanagement or fraud, contribute to donor mistrust, which discourages future contributions.

Blockchain technology, with its inherent properties of immutability, transparency, and decentralization, provides a compelling framework for addressing these issues. By integrating ERC-20 tokens into a blockchain-based charity system, this research aims to create a platform that fosters trust, reduces costs, and enhances efficiency.

The motivation behind this research is to revolutionize the way charitable donations are made and managed, ensuring that funds are used effectively and transparently to create a positive social impact.

1.3 Objectives

The main objective of this thesis is to design, develop, and evaluate a decentralized charity platform called TASADDAQ, based on blockchain technology and Ethereum smart contracts using the ERC-20 standard. This project aims to address the limitations of traditional donation systems, particularly the lack of transparency, centralization of control, insufficient traceability, and low donor confidence.Through this work, we aim to :

- Demonstrate how blockchain can be used to create a secure, transparent, and trustless environment for charitable donations.
- Implement a system where users can connect via a Web3 wallet, create charity campaigns, and make verifiable donations in cryptocurrency.
- Deploy and test a functional prototype in a local blockchain environment (Ganache), utilizing technologies such as Solidity, React.js, Web3, Ethers.js, and Ethereum smart contracts.

1.4 Methodology

The research methodology consists of the following steps:

- **Literature Review :** Analyze existing charity platforms and blockchain-based solutions to identify best practices and gaps.
- **Smart Contract Development :** Design and implement smart contracts in Solidity to handle donations and fund allocation.
- **Web Interface Development :** Build a user-friendly front-end and integrate it with the blockchain using Hardhat and Ethers.js.
- **Testing :** Deploy and test the system on a local blockchain network to evaluate its performance, security, and usability.
- **Evaluation :** Assess the system's effectiveness in achieving transparency, reducing costs, and improving donor trust.

1.5 Thesis Organization

This thesis is structured into four main chapters, each addressing a specific phase of the project — from the introduction and background research to system implementation and conceptual modeling :

- **Chapter 1: Introduction** This chapter introduces the context of the thesis, presents the problem statement, outlines the research objectives, and defines the methodology used.
- **Chapter 2: Literature Review** This chapter explores the essential concepts related to blockchain technology, Ethereum, smart contracts, and the ERC-20 token standard. It also presents an overview of decentralized applications (DApps) and their relevance in promoting transparency in donation systems.
- **Chapter 3: System Design and Conception** This chapter presents the conception phase of the project using UML modeling. It includes use case diagrams, class diagrams, sequence diagrams, and activity diagrams that describe the system's architecture, components, and user interactions. It also covers the technical design choices and validation of the proposed solution.
- **Chapter 4: Implementation and Development Tools** This chapter details the practical implementation of the TASADDAQ platform, including development environment setup, smart contract deployment, frontend-backend integration, and Web3 interactions. Screenshots and technical explanations illustrate each feature, from wallet connection to donation validation.

Chapter 2

Literature Review

2.1 Introduction

Blockchain is a revolutionary technology that is reshaping the way businesses, societies, and political systems operate, along with transforming value exchange mechanisms. This chapter delves into the core principles of Blockchain, highlighting its significance in the development of Decentralized Applications.

2.2 Blockchain Technology

Blockchain technology stands as one of the most groundbreaking innovations of the 21st century, redefining the way digital transactions are processed and authenticated across networks. Functioning as a decentralized, distributed, and publicly accessible digital ledger, it ensures that transactions are securely recorded across multiple computers, making data modification nearly impossible without network-wide consensus. Since Bitcoin's emergence in 2009, blockchain's use cases have expanded far beyond cryptocurrencies, influencing industries such as finance, supply chain management, and healthcare. This in-depth report examines the core principles, technical architecture, real-world applications, and future prospects of blockchain, offering a comprehensive insight into its transformative impact.[**1**]

2.3 Fundamentals of Blockchain

To fully comprehend the transformative potential of blockchain technology, it is essential to first examine its theoretical underpinnings and structural framework. The subsequent subsections provide a comprehensive dissection of blockchain's conceptual foundations and technical architecture, which collectively enable its unique properties of decentralization, security, and transparency.[2]

2.3.1 Definition and Core Concepts

A blockchain, in its most basic form, is a list of transactions that anyone can view and verify. Multiple interconnected computers, known as "nodes"; maintain this digital ledger, with transaction information stored in consecutive "blocks" ; that form a chain. Each block typically contains details such as transaction

time, amount, and the addresses involved in the transactions. What makes blockchain revolutionary is its ability to enable the transfer of value online without intermediaries like banks or payment processors.

2.3.2 Blockchain Architecture and Structure

The architecture of a blockchain consists of several key components that work together to maintain the integrity and security of the distributed ledger. At the foundational level, a blockchain is composed of blocks that function as containers for data.Each block typically contains a cryptographic hash of the previous block (linking it to the preceding block), a timestamp recording when the block was created, transaction data (the actual information being recorded), and a hash of the current block serving as its unique identifier , as shown in Figure 2.1.

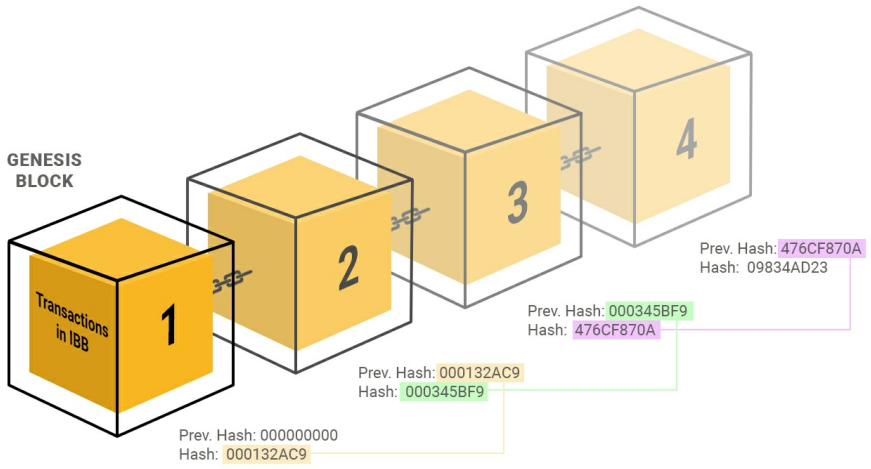

Figure 2.1: Blockchain Architecture [3]

This architecture establishes a sequential chain in which each newly added block references the preceding one, creating an immutable record of all transactions. The blockchain is upheld by a decentralized network of computers, known as nodes, each of which retains a complete copy of the ledger. This distributed framework eliminates single points of failure, enhancing security, ensuring data integrity, and allowing all participants to access and verify transaction records.

2.3.3 Blockchain Components

These are the core Blockchain architecture components :

- **Node:** user or computer within the Blockchain architecture (each has an independent copy of the whole Blockchain ledger)

- **Transaction:** smallest building block of a Blockchain system (records, information, etc.) that serves as the purpose of Blockchain.
- **Block:** a data structure used for keeping a set of transactions which is distributed to all nodes in the network
- **Wallet:** A Blockchain wallet is a digital wallet; it is used to securely store the private information of the user
- **Ledger:** Every node in the Blockchain network is used to maintain a ledger of all transactions and this transaction maintains the state of the data that is being stored on the Blockchain network
- **Peer Network:** It is peer-to-peer network contains a group of independent computers called as nodes that are interconnected to share data among each other with any centralized authority
- **Chain:** a sequence of blocks in a specific order
- **Miners:** specific nodes which perform the block verification process before adding anything to the Blockchain structure
- **Consensus:** a set of rules and arrangements to carry out Blockchain operations
- **Events:** It is used to create notifications for operations performed on the Blockchain

2.3.4 Types of Blockchain Systems

As shown in Figure 2.2 , Blockchain systems can be categorized into several types based on their accessibility and control structure [4] :

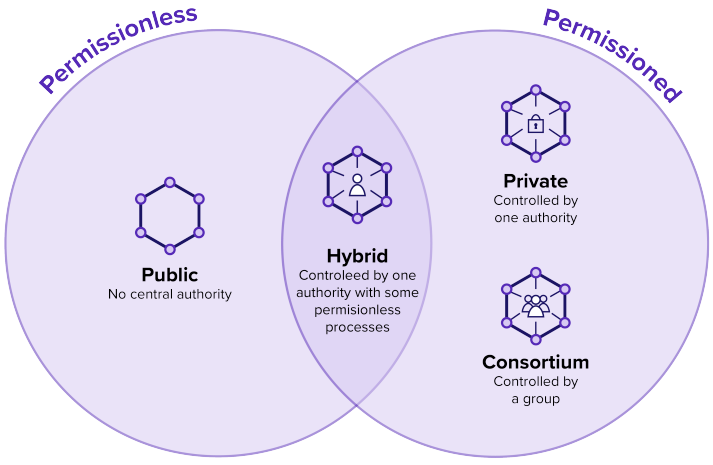

Figure 2.2: Types of Blockchain Systems [5]

1. **Public blockchain :** A public blockchain operates as an open, permissionless network where users can freely join or exit without requiring approval from a central authority. Participation is unrestricted, allowing anyone to engage in transaction validation and data recording. Transactions are added to a decentralized digital ledger maintained by a distributed network of nodes, each running the necessary blockchain software. This ensures transparency, security, and resistance to censorship, making public blockchains ideal for applications such as cryptocurrencies and decentralized finance (DeFi).

2. **Private blockchain :** In a private blockchain, network access is restricted, and participation is governed by predefined rules established by network administrators. Unlike public blockchains, where anyone can join, nodes in a private blockchain must obtain explicit permission from the administrators before gaining access and conducting transactions. Additionally, node identities are verified before they are allowed to participate, ensuring controlled access and enhanced security. This model is commonly used in enterprise applications where privacy, efficiency, and regulatory compliance are essential.

3. **Hybrid Blockchain :** A hybrid blockchain merges elements of both public and private blockchains, balancing decentralization with controlled access. It combines the openness of permissionless networks with the regulatory oversight of a centralized authority, making it a suitable solution for scenarios requiring both transparency and restricted participation. Typically governed by a consortium or a board of administrators, this model enables selective access, ensuring that sensitive data remains confidential while still benefiting from blockchain's security and immutability.

4. **Comparison of blockchain types :**

   The table 2.1 below provides a detailed comparison of the four primary blockchain types—public, private, consortium, and hybrid—highlighting their key characteristics, advantages, disadvantages, and ideal use cases.

Type	**Advantages**	**Disadvantages**	**Use Cases**
Public	✓Independence ✓Transparency ✓Trust	× Performance × Scalability × Security	Cryptocurrencies, DeFi
Private	✓Access Control ✓High Performance	× Trust Issues × Limited Auditability	Ownership Verification
Consortium	✓Access Control ✓Scalability ✓Security	× Less Transparency	Banking, Cross-Industry Apps

Table 2.1: Comparison of Blockchain Types

- **Choosing the right blockchain :** From the comparison, we can see that public and private blockchains serve different purposes, and public blockchains are ideal for applications that require transparency and trust like in our case, while private blockchains offer control and efficiency.

2.3.5 Consensus Mechanisms

For a blockchain to function effectively without central authority, the network needs a way to agree on the state of the ledger - this is where consensus mechanisms become essential. Consensus mechanisms are protocols that ensure all nodes in the network agree on which transactions are valid and should be added to the blockchain.

Different blockchains employ various consensus mechanisms, with the most well-known being Proof of Work (PoW), used by Bitcoin, which requires nodes (miners) to solve complex mathematical puzzles to validate transactions and create new blocks. This process demands significant computational power and energy. More energy-efficient alternatives include Proof of Stake (PoS), where validators are chosen based on the amount of cryptocurrency, they "stake" or lock up as collateral, and Delegated Proof of Stake (DPoS), where token holders vote for "delegates" who validate transactions and maintain the blockchain.

These consensus mechanisms ensure that all participants in the network can trust the blockchain without needing to trust a central authority to verify transactions.[6]

2.4 Smart Contracts

A smart contract is a type of computer program that runs on a blockchain and automatically carries out actions when certain conditions are met. These contracts are written in code and do not need a middleman, making transactions faster, more secure, and more transparent. They are an important part of blockchain technology and are widely used in cryptocurrencies.[7]

In simple terms, a smart contract works on its own once the conditions are fulfilled. Unlike traditional contracts, which need people to check and enforce them, smart contracts follow fixed rules and always give the same result if the same conditions happen. This removes confusion and reduces arguments since everything is decided by the program, not by people.

2.4.1 Life Cycle of a Smart Contract

A smart contract follows a structured process from creation to completion. This life cycle consists of four main phases, ensuring that the contract functions correctly and securely on the blockchain.

1. **Creation Phase :** This is the first step, where the smart contract is designed and coded. It involves:
   (a) **Defining the contract's rules :** The developer writes the contract using a programming language like Solidity (for Ethereum-based contracts).
   (b) **Setting conditions and actions :** The code includes specific conditions that must be met for the contract to execute.
   (c) **Testing the contract :** Before deployment, the contract is tested on a local or test network (e.g., Ethereum Testnet) to check for errors, security issues, and logic flaws.

2. **Deployment Phase :** Once the contract is ready, it is deployed on the blockchain, making it accessible and immutable. This phase involves:

   (a) **Publishing on the blockchain :** The contract is uploaded using blockchain tools like Remix, Truffle, or Hardhat.

   (b) **Assigning a unique address :** After deployment, the contract gets a permanent address on the blockchain, which users can interact with.

   (c) **Paying gas fees :** The deployment process requires a fee (gas cost) paid in cryptocurrency (e.g., ETH for Ethereum).

3. **Execution Phase :** This is the core phase, where the contract performs its functions based on predefined conditions.

   (a) **Triggering the contract :** A user or system interacts with the contract, activating its logic.

   (b) **Checking conditions :** The contract verifies whether the required conditions are met.

   (c) **Executing actions :** If the conditions are satisfied, the contract performs actions like transferring ERC-20 tokens, verifying identities, or storing data.

   (d) **Recording on the blockchain :** Every transaction is recorded permanently, ensuring transparency and security.

4. **Completion Phase :** The smart contract reaches its final state, meaning it has served its purpose. There are two possible outcomes :

   (a) **Successful execution :** The contract completes its tasks, and all transactions are verified and stored on the blockchain.

   (b) **Expiration or termination :** If the contract has an end date or specific condition for termination, it stops functioning after completion.

- **Why This Matters for a Charity Decentralized Application :**

  In a charity DApp using an ERC-20 token, the smart contract life cycle plays a crucial role :

  - **During deployment, the contract ensures that donations are handled securely.**
  - **In the execution phase, it verifies and processes donations automatically.**
  - **At completion, it ensures that funds are transferred as intended without middlemen, reducing fraud risks.**

  By understanding these phases, we can build a reliable and transparent donation system powered by blockchain.

2.4.2 Features of Smart Contracts

Smart contracts offer many benefits, making them useful for different applications, including our charity DApp. Below are some of the main advantages:

1. **Independence :** No middlemen are needed. The contract runs on its own, meaning no banks, lawyers, or third parties are required.
2. **Trust and Transparency :** The contract is stored on a shared blockchain, where no one can change or delete it without agreement from the network. This builds trust among users.
3. **Data Backup :** Since blockchain keeps multiple copies of the contract across different computers, your data is never lost.
4. **Security :** Blockchain uses cryptography to protect smart contracts, making it extremely difficult for hackers to alter the information.
5. **Speed and Efficiency :** Everything is automated with smart contracts, meaning tasks that usually take hours or days (like verifying payments) are done in seconds.
6. **Cost Savings :** No need to pay middlemen like banks or legal experts. Transactions happen directly between users, saving money.
7. **Accuracy :** Since everything runs through computer code, there are no human mistakes, unlike manual paperwork.

2.4.3 Challenges of Smart Contracts in a Charity DApp

The table 2.2 below outlines the key technical, operational, and regulatory challenges associated with implementing smart contracts in a Charity Decentralized Application (DApp). These challenges impact transparency, security, and trust—critical factors for donor confidence and regulatory compliance in blockchain-based philanthropy.

Challenge	Problem	How It Affects Our Charity DApp	Solution
Creating the Contract	• Hard to read and understand due to complex programming languages. • Bugs in the code can be exploited by hackers.	• If the contract has an error, funds could be lost or stolen. • Donors might hesitate if they don't trust the system.	• Use clear and simple code with commenting. • Test the contract thoroughly.
Deploying the Contract	• Once deployed, it cannot be changed. • Complex smart contracts become harder to manage over time.	• If an issue is found after launch, fixing it is difficult. • Future updates require deploying a new contract.	• Perform thorough testing before launch. • Use an upgradeable smart contract if needed.
Executing Transactions	• Transactions may not always execute in the expected order. • Too many transactions at once can slow down processing.	• Donations might take longer to process. • Some transactions could fail due to network congestion.	• Optimize contract logic to minimize transaction costs. • Implement layer-2 solutions to improve speed.
Storing and Sharing Data	• Large amounts of data (e.g., donor info) can slow operations. • Storing excessive data on-chain is expensive.	• If donation records are too large, transaction fees increase. • Need to balance transparency and efficiency.	• Store only essential data on-chain. • Use off-chain storage for reports and detailed records.

Table 2.2: Challenges of smart contracts

2.5 Choosing a Blockchain Framework

A blockchain framework is a software solution that includes the infrastructure and libraries needed to develop blockchain applications. The network infrastructure, or simply the blockchain infrastructure, consists of nodes and software components running on them.(See Table 2.3 for a comparative analysis of popular blockchain frameworks and their core features.) These frameworks provide essential features such as user identity management, transaction validation, consensus protocols, and overall identity control mechanisms within a blockchain network.[8]

Feature	**Bitcoin**	**Ethereum**	**Hyperledger**	**Corda**
Privacy Features	No private transactions	Public (supports Private) transactions	Private channels, private transactions	Private transactions
Access	Public	Public (supports permissioned networks)	Permissioned	Permissioned
Consensus Mechanism	PoW (Proof of Work)	PoW (Proof of Work)	Multiple approaches (PBFT, RAFT, etc.)	Multiple approaches (Not PoW)
Smart Contracts	Not available	Solidity	Java, Go, JavaScript	Kotlin, Java, C#
Industry Focus	Finance	Cross-industry	Cross-industry	Finance

Table 2.3: General Comparison of Blockchain Frameworks

- **Why Ethereum is the Best Choice for the Charity DApp ?**
  Five key factors make Ethereum ideal for charity DApps:

  1. **Smart Contracts and DApp Support :** Ethereum natively supports smart contracts, allowing transparent, automated transactions without intermediaries. This is crucial for a charity system where funds must be transparently allocated to intended beneficiaries.
     - Smart contracts eliminate fraud and ensure donations reach the right recipients.
     - Ethereum's Solidity programming language is widely used, with strong developer support.
     - ERC-20 token standard makes it easy to create a donation token for tracking contributions.
  2. **Transparency and Public Accessibility :** Since Ethereum is a public blockchain, donors and beneficiaries can track all transactions in real time. This builds trust in the charity system.
     - Publicly visible transactions prevent fraud and corruption in the donation process.
     - Ethereum allows hybrid private-public approaches, enabling privacy where necessary.

3. **Security and Reliability:** Ethereum is one of the most battle-tested blockchains, with a large network of validators ensuring security. Unlike newer platforms, Ethereum has been extensively used in DeFi, enterprise solutions, and large-scale applications.
   - Decentralized security model protects against tampering.
   - Funds are secured by cryptographic verification, making unauthorized access nearly impossible.
4. **ERC-20 Token for Donations :** Our Charity DApp plans to use an ERC-20 token, which Ethereum fully supports. The advantages of using Ethereum's token standard include:
   - Interoperability with wallets, exchanges, and DeFi platforms.
   - Easy tracking and auditing of donations.
   - Potential to integrate with staking mechanisms for rewarding donors.
5. **Cost and Scalability Considerations :** While Ethereum's mainnet can have high gas fees, multiple Layer 2 solutions (e.g., Polygon, Arbitrum, Optimism) provide cheap and fast transactions.
   - Ethereum 2.0 and Layer 2 scaling allow for thousands of TPS, reducing fees.
   - Off-chain storage (e.g., IPFS for documents) minimizes data costs.

Ethereum is the best choice for our charity DApp due to its smart contract capabilities, privacy options, and strong developer support. It ensures transparency, security, and automation for donations, making it ideal for blockchain-based philanthropy.

2.6 ERC-20 Tokens

The ERC-20 token standard has become the foundation for creating fungible digital assets on the Ethereum blockchain.[9]

2.6.1 Introduction to ERC-20 Tokens

ERC-20 (Ethereum Request for Comment 20) pertains to a set of proposed technical standards used for the generation of fungible tokens on the Ethereum blockchain. As the name indicates, fungible tokens, which are interchangeable, are identical in terms of value and functionality. This standard was suggested in 2015 by Fabian Vogelsteller and adopted more than any other token standard in the blockchain ecosystem.

ERC-20 tokens are used for a variety of purposes, including: crowdfunding, payments and governance. By standardizing ERC-20 tokens, there is a guarantee of integration with Ethereum based wallets, exchanges, and smart contracts, thus the implementation in decentralized applications (DApps) is much simpler.[**6**]

2.6.2 Technical Specifications of ERC-20 Tokens

The ERC-20 standard outlines a list of rules relating to both a token's mandatory and optional functions. These functions assist in the creation, transfer, and governance of these tokens on the Ethereum blockchain.

1. **Mandatory Functions :**

(a) **totalSupply :**

- Returns the total number of tokens in circulation.
- Example: A charity platform might issue 1,000,000 tokens to represent donations.

(b) **balanceOf :**

- Returns the token balance of a specific address.
- Example: A donor's wallet address might hold 500 tokens.

(c) **transfer :**

- Allows a user to send tokens to another address.
- Example: A donor transfers 100 tokens to a charity's wallet.

(d) **transferFrom :**

- Allows a third party (e.g., a decentralized exchange) to transfer tokens on behalf of a user.
- Example: A smart contract automatically transfers tokens from a donor's wallet to a charity's wallet.

(e) **approve :**

- Allows a user to authorize a third party to spend a specific amount of tokens.
- Example: A donor approves a decentralized exchange to spend up to 200 tokens.

(f) **allowance :**

- Returns the remaining number of tokens that a third party is allowed to spend on behalf of a user.
- Example: A smart contract checks how many tokens a donor has approved for a specific transaction.

2. **Optional Functions :**

(a) **name :**

- Returns the name of the token ("CharityToken").

(b) **symbol :**

- Returns the symbol of the token ("CTK").

(c) **decimals :**

- Returns the number of decimal places the token can be divided into (e.g., 18 decimals for high precision).

2.6.3 How ERC-20 Tokens Work

Tokens that fall under the ERC-20 standard are produced and updated using smart contracts on the Ethereum blockchain. Smart contracts virtually outline the governance processes regarding the token's issuance, transferring, and management. The specific processes are described below:[10]

1. **Token Creation :**

- A developer writes a contract on Solidity (the programming language of Ethereum) which complies with ERC-20.
- The token is minted upon deploying the contract into Ethereum Blockchain.

2. **Token Distribution :**
   - ICOs, direct transfers, or airdrops are some of the methods that can be used for distributing tokens.
   - For Example, a charity platform rewards users by transferring tokens to donors as donations.
3. **Token Transactions :**
   - Tokens can be transferred by other addresses through the transfer function.
   - Users can authorise third parties (token swapping platforms) to transfer funds using the transferFrom function.
4. **Token Management :**
   - Users are able to query their token balance by executing the balanceOf function.
   - With the totalSupply function, developers are able to track the total amount of tokens superb.

2.6.4 Use Cases of ERC-20 Tokens

ERC-20 tokens have a wide range of applications across various industries. Some of the most common use cases include:

1. **Crowdfunding :**
   - Tokens can be sold to raise funds for projects, startups, or charitable causes.
   - Example: A charity platform issues tokens to represent donations and rewards donors with governance rights.
2. **Payments :**
   - Tokens can be used as a medium of exchange for goods and services.
   - Example: A donor uses tokens to contribute to a charity campaign.
3. **Governance :**
   - Tokens can represent voting rights in decentralized organizations (DAOs).
   - Example: Donors use tokens to vote on how funds are allocated within a charity platform.
4. **Loyalty Programs :**
   - Tokens can be used to reward users for their participation or contributions.
   - Example: A charity platform rewards donors with tokens for their ongoing support.

2.6.5 Features of ERC-20 Tokens

Building on the foundational role of ERC-20 in Ethereum's ecosystem, this standard offers several key benefits that have fueled its widespread adoption. These advantages make ERC-20 the go-to choice for most fungible token implementations on Ethereum:

1. **Interoperability :**
   - ERC-20 tokens can be used across all Ethereum wallets, exploitations, and smart contracts which makes their integration into different systems seamless.
2. **Ease of Creation :**
   - Developers looking to create ERC-20 tokens can effortlessly do so with the use of public templates and tools, for instance, OpenZeppelin.
3. **Liquidity :**
   - Platforms like Uniswap enable trading in ER20 tokens and other DEXs which gives liquidity and allows for the determination of value estimates.
4. **Transparency :**
   - The ownership and transfer of tokens are recorded, which means all transactions can be traced easily and thus guarantees honesty.

2.7 Decentralized applications

Decentralized applications, from a business perspective, allow individual units to make local decisions and use their resources to achieve specific goals. While these units must collaborate with others and report to central management, they have the freedom to design their internal structures for efficiency. Central coordination is still necessary to maintain alignment and prevent chaos, setting boundaries without stifling local autonomy. In essence, decentralization balances local flexibility and innovation with overall organizational harmony. [11]

2.7.1 Definition

Decentralized applications, or Dapps, are digital programs that operate on a peer-to-peer (P2P) network of computers rather than relying on a single machine. Unlike traditional applications, they function independently of any single authority, giving users greater control over their data. By leveraging blockchain technology, Dapps eliminate the need for centralized intermediaries to manage data, making the service truly decentralized.

Decentralization means that control is distributed among the end users rather than being held by a single entity. In this setup, the peers in the network are responsible for maintaining and sharing content. The more peers that participate, the higher the availability and download speed of the data. One of the key advantages of decentralized systems is their resilience—they have no single point of failure. Even if

multiple nodes go offline, the network as a whole remains operational. Since no single entity controls the network, the chances of it going down are virtually zero unless every node fails simultaneously, which is highly unlikely. This makes decentralized systems robust, reliable, and user-empowered.

2.7.2 Centralized Systems

Centralized systems operate on a client and server model, where one or more client devices—like computers or mobile phones—connect directly to a central server (see Figure 2.3 for a visual representation of this architecture). This setup is widely used across many organizations. In such systems, a client sends a request to the company's server and waits for a response. The key components of a centralized system include:

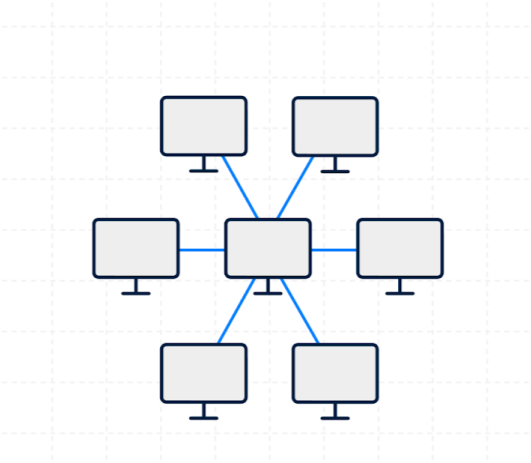

Figure 2.3: Centralized Systems[12]

- **Nodes :** These are the devices used by clients, such as computers, smartphones, or tablets.
- **Server :** The central hub that processes requests and sends back responses.
- **Communication Link :** The connection between nodes and the server, which can be wired (like cables) or wireless (like Wi-Fi).

This structure is simple and efficient, making it a popular choice for businesses and organizations.

2.7.3 Decentralized Systems

Unlike centralized systems, decentralized systems distribute responsibilities across multiple nodes (as illustrated in Figure 2.4). In this setup, all or several nodes store essential information, handle query processing, and manage tasks independently. This means users don't have to rely on a central authority every time they need to analyze a query or access resources. As a result, decentralized systems are more resilient to failures and significantly more scalable. By spreading the workload and data across multiple

points, these systems can handle larger demands and continue functioning even if some nodes go offline. This makes them robust, flexible, and better suited for growing needs.

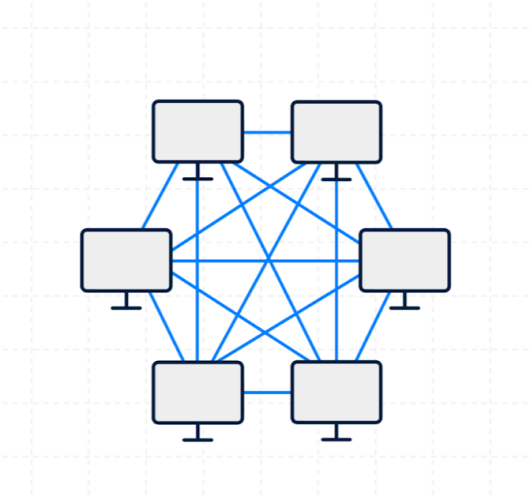

Figure 2.4: Decentralized Systems[13]

The concept of Peer-to-peer introduces many significant advantages in different aspects of resource discovery including scalability (due to collaborative resource sharing between peers), reliability (fault-tolerance due to the equality essence of peers), and robustness due to self-organization against peer or system failures. For resource discovery in P2P systems.

2.7.4 Centralized vs Decentralized

Decentralization should be implemented where it truly adds value. The primary aim of any blockchain solution is to meet the needs of its users, and this doesn't always require full decentralization. To better grasp how decentralized networks function, the table 2.4 below highlights the key differences between decentralized networks and the more familiar centralized networks. This comparison helps clarify when and why decentralization might be the right choice.[14]

	Centralized	Decentralized
Network/hardware resources	Maintained & controlled by single entity in a centralized location.	Resources are owned & shared by network members; difficult to maintain since no one owns it.
Solution components	Maintained & controlled by central entity.	Each member has exact same copy of distributed ledger.
Data	Maintained & controlled by central entity.	Only added through group consensus.
Control	Controlled by central entity.	No one owns the data & everyone owns the data.
Single Point of Failure	Yes	No
Fault Tolerance	Low	Extremely high
Security	Maintained & controlled by central entity.	Increases as the number of network members increase.
Performance	Maintained & controlled by central entity.	Decreases as the number of network members increase.
Example	ERP Systems	Blockchain

Table 2.4: Centralized vs Decentralized

We explored the design features of both centralized and decentralized systems and gained insight into the technical advantages of decentralized systems over centralized ones. One of the key benefits of decentralized systems is enhanced data privacy. In such systems, information flows through multiple points rather than a single entity, making it much harder to track or monitor. A prime example of a decentralized network is blockchain, which embodies these principles by distributing data across a network of nodes, ensuring greater security and privacy for users.

2.8 Blockchain in Charity Systems

Addressing issues like poverty, education, and disaster relief requires the assistance of charity organizations. However, traditional charity systems often have problems like insufficient transparency, high expenses, and deep-seated donor distrust. Such issues can easily be remedied by the use of blockchain technology. Charities can account for donations and allocate resources through funds in a more dependable, transparent, and efficient way. [15]

2.8.1 How Blockchain Works in Charity Systems

Blockchain technology can be applied to charity systems in several ways:

- **Transparent Donations :** Every donation is recorded on the blockchain, creating a permanent and tamper-proof record. Donors can track how their contributions are used in real-time, ensuring accountability.
- **Smart Contracts for Fund Allocation :** Smart contracts can automate the distribution of funds based on predefined rules. For example, funds can be automatically released to a charity when specific milestones are met.
- **Reduced Transaction Costs :**By eliminating intermediaries like banks and payment processors, blockchain reduces transaction fees, ensuring that more funds reach the intended beneficiaries.
- **Global Accessibility :**Blockchain enables cross-border donations without the need for currency conversion or high fees, making it easier for people around the world to contribute.

2.8.2 Benefits of Blockchain in Charity Systems

With the implementation of blockchain technology, charities can benefit from its transparency and efficiency. Donors cancel out any suspicions they may have over the credibility of the charity organizations and track every dime they put in using the transparent system on the blockchain. Administrative costs, as well as the costs required for hiring intermediaries, are reduced with the implementation of smart contracts for donation allocation. Not only does this make the process more seamless, but intermediaries are often known to take a huge portion of the funds intended to help. Furthermore, blockchain technology allows the donors to execute global transactions with nominal fee deductions and irrespective of the currency type, thereby broadening the donor scope. The gap of trust that lies between charities and donors is filled with the reduced chances of fraud and the indisputable features of mismanagement blockchain has to offer. Improving efficiency, accountability as well as having a greater impact on the charity systems, makes blockchain the better alternative.

2.8.3 Real-World Applications of Blockchain in Charity

Several organizations and platforms are already using blockchain to improve charity systems (see Figure 2.5) . Here are a few examples [16]:

- **Binance Charity :** Binance Charity uses blockchain to enable transparent donations to global causes. Donors can track how their contributions are used, and funds are distributed directly to beneficiaries.
- **Giveth :** Giveth is a decentralized platform that allows users to donate to social impact projects using Ethereum-based tokens. Smart contracts ensure that funds are used as intended.
- **Alice :**Alice uses blockchain to connect donors with social impact projects. Funds are only released when predefined milestones are achieved, ensuring accountability.

- **GiveDirectly :**GiveDirectly uses blockchain to facilitate direct cash transfers to individuals in need. Donors can track how their contributions are used, and funds are distributed efficiently.

Figure 2.5: Real-World Applications of Blockchain in Charity

2.8.4 Challenges of Blockchain in Charity Systems

While blockchain offers many benefits, it also comes with challenges that need to be addressed:

- **Scalability Issues :** The increase in transaction volume causes certain blockchain networks to become slow and expensive.
- **User Adoption :** Both charitable organizations and donors may not be technologically savvy regarding the use of blockchains which makes adoption more challenging. This barrier must be addressed with education and better interfaces.
- **Regulatory Compliance :** All charities making use of blockchains must ensure compliance with local, national and international regulations which tend to become perplexing and differs with countries.
- **Environmental Impact :** Charities have to take into account the significant amount of energy certain blockchains use, like Bitcoin. More energy efficient blockchains such as Ethereum 2.0 or polygon should be preferred.

2.8.5 The Role of ERC-20 Tokens in Charity Systems

ERC-20 tokens, which are built on the Ethereum blockchain, play a key role in blockchain-based charity systems. Here's how they can be used:

- **Tracking Donations :** Tokens can represent donations, making it easy to track how funds are used.
- **Rewarding Donors :** Charities can reward donors with tokens for their contributions, encouraging ongoing support.
- **Transparent Fund Allocation :** Smart contracts can automate the distribution of tokens to beneficiaries, ensuring that funds are used as intended.
- **Global Donations :** Tokens make it easy for people around the world to donate without worrying about currency conversion or high fees.

2.8.6 Future of Blockchain in Charity Systems

Blockchain technology has the potential to revolutionize charity systems by making them more transparent, efficient, and trustworthy. As the technology continues to evolve, we can expect to see:

- **Increased Adoption :** With increase in understanding of Blockchain's benefits, more organizations and individuals willing to donate are going to utilize it.
- **Improved Scalability:** Progress in blockchain, like Ethereum 2.0 and Layer 2, will improve scalability and decrease transaction fee's helping address these problems.
- **New Use Cases :** Blockchain will bring a new approach in supporting charity work through decentralized donations and tokens for achievement rewards.
- **Greater Impact :** With Blockchain implemented, charity cna minimize the effect of their efforts, guaranteeing that the most money goes to those who need it the most.

2.9 Conclusion

In this chapter, we explored the foundations of blockchain technology and its potential to revolutionize charity systems. We covered the fundamentals of blockchain, including its decentralized, transparent, and immutable nature, and discussed Ethereum and ERC-20 tokens, which enable the creation of standardized digital assets. We also examined real-world blockchain applications in charity systems, highlighting their ability to enhance transparency, reduce costs, and build donor trust, while acknowledging challenges like high fees and scalability. Finally, we looked at decentralized applications (DApps), which use smart contracts to automate processes and improve efficiency. Overall, blockchain offers a transformative solution for creating more transparent, efficient, and trustworthy charity systems, paving the way for greater impact and collaboration in the future.

Chapter 3

Conception

3.1 Introduction

This chapter focuses on the conceptual design of a blockchain-based charity system to facilitate secure, traceable, and efficient donations. The system aims to eliminate intermediaries, reduce fraud, and ensure that funds reach their intended beneficiaries. To achieve this, we employ Unified Modeling Language (UML)—a standardized modeling approach—to systematically define the system's architecture, interactions, and workflows. UML provides a structured way to visualize the system's components, making it easier for developers, stakeholders, and auditors to understand the proposed solution before implementation.

3.2 Purpose

The purpose of this phase is to design the functional and structural blueprint of the TASADDAQ application. It includes identifying system actors, use cases, key components, and the flow of interactions between users and smart contracts. This modeling helps to define how each part of the system behaves, supports communication between team members, and reduces ambiguity during development. It also ensures that the platform meets its goals of decentralization, transparency, and usability.

3.3 Validation

The models presented in this chapter were validated by comparing them to the actual system behavior during the implementation phase. Each diagram reflects the real functionalities and logic implemented in the TASADDAQ platform. The use cases and sequences were tested in a development environment to ensure they align with smart contract interactions. This validation confirms the accuracy and relevance of the design models and ensures that the system architecture is consistent, scalable, and ready for deployment on a public Ethereum network in future stages.

3.4 Architecture overview

This section provides a comprehensive technical breakdown of the system architecture depicted in the figure 3.1, detailing each module's responsibilities, interactions, and implementation specifics :

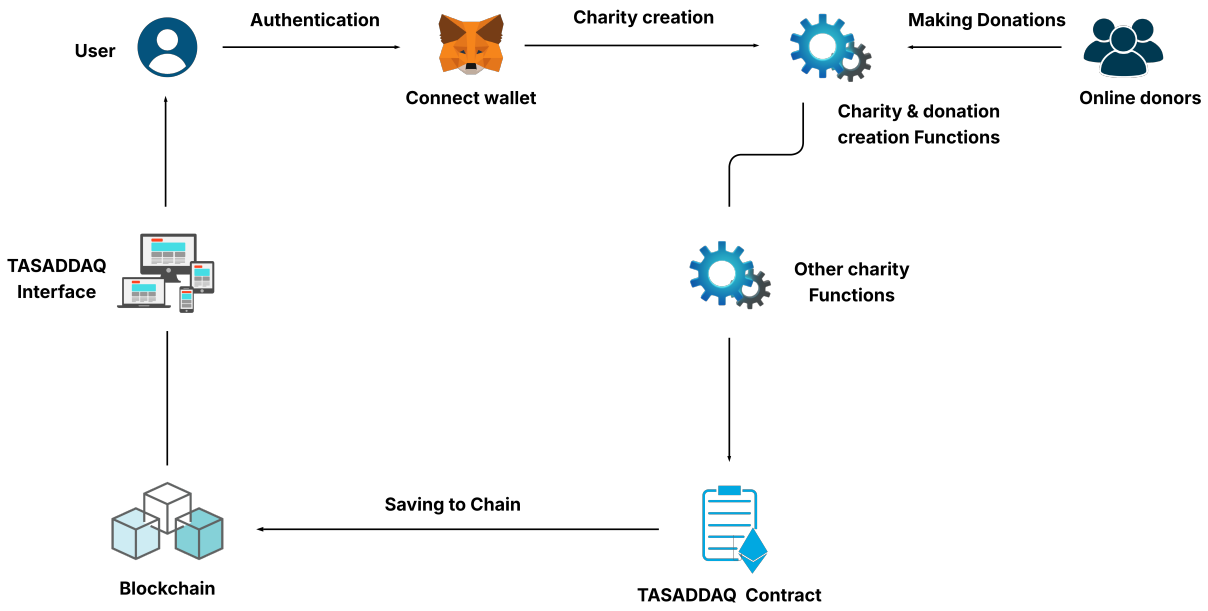

Figure 3.1: Architecture overview

1. Core Layers :

Our platform combines web authentication with blockchain interactions through three tiers. The Authentication Layer handles user sign-ups or logins and wallet connections (like MetaMask), while the Admin Layer empowers privileged users to moderate charities, adjust fees, and ensure compliance. The Charity Operations Layer lets donors browse or contribute to campaigns and allows creators to manage their listings, all powered by the TASADDAQ Contract on the blockchain, which processes transactions, enforces rules, and stores immutable records.

2. Data Flow :

Users initiate actions (e.g., donations) via the frontend, triggering smart contract functions that validate inputs, update on-chain state (like charity balances), and emit events. The frontend listens for these events (e.g., DonationReceived) to dynamically update the UI. This architecture ensures transparency (all operations on-chain), security and scalability (gas-efficient mappings for storage), bridging traditional web with decentralized logic.

3.5 Unified Modeling Language (UML)

The conception phase employs UML to rigorously define the blockchain charity system's architecture. This section systematically presents three UML diagram types, each addressing distinct aspects of the system's design, followed by their project-specific instantiations.[17]

3.5.1 Use Case

Use case diagrams capture functional requirements by mapping interactions between actors and the system (as shown in Figure 3.2). Each use case represents a distinct interaction a user can have with the system:

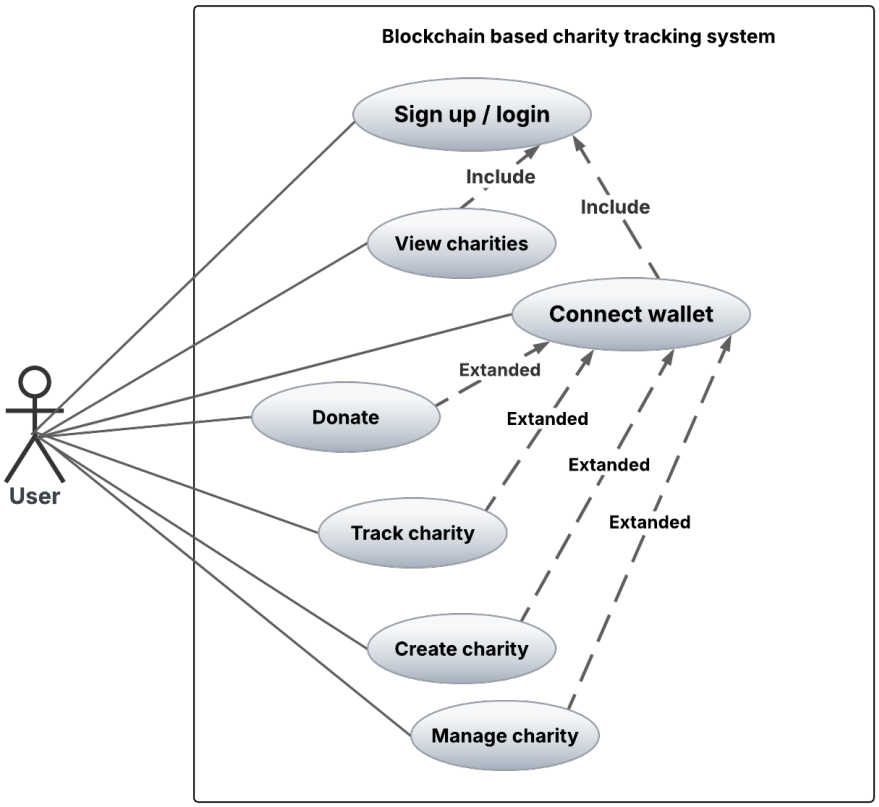

Figure 3.2: Use Case

1. **Core Functionality :**

Our blockchain-based donation tracking system enables users to sign up or login, connect their crypto wallet (like MetaMask), and interact with charity campaigns. Key use cases include browsing charities, creating or managing campaigns (for organizers), donating funds, and tracking contributions. These actions require wallet authentication (via «include» relationships) and trigger smart contract interactions on the blockchain (e.g., recording donations or registering new charities).

2. **Security Roles :**

The system enforces role-based access:

- Regular users can donate or create campaigns, with all transactions validated by smart contracts.
- Donors have exclusive privileges like banning fraudulent charities or adjusting platform settings. The blockchain serves as the trust layer, ensuring transparency (immutable records) and security (wallet-signed transactions).

3.5.2 Sequence Diagram

A UML sequence diagram visually represents how system components interact in a time-ordered sequence, particularly crucial for our blockchain system where transaction flows and smart contract interactions require precise modeling. In our case, there are four core sequence diagrams :

1. **User Login Sequence :**

The TASADDAQ platform provides two methods for user login: the first is a traditional login using an email and password, and the second is a Web3 wallet-based login using MetaMask or another compatible wallet. While the wallet connection method offers a decentralized authentication experience, it is considered optional at the login stage. Regardless of the method used to access the platform, the user must eventually connect a wallet to interact with blockchain-based features such as creating campaigns or making donations.

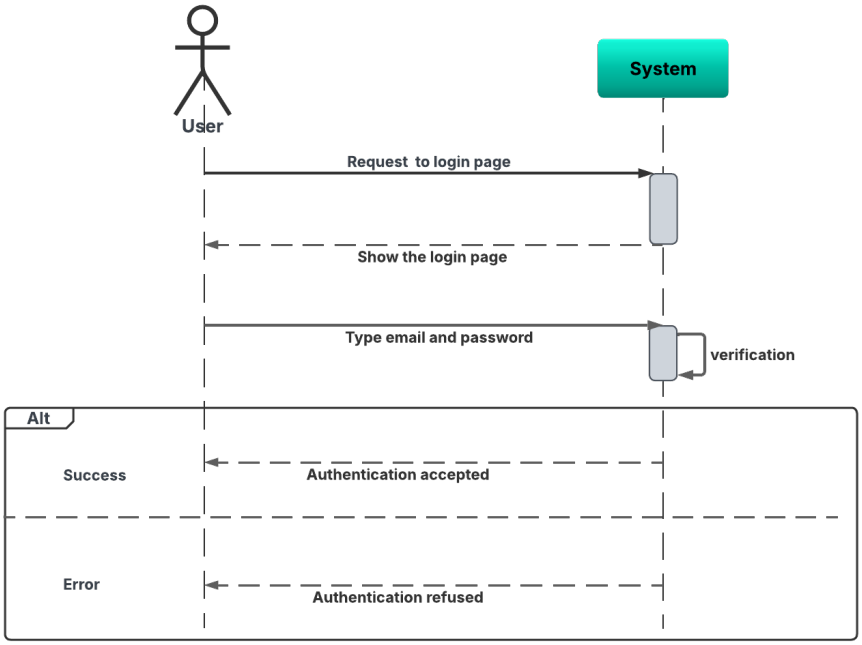

Figure 3.3: User Login Sequence

In this section, we focus on the traditional login flow, begins when a user accesses the platform's login page and submits their credentials (username/email and password). The system immediately performs validation checks to ensure proper formatting before verifying the credentials against the stored user database records.

For successful authentication, the system grants access by creating a secure session and redirecting the user to their personalized dashboard interface. The session maintains the user's authenticated state throughout their browsing experience.

In cases of authentication failure, the system displays a generic error message ("Username or password wrong") without specifying which field was incorrect, maintaining security best practices. Simultaneously, the system records the failed attempt in security logs and implements protective measures that temporarily lock the account after multiple consecutive failed attempts to prevent brute force attacks.

2. Wallet Connection Sequence :

The wallet connection sequence, as shown in Figure 3.4, enables users to link their cryptocurrency wallet (e.g., MetaMask) to the donation platform, establishing the foundation for all blockchain interactions. This critical process occurs after successful user authentication and precedes any on-chain transactions. Users can either log in directly using their wallet or choose the traditional method (email and password), then connect their wallet afterward. In both cases, wallet connection is required before accessing blockchain-related features such as creating or donating to a campaign.

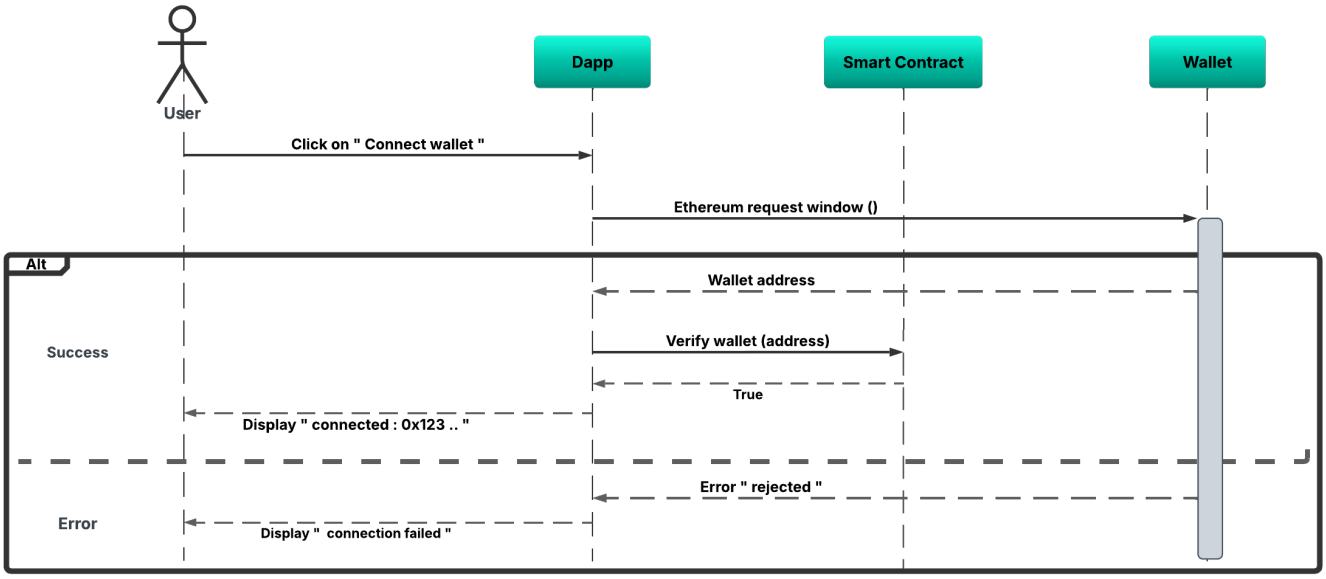

Figure 3.4: Wallet Connection Sequence

The wallet connection process in TASADDAQ ensures secure user authentication and access to blockchain features through a structured three-phase interaction:

(a) Connection Initiation

The wallet connection process begins when the user clicks the "Connect Wallet" button on the platform's web interface. This action triggers the application to send a connection request through the Ethereum provider API (window.ethereum), initiating the wallet linking sequence.

(b) Wallet Interaction

Upon receiving the request, the MetaMask extension opens a connection prompt for user review. This prompt clearly displays:

- The requesting of our decentralized application
- The specific information to be shared (only the public wallet address)
- The blockchain network being accessed

(c) Authorization Outcomes

in case of success or failure, the system responds accordingly :

i. **Successful Connection :**
When approved, MetaMask returns the user's public address (0x...). The web application then:
- Stores the address in the session state
- Updates the interface to display the connected status
- Activates all blockchain-related features
- Associates the address with the user account (for authenticated users)

ii. **Failed Connection :**
The system handles several common failure cases:
- User rejection of the connection request
- Absence of an Ethereum provider
- Network incompatibility

3. Charity Creation Sequence :

This sequence diagram, as shown in Figure 3.5, illustrates the end-to-end process of creating a new charity campaign on the blockchain-based donation platform. The flow combines web form interactions with smart contract executions, ensuring transparency and user control throughout. To achieve this, we perform the following steps :

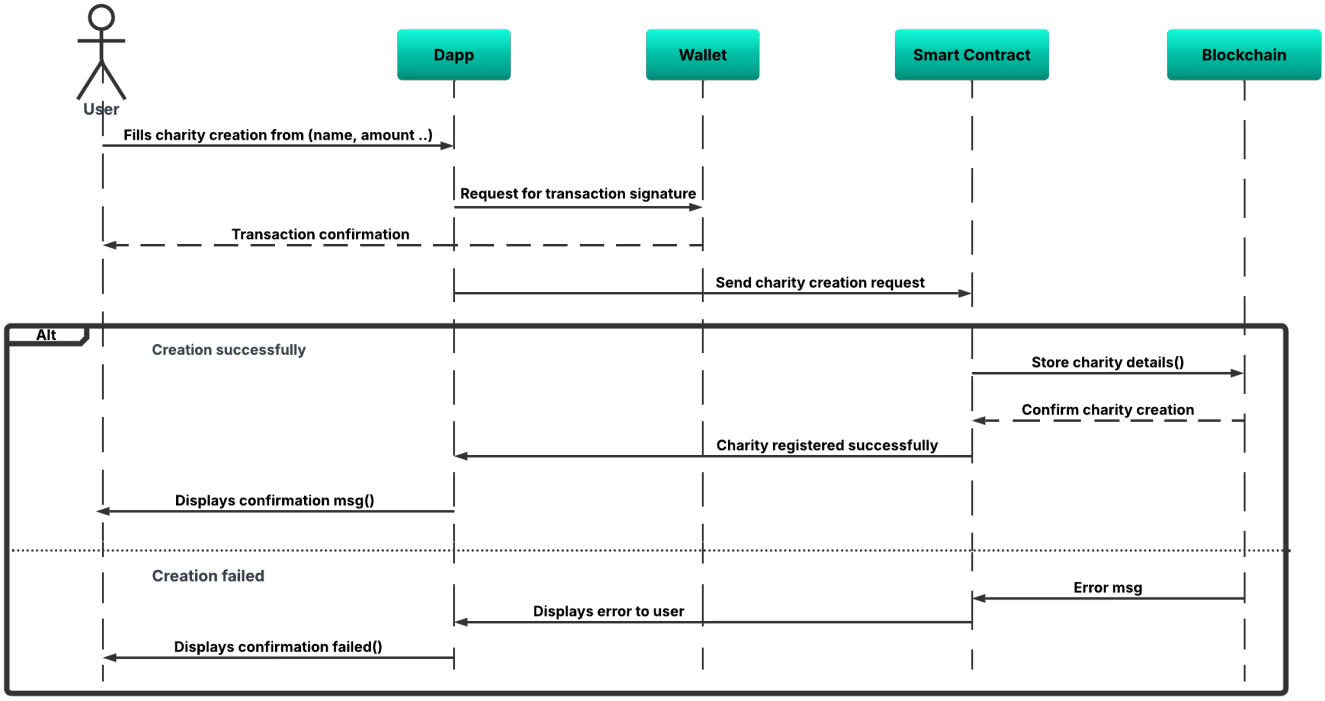

Figure 3.5: Charity Creation Sequence

(a) Form Submission Phase

The user fills out a creation form with:

- Charity name
- Funding target (in ETH or ERC-20 tokens)
- Description and metadata (e.g., image URL)

The web app performs client-side validation (e.g., non-empty fields, valid amount).

(b) Transaction Preparation

- The app constructs a transaction calling the smart contract's createCharity function
- Parameters are encoded, including a unique metadataURI (storing charity details on IPFS)

(c) Wallet Interaction

MetaMask prompts the user to:

- Review the transaction details
- Approve gas fees
- Sign the transaction

User approval triggers the TX broadcast to the blockchain.

(d) On-Chain Execution

The smart contract verifies:

- Valid creator address
- Minimum funding target
- Unique charity ID generation

A CharityCreated event is emitted upon success.

4. Donation Processing Sequence :

This section provides a comprehensive technical breakdown of the donation processing workflow, as shown in Figure 3.6, detailing how user contributions are securely recorded on the blockchain through a series of coordinated interactions between the web interface, cryptocurrency wallet, and smart contract system, by the following steps :

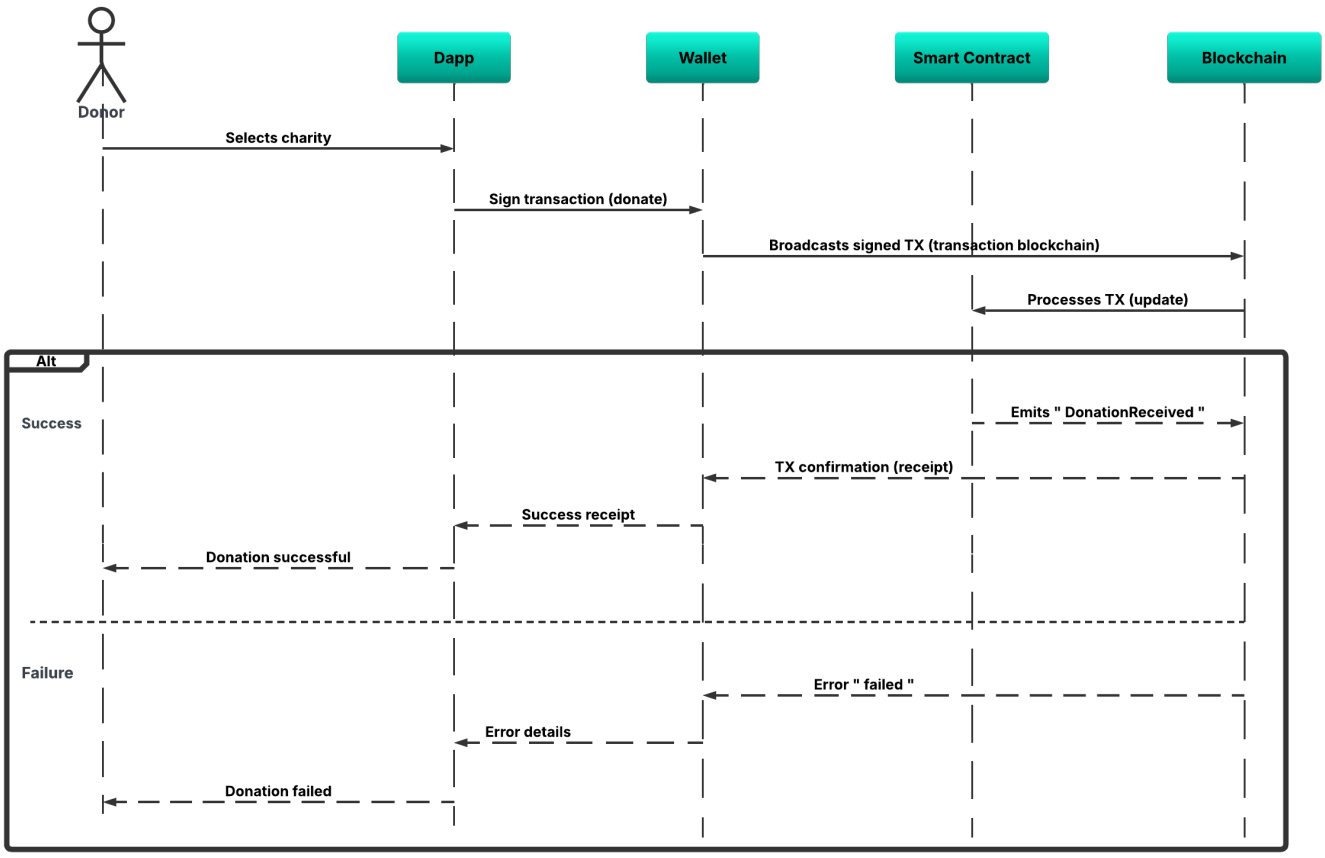

Figure 3.6: Donation Processing Sequence

(a) **Charity Selection**

To choose or select a charity, follow these steps :

- User browses active charity listings
- Selects a cause to support
- Reviews key details (current funding, description)

(b) **Donation Initiation**

To make a donation, follow these steps :

- User enters: Donation amount (in ETH or ERC-20 tokens) and Optional support message
- Web app calculates estimated gas fees
- Presents transaction summary for review

(c) **Transaction Authorization**

MetaMask prompts user to:

- Confirm donation amount
- Approve gas fees
- Sign the transaction

Signed transaction broadcast to network

(d) **Outcome Handling**

in case of success or failure, the system responds accordingly:

i. **Success Case :**

The processes are as follows :

- Transaction mined (average 15 sec on Ethereum)
- Smart contract emits DonationReceived event status
- Web app: Updates charity funding progress , Displays success notification , Adds to user's donation history

ii. **Failure Case :**

Common reasons include :

- Insufficient balance ,Transaction reverted , Network congestion
- Web app displays specific error: "Insufficient ETH balance" , "Transaction reverted: Charity inactive" , "Try with higher gas fee"

3.5.3 Class Diagram

The class diagram, as shown in Figure 3.7, outlines a blockchain-based donation system where Users (either donors or charity creators) connect their Wallet to interact with SmartContracts deployed on the Blockchain. Donors use the donate() method to contribute funds, which creates a Donation record linked to a specific Charity tracking details like amount, timestamp, and transaction hash. Charities, created via createCharity(), store immutable metadata (name, target amount, status) and receive donations through

smart contract functions like processDonation(), emitting events (e.g., DonationReceived) for real-time updates. The Blockchain validates and records all transactions, ensuring transparency and security, while relationships between classes (e.g., User-to-Charity for creation, Donor-to-Donation for contributions) maintain data integrity across the decentralized application.

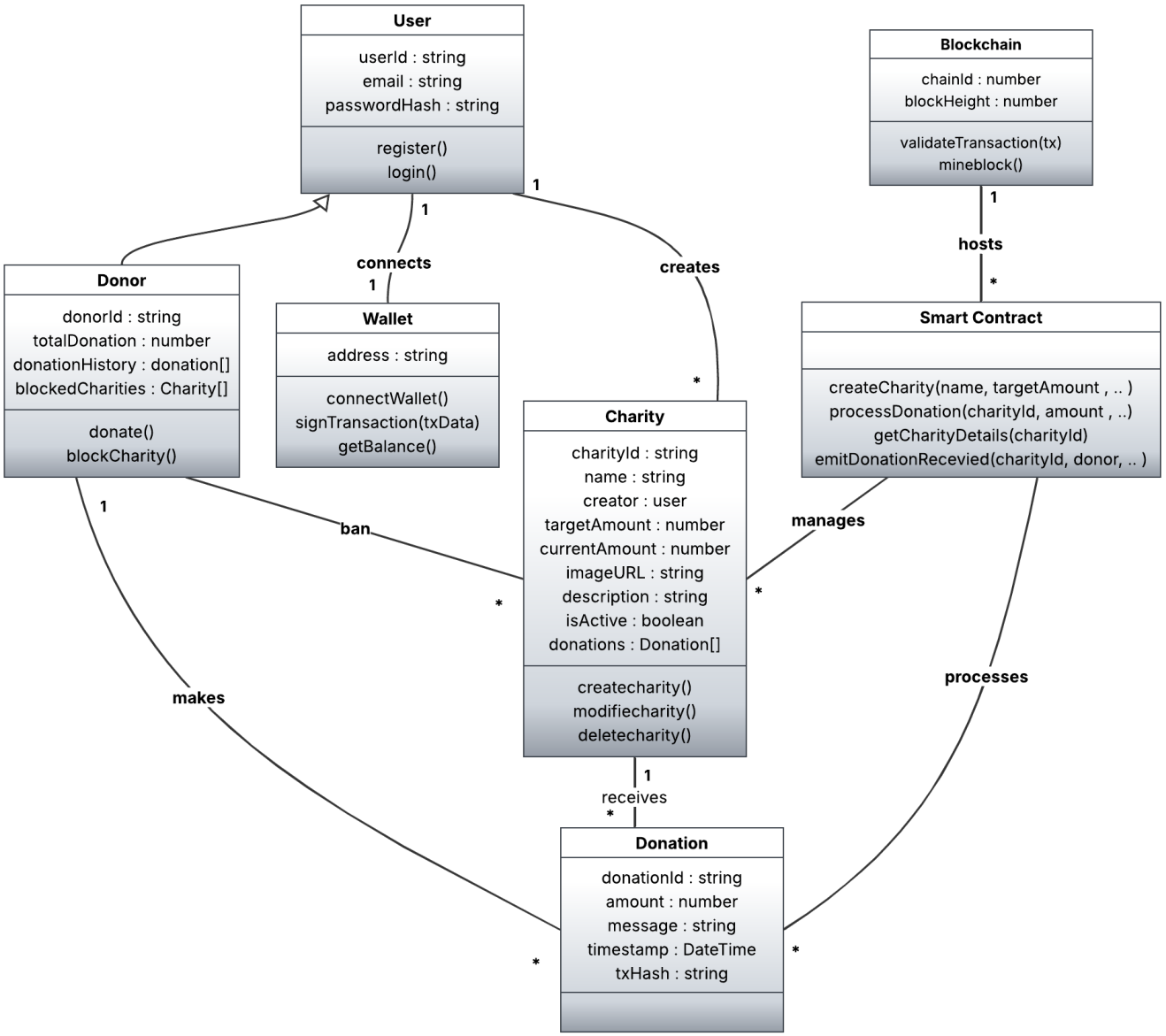

Figure 3.7: Class Diagram

3.5.4 Smart contract class diagram

This class diagram outlines the core architecture of our Dapp smart contract, as shown in Figure 3.8, featuring three main components: the Dapp contract itself with state variables like charityTax and mappings to store charity campaigns and donations, along with key functions for creating charities (createCharity), processing donations (donate), and admin controls (toggleBan); two data structures (Charity and Support) that respectively define campaign details (name, funding goal, raised amount) and donation records (donor, amount, message); and inheritance from Ownable and AccessControl for ownership and permission management. The relationships show how the contract stores multiple charity instances and their associated donations while enforcing access control, creating a secure and efficient system for managing blockchain-based charitable donations with a clear separation of concerns between data storage, business logic, and permissions.

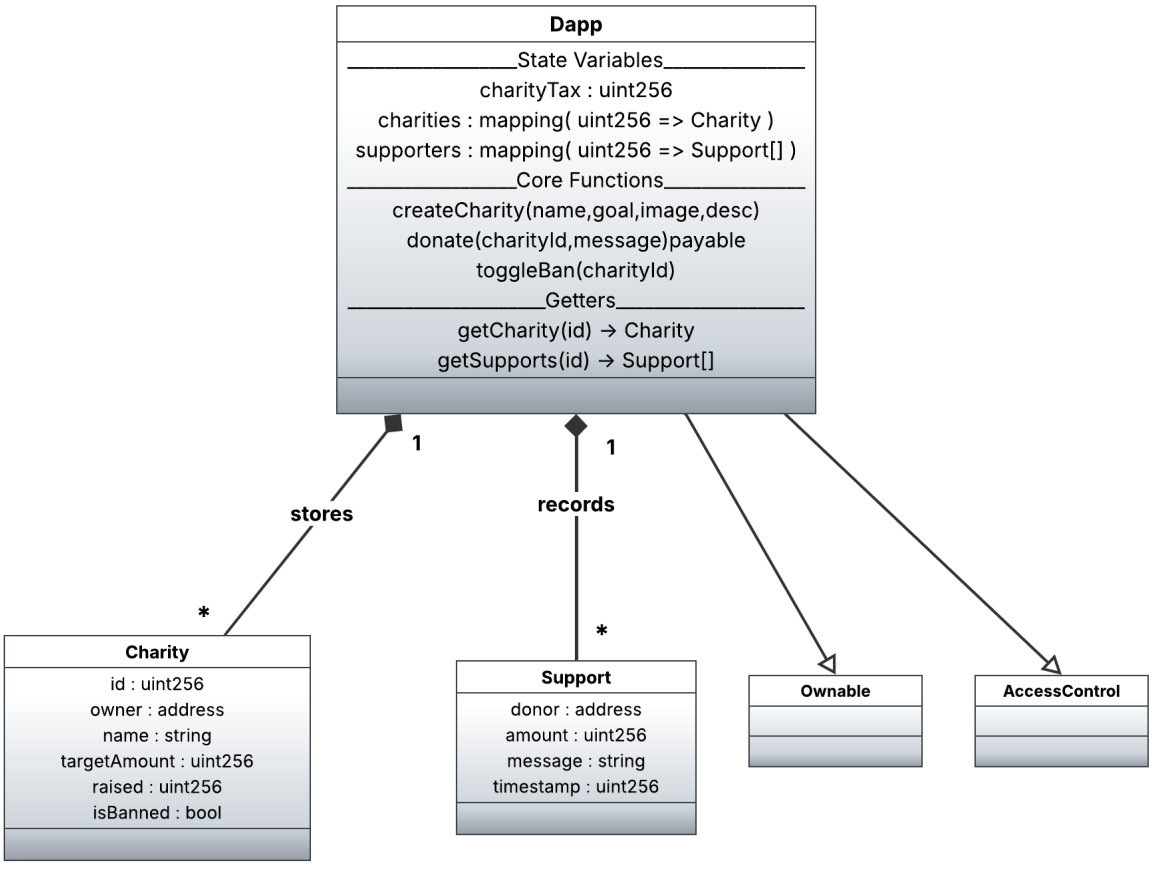

Figure 3.8: Smart contract class diagram

3.6 Conclusion

This conception chapter has established the foundational design of the blockchain-based donation tracking system, defining its architecture through comprehensive use case diagrams that outline user interactions (donations, charity management), class diagrams that structure smart contract components (Charity and Donation data models), and sequence diagrams that detail critical workflows (wallet connection, transaction processing). By integrating blockchain specific elements such as wallet authentication, immutable transaction records, and role-based access control the design ensures transparency, security, and scalability while maintaining a user-friendly experience. This blueprint not only aligns with functional requirements but also sets the stage for implementation, ensuring the system is both technically robust and adaptable for future enhancements.

Chapter 4

Implementation

4.1 Introduction

In this chapter, we present a comprehensive examination of our system's implementation process, providing a detailed technical account of how the theoretical framework and architectural designs established in previous chapters were transformed into a functional blockchain-based donation platform. We systematically document the development journey, beginning with the core system components and progressing through integration challenges to final deployment. The discussion will thoroughly analyze the various methodologies employed during implementation, including our approach to testing and quality assurance. Furthermore, we present a critical evaluation of the implementation outcomes, demonstrating how each component contributes to achieving the system's overarching goals of transparency, security, and usability.

4.2 Development Tools

This system leverages a decentralized tech stack where users interact via a web browser with a ReactJS client application that connects to MetaMask (handling private keys and wallet operations through its browser plugin). The frontend, built with Web3.js, communicates with smart contracts (written in Solidity and deployed via Truffle) on the Ethereum blockchain, while Ganache simulates the blockchain environment for testing. For data storage, IPFS manages off-chain files (like charity metadata) in a decentralized manner, with NodeJS serving as the backend bridge between these components, ensuring seamless integration of on-chain transactions and off-chain data.

This figure below represents an overview of our implementation architecture. It contains all the components .

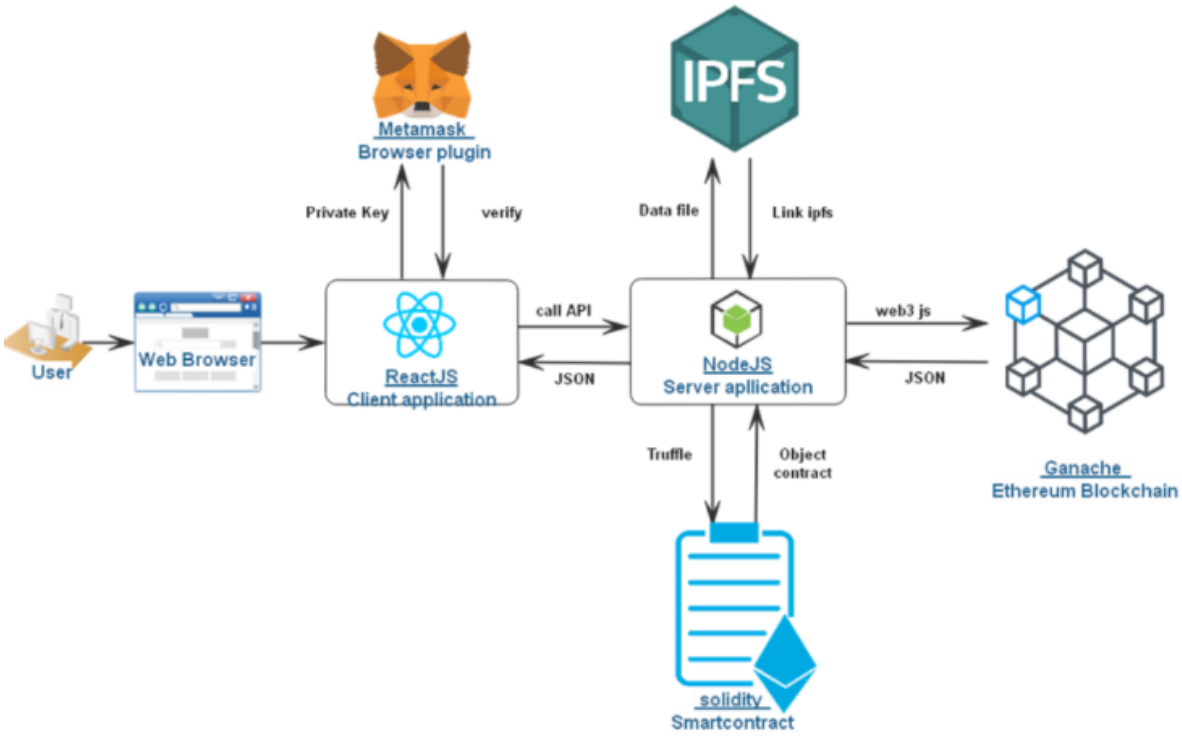

Figure 4.1: Implementation Architecture[18]

4.2.1 Frontend Development

The frontend stack prioritizes developer efficiency and blockchain compatibility through React.js for modular interfaces, Next.js for performance optimization, and Wagmi/Ethers.js for seamless Ethereum integration. These choices reflect contemporary dApp development best practices.

1. **React JS**

   ReactJS is a popular open-source JavaScript library for building interactive user interfaces. Developed by Facebook, it uses a component-based architecture that allows developers to create reusable UI elements and efficiently manage application state. React's virtual DOM enables fast rendering updates, while its declarative approach simplifies coding by automatically handling UI changes when data updates. This makes React ideal for modern web applications that require dynamic content and smooth user experiences.[19]

   In our blockchain donation platform, React serves as the foundation for the frontend interface that connects users to the Ethereum network. We utilize React to build the wallet connection system, display real-time donation data from smart contracts, and manage the complete donor journey - from browsing charities to confirming transactions. Its component structure perfectly matches our needs, allowing us to create reusable donation forms, charity profile cards, and transaction status modals that all stay synchronized with on-chain activity through Web3.js integration.

2. **Next JS**

Next.js is a React framework that enables server-side rendering, static site generation, and API routes for building production-ready web applications. It extends React's capabilities with built-in optimizations like automatic code splitting, image optimization, and hybrid static/dynamic rendering - making it ideal for SEO-friendly, high-performance applications that require both client-side interactivity and server-side processing.[20]

In our donation platform, Next.js powers the frontend architecture by: providing server-rendered pages for fast initial loading of charity listings, hosting API routes that securely interact with our Node.js backend and blockchain nodes, and enabling dynamic client-side transitions for wallet-connected features like donation processing. We specifically leverage its file-based routing system for campaign pages , its Image component for optimized charity logo loading from IPFS, and its API routes to proxy blockchain requests while protecting sensitive API keys.

3. **Wagmi**

Wagmi is a modern React Hooks library for Ethereum development that simplifies Web3 integration with built-in wallet connections, contract interactions, and transaction management. It provides optimized hooks like useAccount, useBalance, and useContractWrite that handle real-time state synchronization, error recovery, and network switching automatically - significantly reducing boilerplate code compared to direct Web3.js implementations.[21]

In our donation platform, Wagmi replaces traditional Web3.js integration to: streamline MetaMask wallet connections with auto-reconnect support, manage donation transactions with built-in status tracking (loading/success/error states), and efficiently read smart contract data (charity details, donation history) through its caching system. We specifically leverage its usePrepareContractWrite hook to estimate gas fees before transactions and useWaitForTransaction for real-time donation confirmation updates in the UI.

4. **Ethers JS**

Ethers.js is a lightweight JavaScript library designed for interacting with the Ethereum blockchain, offering a streamlined alternative to Web3.js. It provides essential tools for connecting to wallets, sending transactions, and reading smart contract data through a modular architecture with clear separation between providers (network connections), signers (wallet operations), and contracts (ABI interactions). Its TypeScript-native design and smaller bundle size make it ideal for performance-sensitive dApps.[22]

In our donation platform, Ethers.js handles the core blockchain interactions by: processing donation transactions through MetaMask with precise gas control, decoding real-time events from your smart contracts (like DonationReceived), and fetching charity statistics from the blockchain with optimized batch queries. We specifically use its Contract class to interact with your deployed DappFund instance, BigNumber utilities for accurate ETH value handling, and Provider API to monitor network status - ensuring reliable transactions even during gas price fluctuations.

5. **Web3 JS**

Web3.js is a JavaScript library that serves as the primary interface between web applications and blockchain networks, enabling direct interaction with Ethereum nodes. It provides essential tools to send transactions, read smart contract data, and listen to blockchain events through JSON-RPC calls, abstracting the complexity of direct node communication while maintaining security and decentralization principles.[23]

In our donation platform, Web3.js acts as the critical link between the React frontend and Ethereum blockchain, specifically handling: wallet integration through MetaMask's provider injection, real-time donation transaction processing via contract method calls, and event listening for instant UI updates when donations occur. Its seamless compatibility with React's state management allows us to dynamically display transaction statuses, wallet balances, and charity funding progress while ensuring all operations adhere to Web3 security best practices.

6. **Tailwind CSS**

Tailwind CSS is a highly customizable, utility-first CSS framework that enables developers to rapidly build responsive and consistent user interfaces by composing small, reusable classes directly in HTML or JSX. Unlike traditional CSS frameworks (e.g., Bootstrap), Tailwind provides low-level utility classes (like flex, pt-4, or bg-blue-500) instead of pre-designed components, giving developers complete control over styling without writing custom CSS.[24]

4.2.2 Smart Contract Development

Developed with security and efficiency as primary constraints, these contracts were built using industry-standard tools to ensure reliability and maintainability.

1. **Solidity**

Solidity is an object-oriented, high-level programming language specifically designed for writing smart contracts that run on the Ethereum Virtual Machine (EVM). As a statically-typed language with syntax similar to JavaScript, it enables developers to create secure, decentralized applications with features like inheritance, libraries, and complex user-defined types. Its compiler enforces strict security patterns to prevent common vulnerabilities like reentrancy attacks and integer overflows, while native support for cryptographic operations makes it ideal for financial applications like donation platforms.[25]

In our donation system, Solidity implements the core smart contract logic through: the Dapp contract that manages charity registration and donation processing, custom data structures (CharityStruct, SupportStruct) for on-chain storage, and event emissions (DonationReceived) for real-time frontend updates. We leverage Solidity's access control modifiers (onlyOwner) for admin functions, payable methods for ETH transfers, and gas optimizations like fixed-size arrays to keep transaction costs low. The compiled bytecode integrates seamlessly with our Web3 frontend through the ABI interface.

2. **Truffle**

   Truffle is a development framework for Ethereum that provides a complete suite of tools for smart contract development, testing, and deployment. It simplifies blockchain workflows with built-in compilation, linking, and migration features, while offering a local development blockchain for testing through Ganache. Truffle's standardized project structure and automated testing capabilities make it ideal for building production-ready decentralized applications.[26]

   In our donation platform, Truffle serves as the backbone of our smart contract development process by: compiling and optimizing Solidity contracts, managing deployment scripts to both testnets and mainnet, and facilitating comprehensive testing of donation flows and admin functions through JavaScript/TypeScript test suites. Its integration with Web3.js and Ganache allows us to simulate real-world scenarios, including gas cost analysis and transaction failure handling, before deploying contracts to live networks. We specifically leverage Truffle's migration system to ensure consistent contract addresses across environments and its console for direct contract interaction during development.

3. **Hardhat**

   Hardhat is a professional Ethereum development environment that streamlines smart contract coding, testing, and deployment. It provides built-in tasks for compilation, debugging, and network management, along with a local Ethereum network for development. Hardhat's console.log capability, stack traces, and TypeScript support make it ideal for complex dApps requiring rigorous testing before mainnet deployment.[27]

   In our donation platform, Hardhat serves as the core development toolkit by: running automated tests for all smart contract functions (donations, charity creation, admin controls), verifying contract bytecode on Etherscan, and scripting deployment workflows across testnets and mainnet. We specifically use its console.log to debug donation transactions, the Hardhat Network fork for simulating mainnet conditions, and its plugin system to integrate with Ethers.js and Waffle for comprehensive testing coverage.

4. **Remix IDE**

   Remix IDE is an open source web and desktop application. It fosters a fast development cycle and has a rich set of plugins with intuitive GUIs. Remix IDE allows developing, deploying and administering smart contracts for Ethereum Blockchain. We will use it for developing process only of the smart contract.[28]

4.2.3 Blockchain and Testing

To ensure reliable and secure blockchain interactions, our Dapp employs a robust testing framework and development blockchain environment. This infrastructure enables thorough validation of smart contract functionality and user transactions before deployment to mainnet.

1. **Ganache**

   Ganache is a personal Blockchain for rapid Ethereum and Corda distributed application development. You can use Ganache across the entire development cycle; enabling you to develop, deploy, and test your Dapps in a safe and deterministic environment.[29]

   In our donation platform, Ganache serves as the testing sandbox where we: simulate donation transactions with zero latency, validate gas consumption patterns before mainnet deployment, and test edge cases like failed transactions or contract reverts in a risk-free environment. Its seamless integration with Truffle allows us to run automated tests of our complete donation workflow - from charity creation to fund distribution - while the built-in block explorer gives real-time visibility into contract states and transaction details during debugging.

2. **MetaMask**

   MetaMask is a cryptocurrency wallet and gateway to blockchain applications, available as a browser extension and mobile app. It securely manages users' private keys, enables interaction with Ethereum-based dApps through a simple interface, and supports transactions across multiple networks (Ethereum, Polygon, etc.). By injecting the window.ethereum provider into web applications, MetaMask handles all cryptographic operations client-side, ensuring users retain full control of their funds and identity without centralized intermediaries.[30]

   In our donation platform, MetaMask serves three critical functions: authenticating donors through wallet connection , signing donation transactions before submission to the blockchain, and displaying real-time gas fee estimates and transaction confirmations. We specifically optimize for MetaMask's standards , displaying human-readable transaction details through WalletConnect, and handling common user scenarios like network switching (e.g., prompting donors to switch from Ethereum Mainnet to another for lower fees).

4.2.4 Decentralized Storage

To achieve true decentralization while maintaining data availability, our Dapp implements IPFS (InterPlanetary File System) for off-chain storage needs. This approach provides:

1. **IPFS**

   IPFS (InterPlanetary File System) is a decentralized protocol for storing and sharing files across a peer-to-peer network, using content-addressing to uniquely identify each file with a cryptographic hash. Unlike traditional location-based storage (URLs), IPFS ensures data remains available even if individual nodes go offline, making it ideal for permanent, censorship-resistant storage of charity metadata like images, descriptions, and documents.[31]

   In our donation platform, IPFS handles off-chain data storage by: hosting charity profile images and detailed descriptions (saving blockchain storage costs), providing immutable audit trails for charity verification documents, and enabling fast content delivery through its distributed network. We integrate IPFS via the web3.storage API, which automatically pins files to multiple nodes, and references the returned Content Identifiers (CIDs) in our smart contracts - creating a hybrid architecture where on-chain transactions link to off-chain data without sacrificing transparency.

4.2.5 Backend Runtime

The server-side infrastructure of our Dapp leverages Node.js to create a robust runtime environment that bridges traditional web services with blockchain operations.

1. **Node JS**

   Node.js is a JavaScript runtime environment that executes server-side code outside web browsers, using Google Chrome's V8 engine. It enables building scalable network applications through its event-driven, non-blocking I,O model, which efficiently handles multiple concurrent operations. Node.js revolutionized backend development by allowing JavaScript to be used for both client and server-side programming, with access to a vast ecosystem of packages via npm (Node Package Manager).[32]

   In our blockchain donation platform, Node.js powers the backend services that bridge our React frontend with blockchain networks and IPFS storage. We use it to create API endpoints that fetch aggregated donation data, process off-chain computations before submitting to smart contracts, and manage file uploads to IPFS for charity metadata storage. Its lightweight architecture and asynchronous capabilities make it ideal for handling real-time blockchain events while maintaining smooth communication between our React interface, wallet providers, and decentralized storage solutions.

4.3 Project Overview

Our platform is named TASADDAQ , an Arabic word meaning "Give in charity". The name reflects the central goal of the platform: to simplify and secure charitable giving through blockchain technology. The logo of TASADDAQ (Figure 4.2) symbolizes trust, decentralization, and social solidarity, conveying the values of transparency, fairness, and ethical donation in the digital era.

Figure 4.2: TASADDAQ logo

Like we said before, there are challenges, but TASADDAQ addresses these challenges by introducing a decentralized charity system that empowers both donors and beneficiaries through smart contracts, reducing reliance on intermediaries and increasing transparency. By leveraging smart contracts and the ERC-20 token standard, TASADDAQ ensures that every transaction is immutable and verifiable on the blockchain.

4.4 Implementation details

This section provides a comprehensive technical breakdown of the platform's implementation, translating the architectural designs into a fully operational system. The development focused on multi-interconnected layers:

4.4.1 Ganache – Local Blockchain Environment

The development process begins by launching Ganache. Upon opening, Ganache initializes a local blockchain instance with a set of pre-funded Ethereum accounts, as shown in Figure 4.3. These accounts are essential for testing transactions, deploying smart contracts, and interacting with the application in a controlled environment. Each account is provided with a unique address and a balance of 100 ETH, which simplifies the simulation of donation flows within the charity platform.[33]

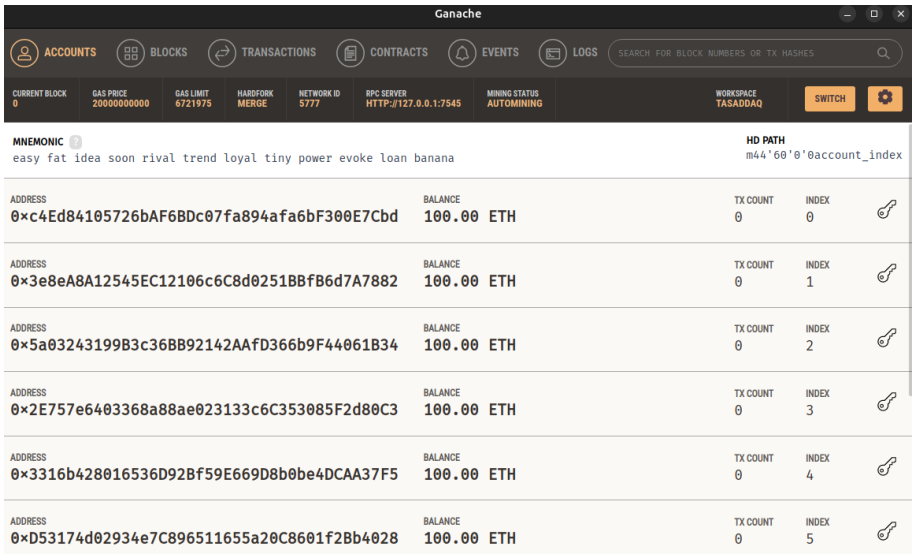

Figure 4.3: The list of pre-funded test accounts and their respective balances

Ganache runs on a local server (typically at http://127.0.0.1:7545) and is configured to work seamlessly with Truffle and MetaMask.

This local blockchain environment ensures that all features of the platform, including ERC-20 token interactions and donation tracking, are thoroughly tested before deployment to a live network.

4.4.2 Running the Project Locally

After initializing the local blockchain environment with Ganache, the next step is to run our project on a local development server. This allows us to test the full functionality of the charity platform, including user interactions, smart contract integration, and donation processes.

The project can be executed on either Linux or Windows, and the commands used vary slightly depending on the operating system.

After executing, the application runs locally, typically accessible via http://localhost:3000. At this point, the website is connected to the Ganache local blockchain and ready for interaction.

4.4.3 Frontend of TASADDAQ

The First page serves as the introductory interface of the TASADDAQ platform. It provides a clear overview of the platform's mission, values, and functionalities, emphasizing transparency, ethical giving, and the use of blockchain to build trust in charitable donations, as shown in the figure below.

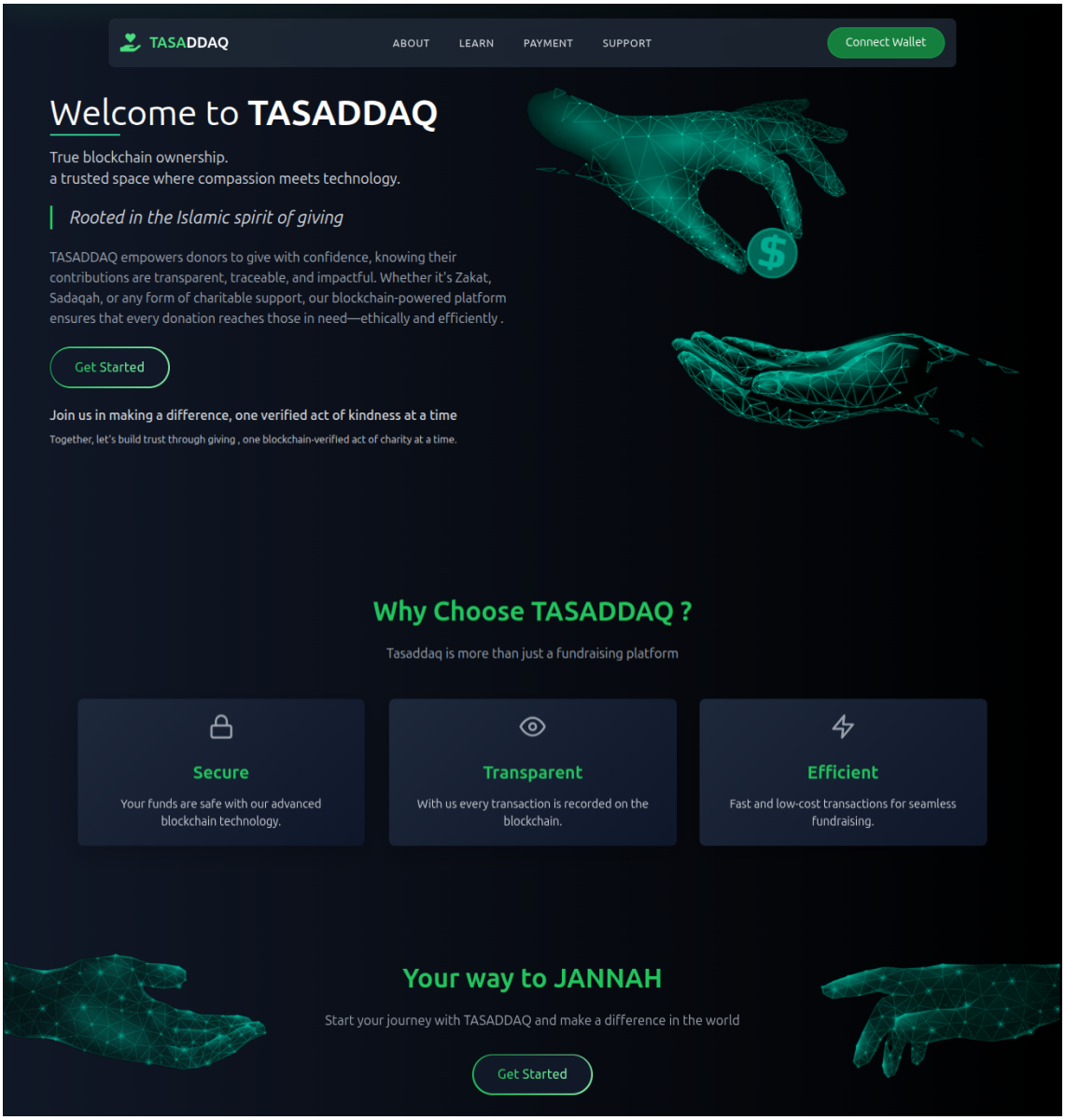

Figure 4.4: Home page

Upon launching the application, users are greeted by a modern and visually engaging landing page. The header includes navigation links such as About, Learn, Payment, and Support, along with a prominent Connect Wallet button that encourages users to initiate blockchain-based interactions.

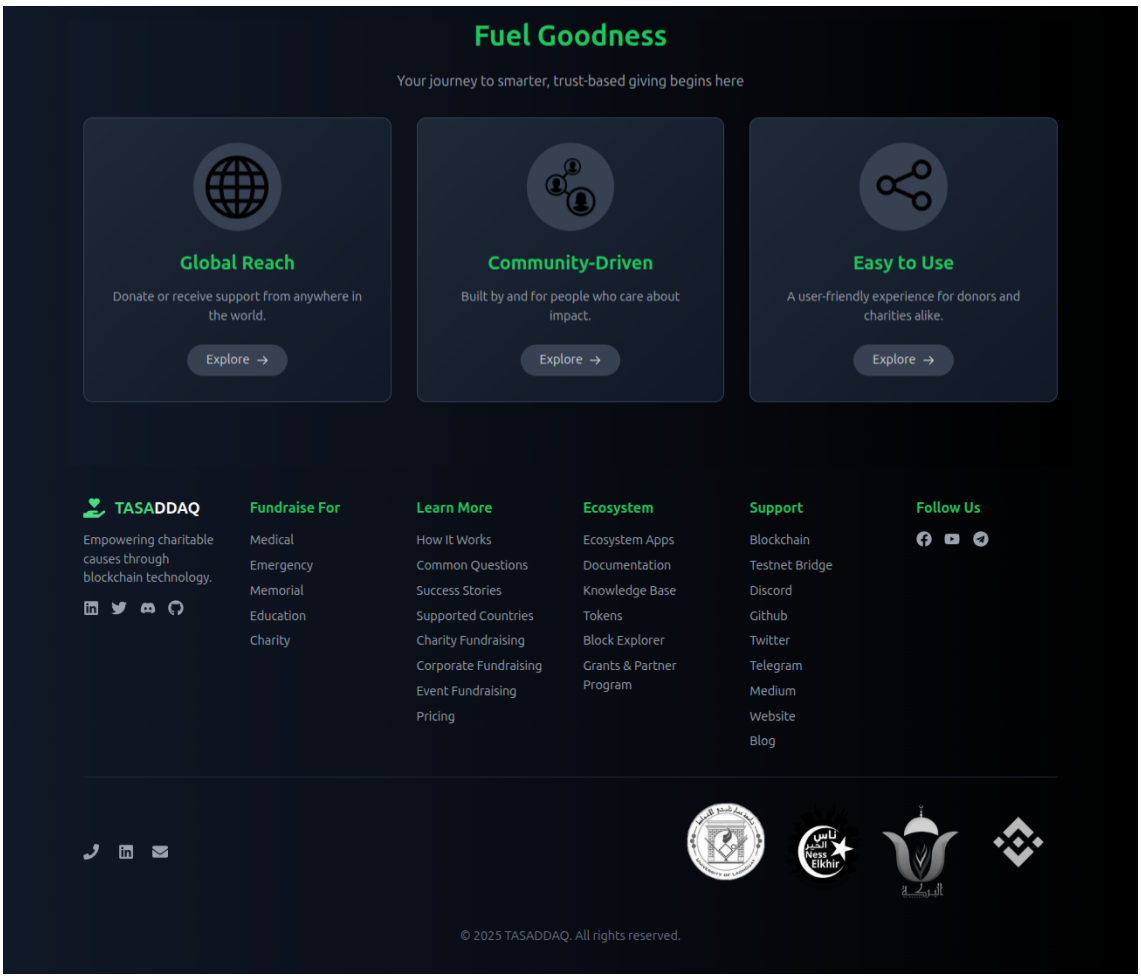

Figure 4.5: The footer

The main body of the page highlights TASADDAQ's unique value proposition:

- Rooted in the Islamic spirit of giving, it appeals to users motivated by faith-based philanthropy.
- A "Get Started" call-to-action button guides users toward engagement.
- Three key features Secure, Transparent, and Efficient are presented in cards with iconography to enhance visual understanding.
- Additional sections promote the platform's Global Reach, Community-Driven model, and User-Friendly experience.
- The footer (Figure 4.5) contains important platform links, social media icons, and institutional logos, reinforcing credibility and ecosystem connections.

Technical implementation :

- **Frontend Framework :** Developed using Next.js and styled with Tailwind CSS for responsive design and clean layout.
- **Web3 Integration :** The Connect Wallet button uses ethers.js to enable MetaMask or other Web3 wallet connections.
- **Responsiveness :** The layout adapts to different screen sizes, ensuring accessibility on desktop and mobile devices.
- **Accessibility :** Semantic HTML tags and high-contrast design support usability and accessibility.

This page establishes the first impression for users, communicates the ethical and technical foundations of the platform, and offers a gateway to deeper user engagement.

4.4.4 Authentication Page

This page allows users to access the TASADDAQ platform through a secure authentication system. It provides two methods of login traditional email and password and blockchain-based login via a Web3 wallet ensuring accessibility for both non-technical users and crypto savvy users, as shown in the figure below :

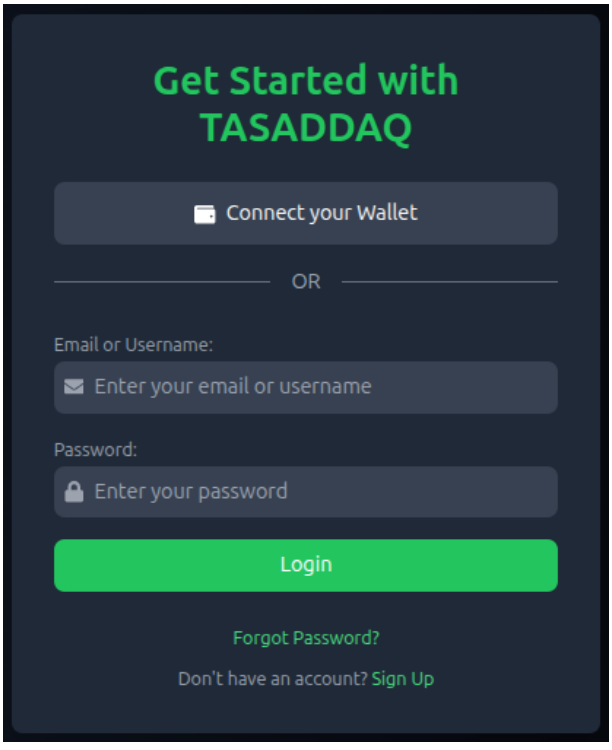

Figure 4.6: Authentication Page

When the user clicks the "Get Started" button on the Home page, the system redirects them to the authentication interface. The interface offers two clearly defined options:

1. **Web3 Wallet Login :**
   A prominent “Connect Wallet” button allows users to authenticate using MetaMask or any compatible Web3 wallet. Once the wallet is connected, the user’s public wallet address is retrieved and used as their blockchain-based identity. This option enables seamless interaction with smart contracts for donation and fund management.

2. **Email and Password Login, Registration :**
   For users who prefer a traditional approach, the platform offers standard login and registration forms. New users can register by providing a name, email, and password, while returning users can log in using their credentials.

Both login methods are integrated into the same user-friendly interface, enabling smooth navigation and a clear user experience.

Authentication System :

- **Web3 Login :**Implemented using ethers.js and MetaMask, with account validation through wallet signatures
- **Traditional Login :** Powered by Node.js and the backend database for storing user credentials securely

This hybrid login model ensures that TASADDAQ is inclusive for all users those who are familiar with blockchain as well as those who are not—while maintaining a high standard of security and usability.

4.4.5 Home Page

The home page is the user’s main interface after a successful login (Figure 4.7). It acts as the central hub where users can browse existing charity projects, initiate new ones, or simply explore the mission and values of TASADDAQ. This page blends functionality and inspiration, making it both informative and action-oriented.

Once authenticated, the user is redirected to this main dashboard interface, which provides the following key functionalities and elements:

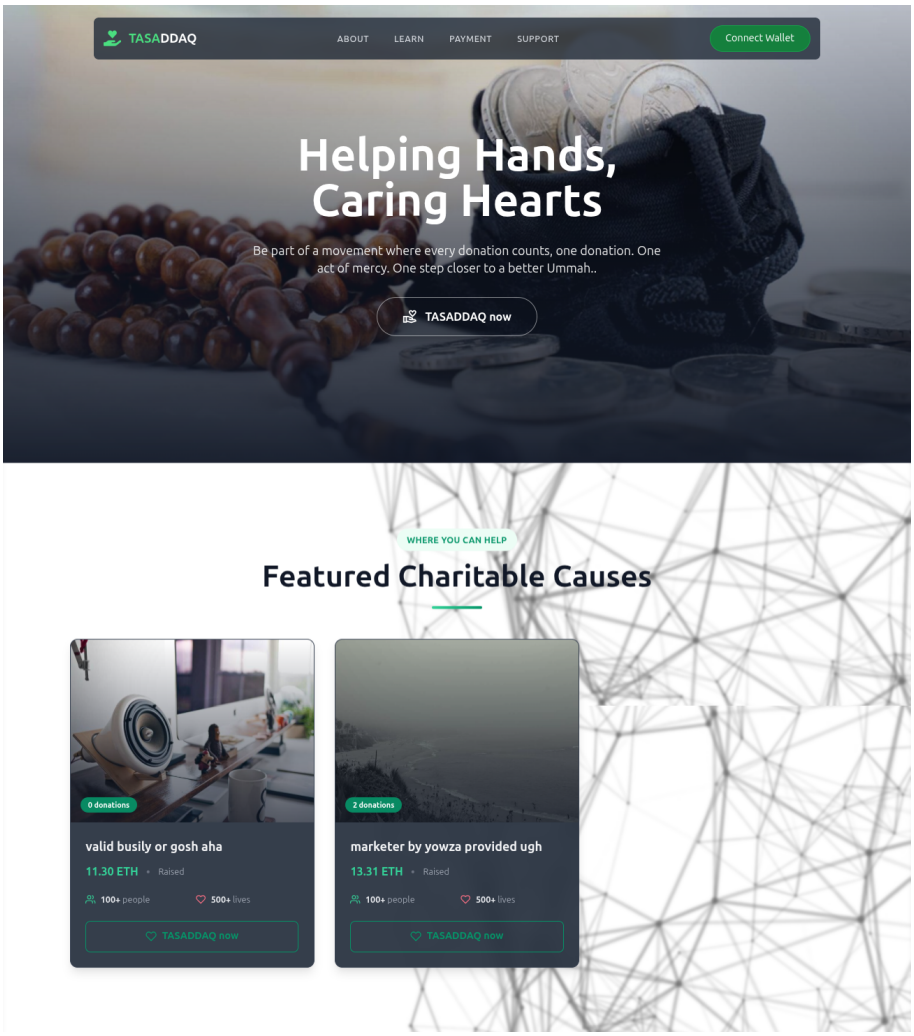

Figure 4.7: Home Page

1. **Wallet Connection :**

   In the top-right corner, users have access to the "Connect Wallet" button. This functionality enables users to link their Web3 wallets (e.g., MetaMask) at any point—whether immediately after login or later when they are ready to interact with smart contracts (e.g., creating or funding a charity). This flexibility improves user autonomy and experience.

2. **Call to Action "TASADDAQ now" :**

   This button serves as a shortcut for users who wish to create a new charity project. Clicking it redirects the user to the "Create Charity" interface, where they can fill out relevant details and deploy their campaign on the blockchain. This promotes active engagement from users who want to contribute not just through donations, but by starting their own initiatives.

3. **Existing Charitable Projects – Featured Cards :**

Below the header section, users are presented with a list of active or featured charitable campaigns displayed in card format. Each card includes: Charity Name and Image, amount Raised (in ETH), number of Donors, lives Impacted, call to Action Button ("TASADDAQ now") to donate directly.

These cards are dynamically fetched from the backend (or blockchain) and serve as direct entry points for users who want to make a donation. This section exemplifies the platform's transparency and its blockchain-enabled tracking of donations, as shown in the figure below.

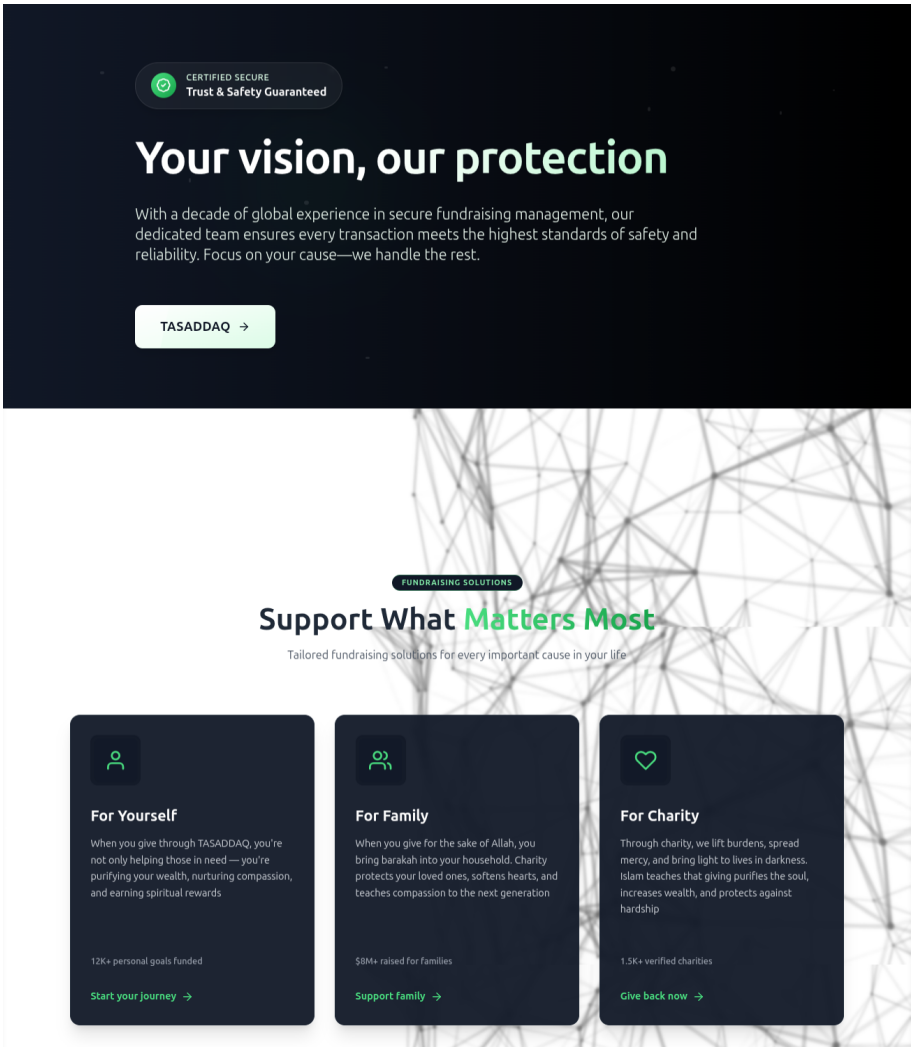

Figure 4.8: Second half of home Page

4. **Platform Values and Trust Elements :**

Further down the page, informational sections highlight the values and guarantees provided by TASADDAQ:

- **"Your Vision, Our Protection"**explains the platform's secure, blockchain-based structure.
- **"Support What Matters Most"** breaks down the available donation categories:

- For Yourself (Zakat and personal spiritual growth)
- For Family (charity for household blessings)
- For Charity (general community causes)

This educational and motivational content fosters user trust and emotional connection with the platform.

4.4.6 Wallet Connection Interface

To perform any blockchain-based operation (such as creating a charity project or making a donation), users must first connect a crypto wallet, as shown in the figure below. This ensures secure authentication and enables interaction with the Ethereum blockchain through smart contracts. For more details :

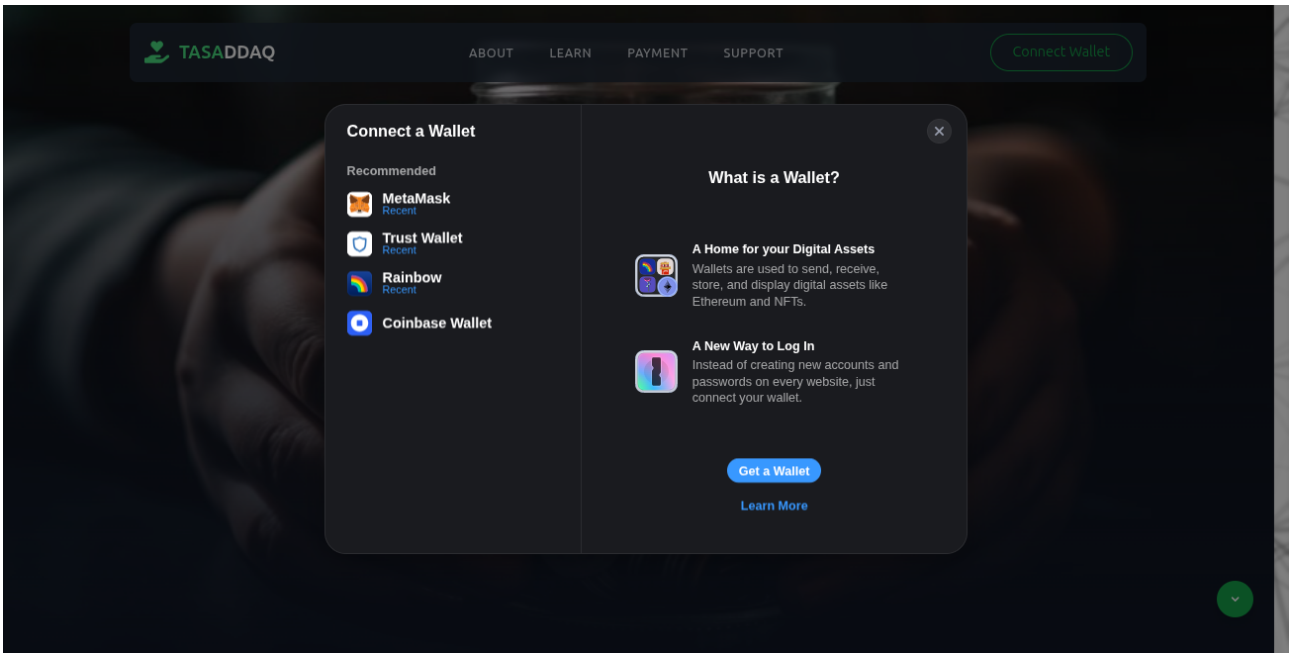

Figure 4.9: Wallet selection interface for TASADDAQ platform

1. **Wallet Connection Flow :**

   Users can connect their wallet in one of two ways: By clicking "Connect Wallet" from the top-right navigation bar at any time. Or, when attempting to initiate blockchain actions like creating a charity or donating, a notification popup prompts them to connect their wallet first if they haven't done so.

2. **Modal Window Components :**

   The wallet connection popup is designed with clarity and accessibility in mind. It consists of:

- **Left Section – Recommended Wallets :** Lists the most commonly used Ethereum-compatible wallets, including: MetaMask, Trust Wallet, Rainbow, Coinbase Wallet
- **Right Section – Wallet Explanation :** Educates users who are new to blockchain or Web3. It provides a simple explanation of:
  - What is a Wallet? A digital tool to store, send, and receive assets like ETH or NFTs.
  - Why Connect? Allows for secure login without traditional passwords. Enables smart contract interaction.

3. **MetaMask Login Prompt :**

After clicking the "Connect Wallet" button, users are presented with a list of supported wallets. In this example, the user selects MetaMask.
MetaMask automatically opens and asks the user to log in (if not already logged in) and choose an Ethereum account.

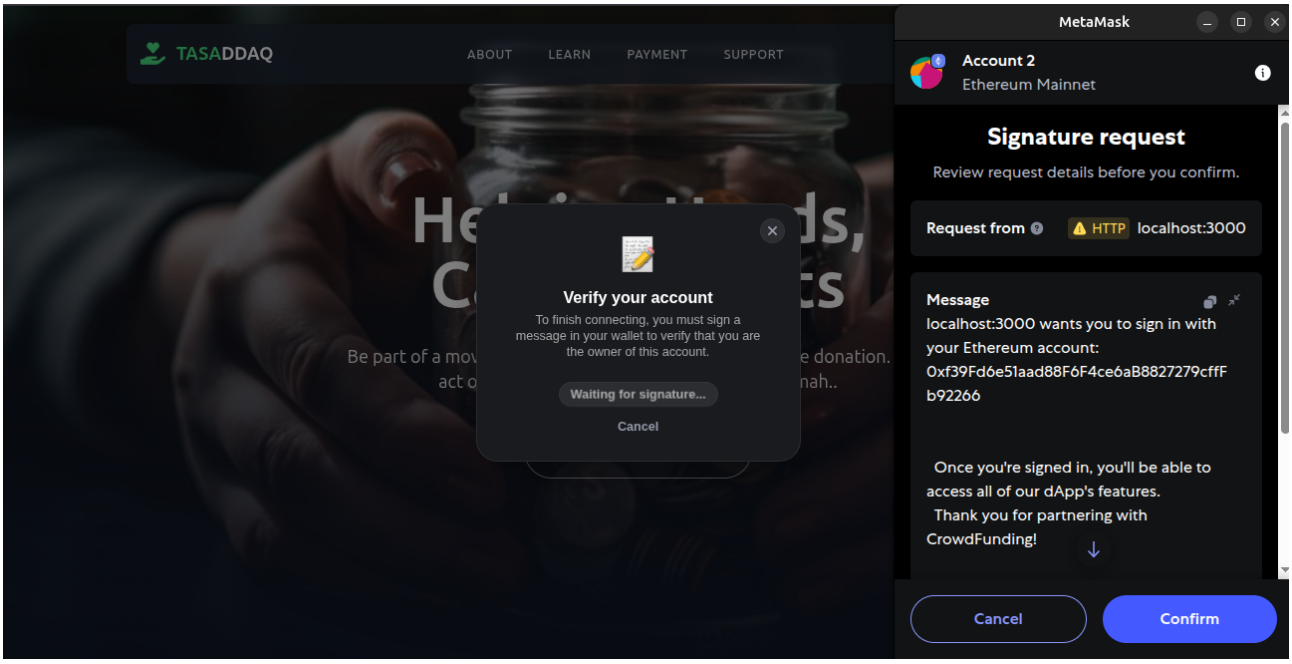

Figure 4.10: MetaMask signature prompt for DApp login

4. **Signature Request Popup :**

Once an account is selected, MetaMask displays a signature request window (Figure 4.10). This is not a transaction, it's a method of securely verifying the user's identity :

- **Proof of Ownership :** Signing the message proves that the user controls the private key for the Ethereum address they are connecting with.
- **Secure Login :** t avoids traditional logins or passwords, reducing risk of phishing or credential leaks.
- **Session Access :** Once signed, the DApp enables full access to its blockchain features such as: Making contributions using ETH or ERC-20 tokens

4.4.7 Charity Creation Page

Allows registered users to create and deploy a new charity campaign as a smart contract on the blockchain. This ensures immutable record-keeping and transparent fund allocation.

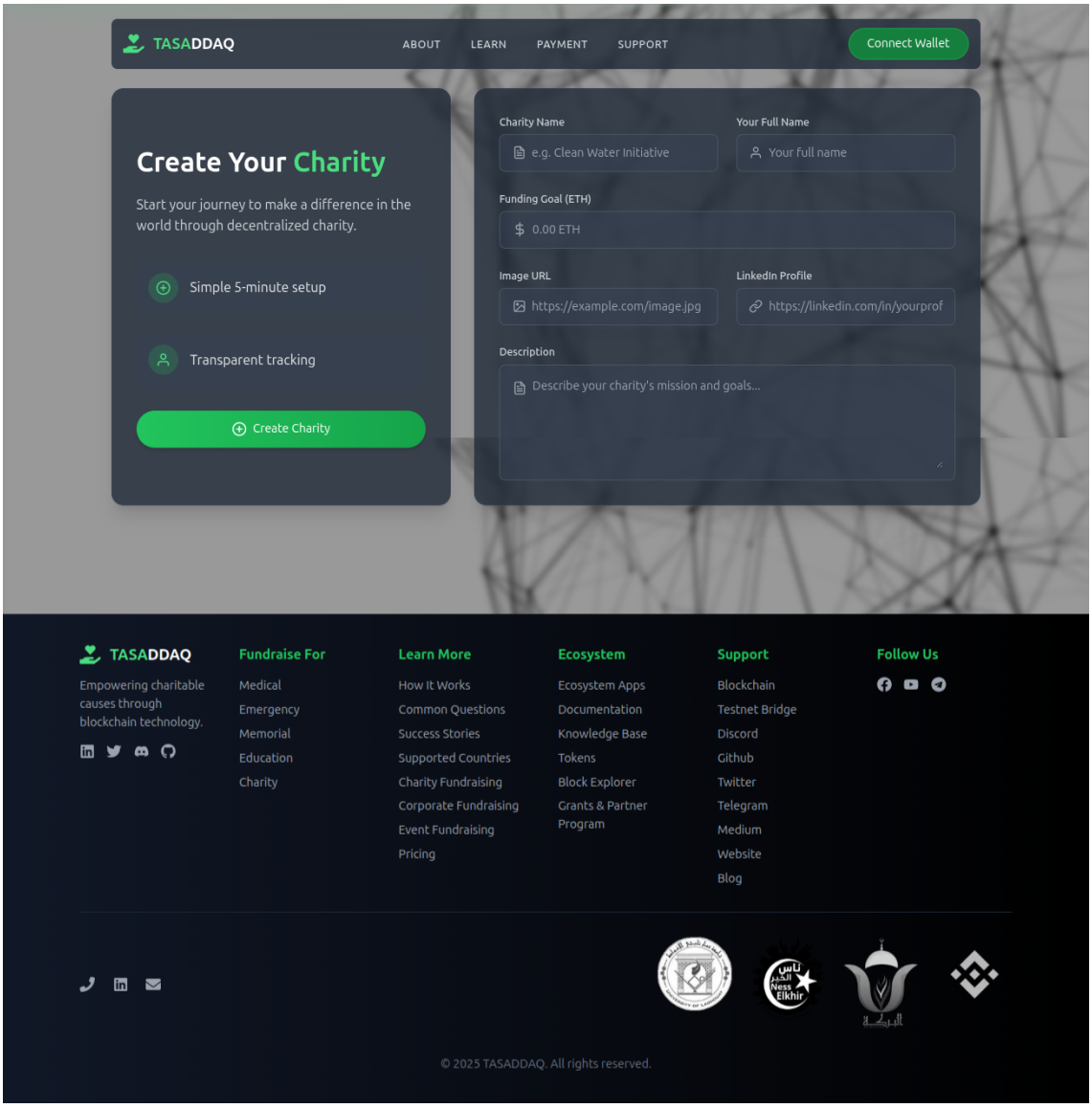

Figure 4.11: Charity Creation Page

On the Home Page, once the user is authenticated via wallet, they can click the “TASADDAQ now” button. This action redirects them to the Create Charity Page.

This figure 4.11 presents a simple form where the user must enter their information

After submitting charity details, users must confirm a blockchain transaction to deploy the charity's smart contract (Figure 4.12). This ensures on-chain immutability and gas fee payment. For more details:

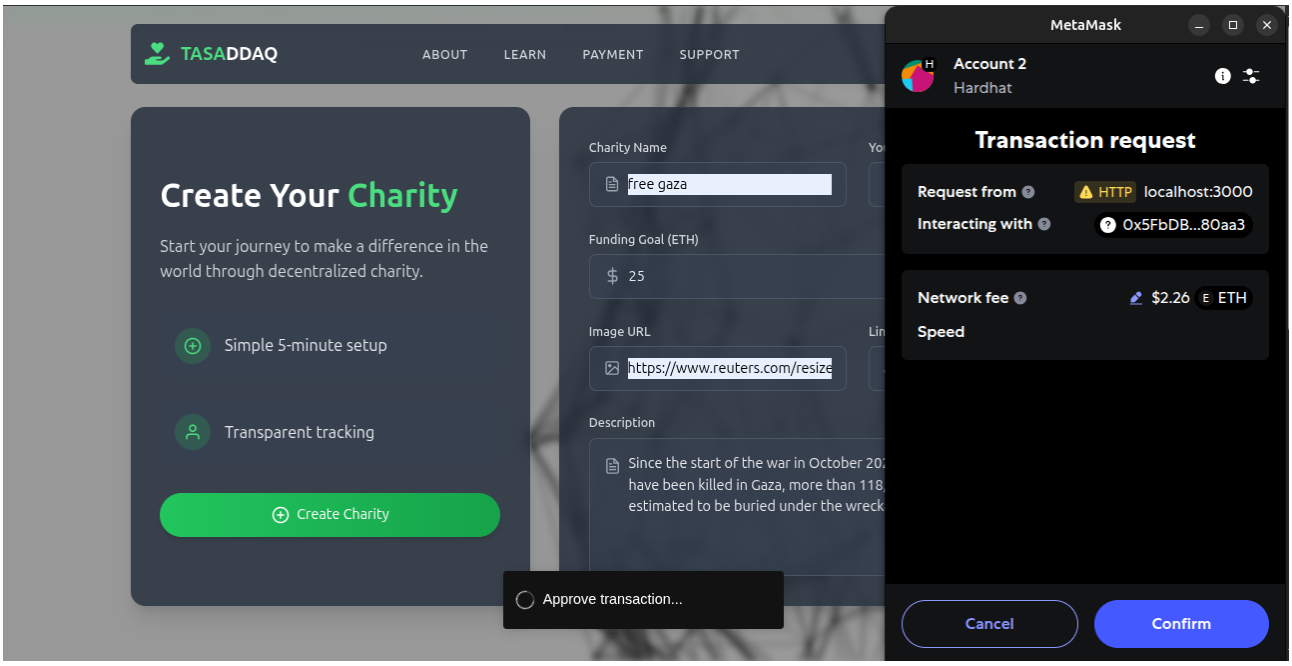

Figure 4.12: Charity Creation Transaction Flow

1. **Transaction Request Window :**
   Upon submitting charity details, the wallet MetaMask displays a transaction request window, confirming the interaction with the platform's smart contract at 0xOsSFbDB...80aa3 . The request originates from [HTTP] localhost:3000 (development environment, replaced post-deployment) and includes a dynamically calculated gas fee , adjustable for processing speed. Users must approve this transaction to deploy the charity's smart contract, ensuring on-chain immutability and transparency.

2. **User Confirmation :**
   Upon user confirmation, the system automatically deploys the charity's smart contract to the blockchain, generating an immutable transaction hash (e.g., 0x123...abc) that serves as both proof of creation and permanent on-chain record. This hash enables full transaction traceability through blockchain explorers, ensuring complete transparency for all platform participants while maintaining security through cryptographic verification of each deployment.

3. **Blockchain Integration :**
   The deployment mechanism requires payment of network fees (denominated in ETH or BNB depending on the underlying chain) to compensate validators, with gas costs dynamically adjusting based on real-time network congestion - users may optionally prioritize their transaction by increasing the gas price. This entire process, from data encoding to contract deployment, establishes an

auditable, tamper-proof record of the charity's formation while leveraging the blockchain's consensus mechanisms for trustless verification.

4. **Display of Created Charities :**

   After successfully creating a charity, the new project is immediately added to the list of existing campaigns on the home page, where it appears as a card displaying key details such as the charity name, image, funding goal, and a donation button. However, when the user select his projects , the platform filters the view to show only the projects they have created, offering a personalized dashboard experience. This design ensures public visibility of all campaigns to encourage donations, while also providing users with a focused view to manage and monitor their own charity submissions efficiently.

4.4.8 Donation Page

If the logged-in user is a donor, they will see all available charity campaigns displayed as cards on the home page, including those created by other users, as shown in Figure 4.13. Each card provides key information such as the charity name, image, funding goal, and a donation button.

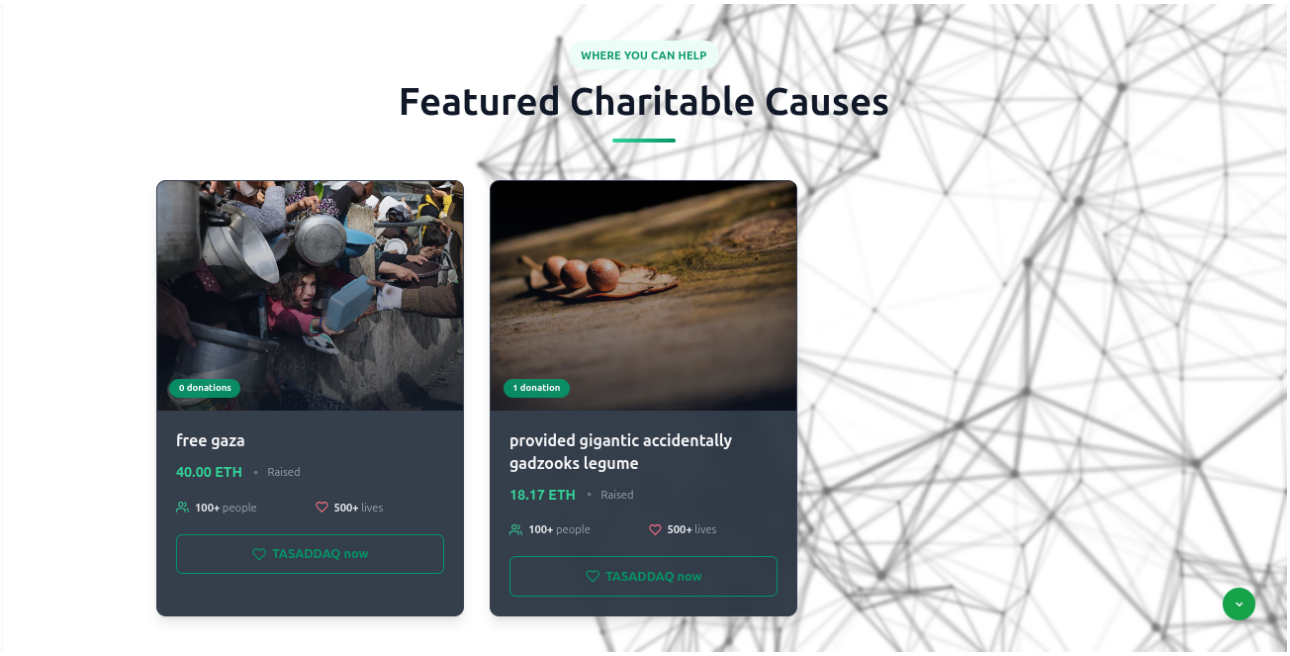

Figure 4.13: Donation Page

This ensures that donors have full access to explore and support any active project on the platform. Unlike the creator view, donors are not limited to seeing only their own data; they can browse and contribute to all visible charities, including those previously created by themselves or others.

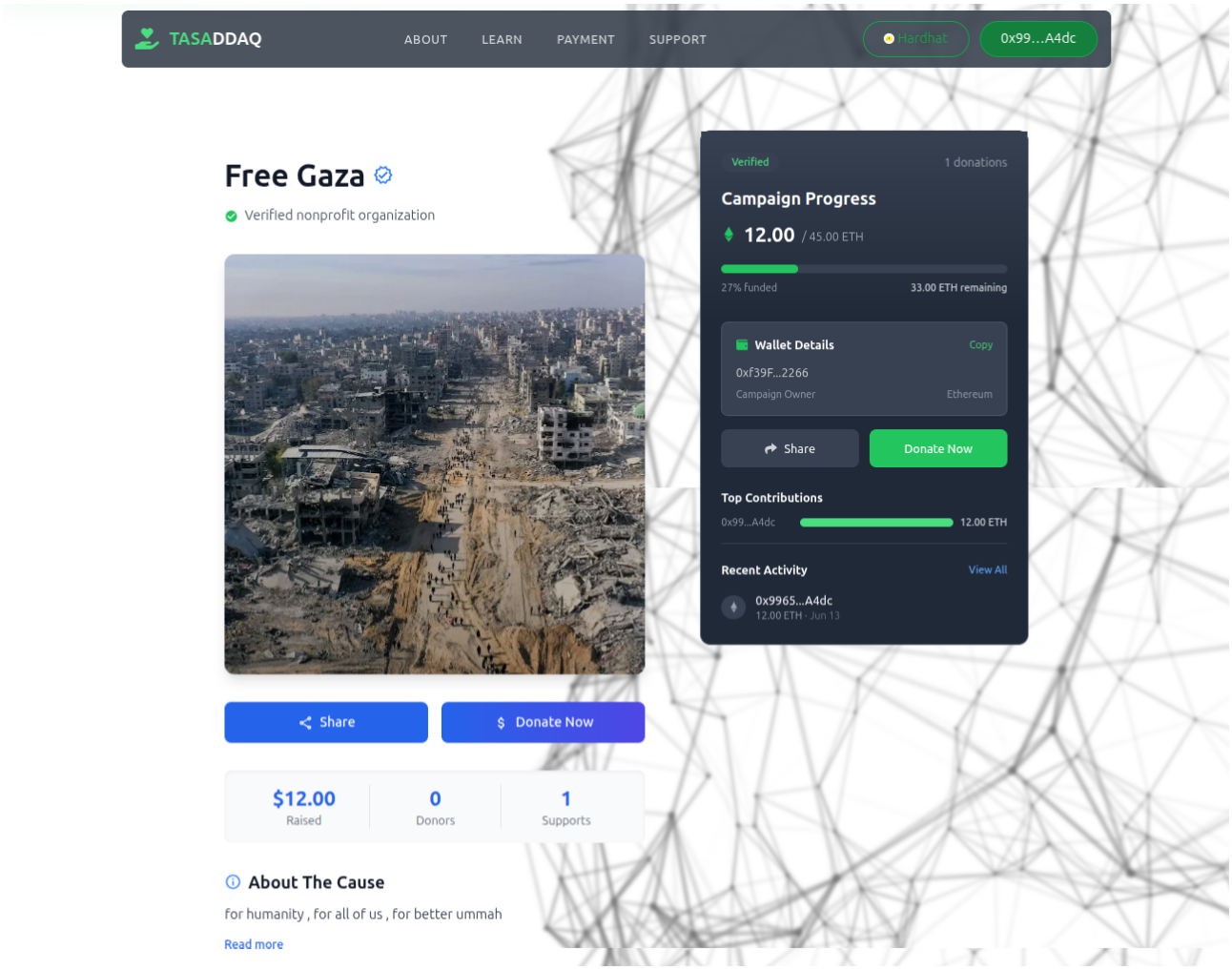

Figure 4.14: Featured Charitable Causes

When the donor selects a specific charity, for example, "Free Gaza," they are taken to the Campaign Details Page (Figure 4.14). This page provides full information about the selected charity, including the funding goal, current amount raised, number of donors, and support messages. It also confirms the charity as a Verified Nonprofit Organization with a verification badge.

To assist users, there's even a chatbot on the page that can answer questions and guide the donor through the donation process, as illustrated in Figure 4.15. This is especially helpful for first-time users or those unfamiliar with blockchain-based platforms.

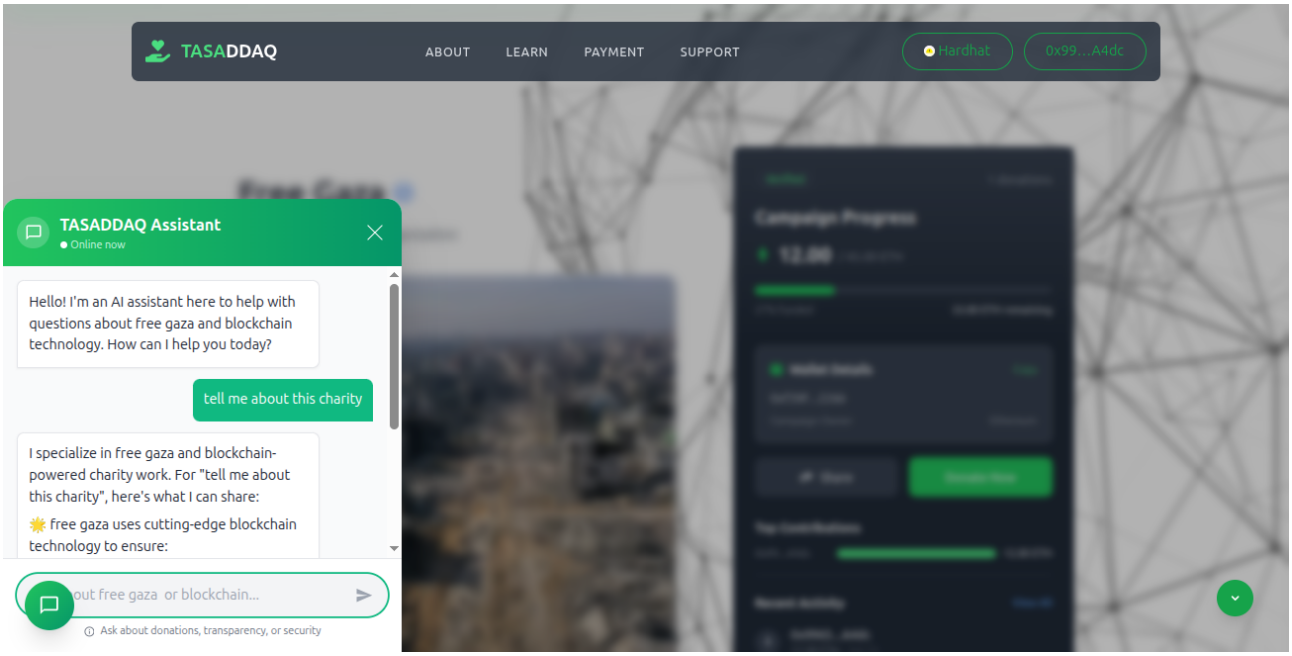

Figure 4.15: TASADDAQ assistant

When the donor clicks the "Donate Now" button from the campaign details page, a donation confirmation modal appears (Figure 4.16). In this modal, the donor is asked to fill in three fields: Name (To identify the contributor), Message of Support (A short motivational or supportive message), Donation Amount (The amount the donor wishes to contribute using Ethereum).

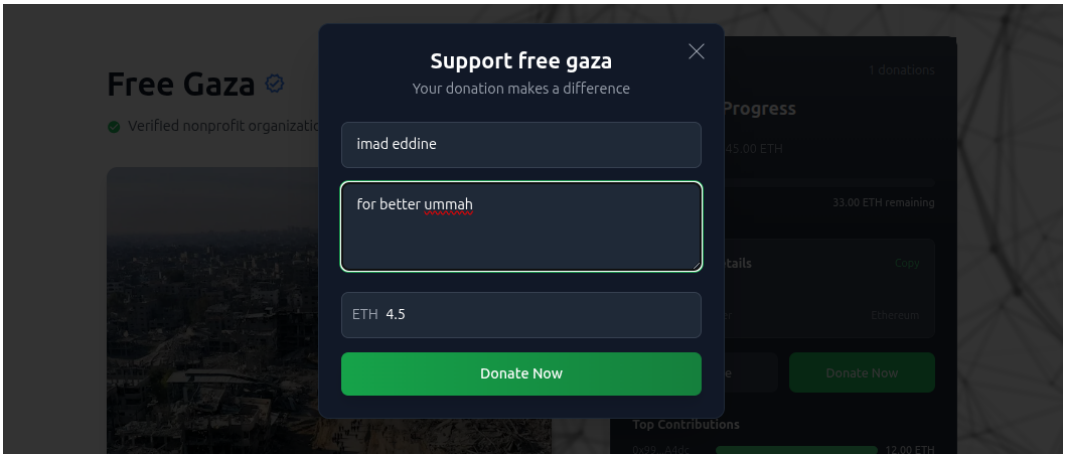

Figure 4.16: Confirming and Submitting a Donation

Once the fields are completed, the donor clicks the green "Donate Now" button. This action triggers a MetaMask (or Web3 wallet) transaction window, where the user is asked to confirm the payment including a gas fee, as shown in Figure 4.17. After confirmation, the donation is securely processed and added to the

campaign, and the contribution will appear under the campaign's Top Contributions and Recent Activity.

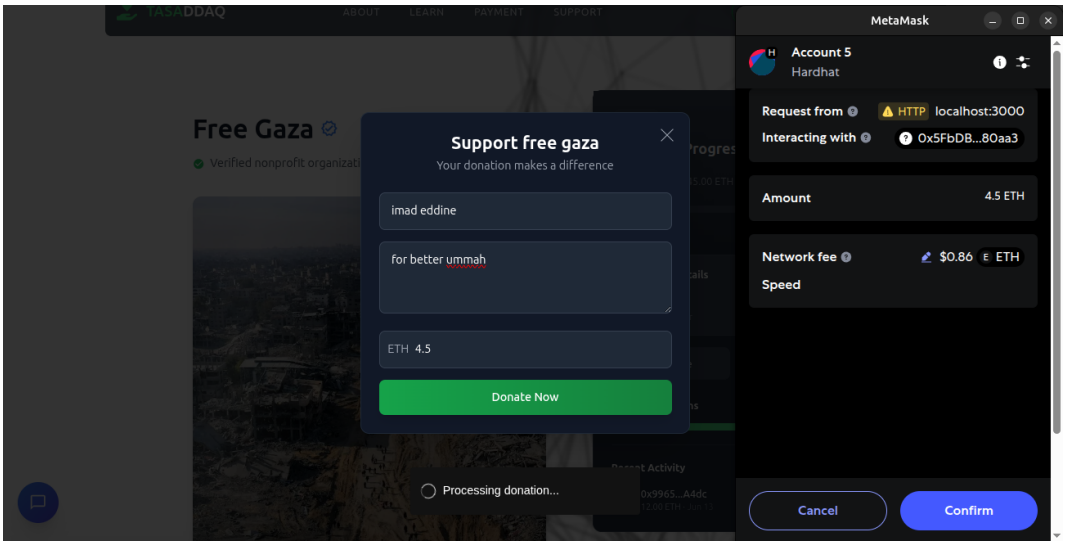

Figure 4.17: Transaction window

After the donor confirms the donation, a MetaMask transaction window is triggered for final approval. Once approved, the transaction is processed and sent to the smart contract that manages the donation logic.

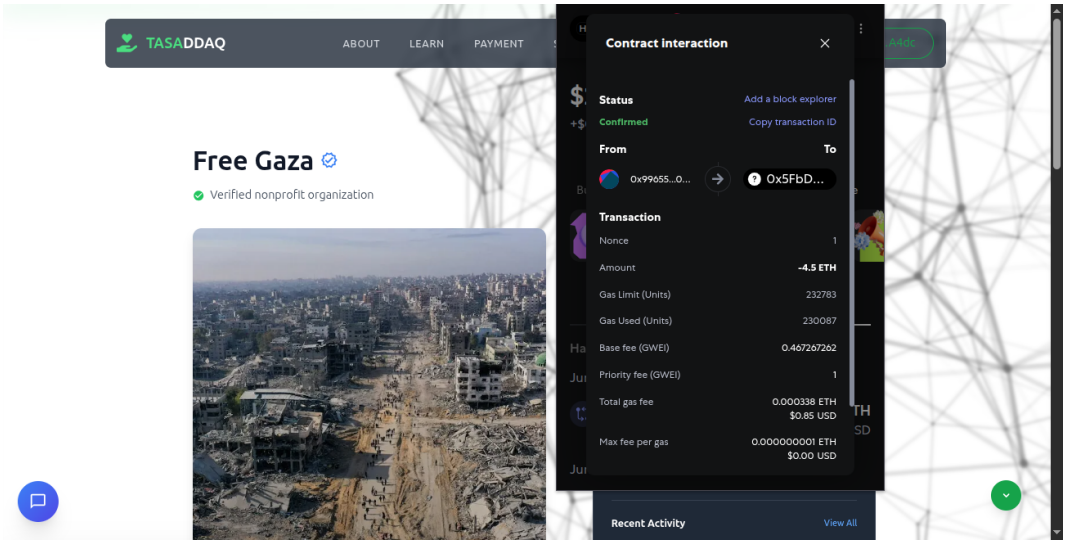

Figure 4.18: Transaction Confirmation and Wallet Verification

In this case, the transaction sent 4.5 ETH from the donor's wallet to the charity campaign's wallet (0x5FbD...). The status of the transaction is marked as "Confirmed," as shown in Figure 4.18, which

means it has been successfully

As a final step in the donation process, the platform also offers donors the ability to ban a charity campaign if they suspect suspicious activity or misuse. This action is designed to reinforce transparency and give the community a level of control over campaign quality. When a donor clicks on the "Ban Campaign" button, the platform sends a smart contract call that flags the campaign, preventing it from receiving further donations unless reviewed or revalidated by the system.

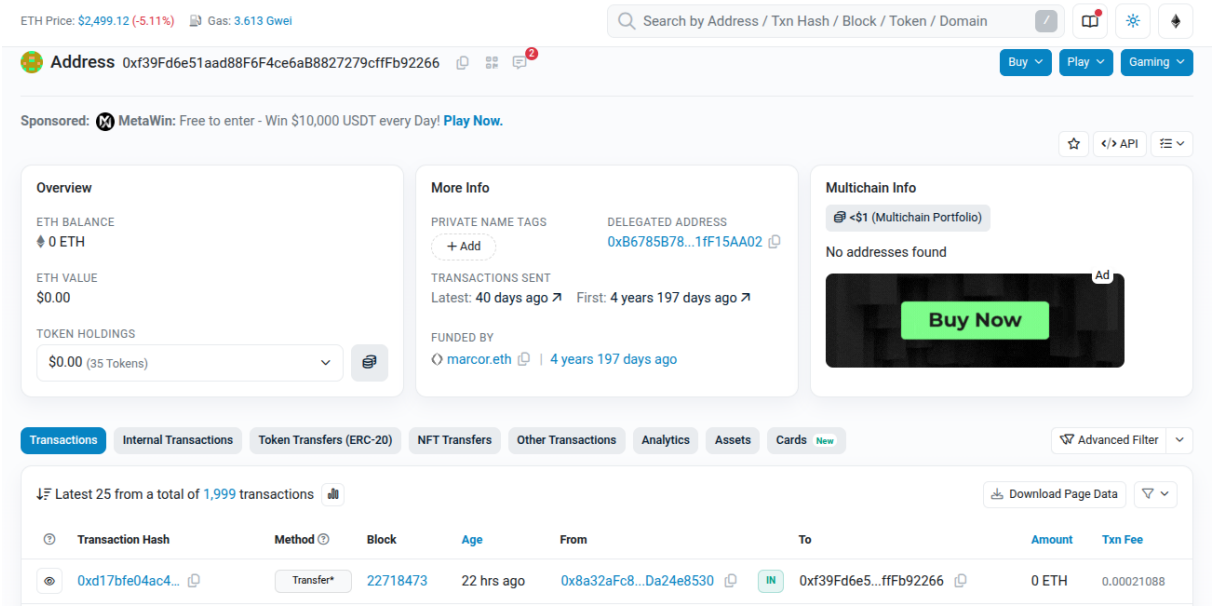

Figure 4.19: Etherscan

Every blockchain transaction related to donations including campaign creation and ETH transfers—can be viewed directly on Etherscan, as shown in Figure 4.19.

This system ensures that the donation process is transparent, simple, and secure, leveraging the power of ERC-20 smart contracts on Ethereum to empower charitable giving with trust and accountability.

4.5 Conclusion

In this chapter, we presented the complete implementation process of the TASADDAQ platform, a decentralized charity system built to ensure transparency and trust in donations using blockchain technology. We began by outlining the tools and technologies used throughout development. We then explored the key features of the project, such as charity creation, donor authentication, wallet connectivity, and campaign browsing. Through detailed implementation steps and interface walkthroughs, we demonstrated how users can interact with the system, donate securely, and track their transactions in real-time on the blockchain. The successful execution of donations along with verification via wallet history confirms the robustness and reliability of the implemented smart contract logic and user interface. This practical implementation lays the foundation for a more transparent, secure, and decentralized approach to online giving.

General conclusion

The work presented in this thesis has focused on the development of TASADDAQ, a decentralized web platform designed to improve the transparency, security, and efficiency of charitable donations by leveraging blockchain technology and ERC-20 smart contracts. This project responds to a real and persistent issue within the charity sector: the lack of trust in how funds are collected, managed, and distributed. In many traditional systems, donors have limited visibility into where their contributions go, which organizations are credible, and how efficiently funds are used.

In the early chapters, we explored the theoretical foundations of blockchain systems, smart contracts, and decentralized applications (DApps), particularly those based on the Ethereum ecosystem. We examined how blockchain technology enables immutable and transparent record-keeping, while ERC-20 tokens allow programmable, tokenized value transfer. These technologies collectively form the backbone of TASADDAQ, allowing us to design a solution that removes the need for intermediaries and enhances trust among users.

Following this, the thesis detailed the conception phase, where we modeled the platform using UML diagrams. These models helped to structure and define the system's actors, functionalities, workflows, and internal logic in a clear and maintainable way.

The implementation phase demonstrated the translation of design into a working product. Using technologies such as Solidity for smart contracts, React.js for the frontend, we successfully built and tested a decentralized platform where users can interact directly with the Ethereum blockchain. All smart contract interactions—including creating a campaign and donating—are recorded and verifiable, giving users confidence that their actions have been executed correctly and securely. From a broader perspective, this thesis illustrates the practical application of blockchain technology to a meaningful real-world problem. By enabling traceable, peer-to-peer donations, TASADDAQ empowers users and builds a more ethical and transparent charitable ecosystem. The system promotes Islamic values of trust and giving, while also utilizing the most modern tools in decentralized technology.

In conclusion, TASADDAQ is more than a technical project—it is a proof of concept for how blockchain technology can transform charity by making it transparent, accessible, and verifiable. It opens the door to further research and development in blockchain-based philanthropy, with the hope of creating a fairer, more trustworthy future for donors and beneficiaries alike.

bibliography

[**1**] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf [Accessed: Jan. 3, 2025].

[**2**] M. Swan, Blockchain: Blueprint for a new economy. Sebastopol, CA, USA: O'Reilly Media, 2015.

[**3**] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, Bitcoin and cryptocurrency technologies. Princeton, NJ, USA: Princeton University Press, 2016.

[**4**] V. Buterin, "Ethereum white paper," 2013. [Online]. Available: https://ethereum.org/en/whitepaper/ [Accessed: Jan. 15, 2025].

[**5**] A. M. Antonopoulos and G. Wood, Mastering Ethereum: Building smart contracts and DApps. Sebastopol, CA, USA: O'Reilly Media, 2018.

[**6**] G. Wood, "Ethereum yellow paper," 2014. [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf [Accessed: Feb. 2, 2025].

[**7**] F. Vogelsteller and V. Buterin, "ERC-20 token standard," Ethereum Improvement Proposal 20, Nov. 2015. [Online]. Available: https://eips.ethereum.org/EIPS/eip-20 [Accessed: Feb. 2, 2025].

[**8**] IBM, "Centralized vs. decentralized networks," IBM Blockchain, 2021. [Online]. Available: https://www.ibm.com/blockchain [Accessed: Feb. 5, 2025].

[**9**] World Economic Forum, "Blockchain for social impact: Moving beyond the hype," 2021. [Online]. Available: https://www.weforum.org/reports/blockchain-for-social-impact [Accessed: Feb. 10, 2025].

[**10**] Charity Foundation, "Transparent donation tracking via blockchain," 2020. [Online]. Available: https://www.binance.charity [Accessed: Feb. 17, 2025].

[**11**] V. Buterin, "Decentralized Applications," in Ethereum White Paper, 2013, [Online]. Available:: https://ethereum.org/en/whitepaper/ [Accessed: Feb. 17,2025].

[**12**] IBM, "Centralized vs. decentralized networks," IBM Blockchain, 2021. [Online]. Available: https://www.ibm.com/blockchain [Accessed: Feb. 20, 2025].

[**13**] Blocks, "Centralized vs. Decentralized Networks," 2020. Online]. Available: https://www.blocks.com/blockchain [Accessed: Feb. 24, 2025].

[**14**] A. M. Antonopoulos, "Decentralized Applications," in Mastering Ethereum, O'Reilly, 2018

[**15**] World Economic Forum, "Blockchain for social impact: Moving beyond the hype," 2021. [Online]. Available: https://www.weforum.org/reports/blockchain-for-social-impact [Accessed: Mar. 1, 2025].

[**16**] Binance Charity Foundation, "Transparent donation tracking via blockchain," 2020. [Online]. Available: https://www.binance.charity [Accessed: Feb. 17, 2025].

[**17**] Object Management Group, "Unified Modeling Language (UML) specification," Version 2.5.1, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML/ [Accessed: Mar. 3, 2025].

[**18**] React.js Documentation, "Component Architecture," 2023.

[**19**] React Documentation Team, "React official documentation," 2023. [Online]. Available: https://reactjs.org/docs/getting-started.html [Accessed: Mar. 4, 2025].

[**20**] Vercel, "Next.js documentation," 2023. [Online]. Available: https://nextjs.org/docs [Accessed: Mar. 5, 2025].

[**21**] Wagmi Team, "React hooks for Ethereum," 2025. [Online]. Available: https://wagmi.sh [Accessed: Mar. 5, 2025].

[**22**] R. Roberts, "Ethers.js: Ethereum JavaScript library," 2025. [Online]. Available: https://docs.ethers.org [Accessed: Mar. 6, 2025].

[**23**] Ethereum Foundation, "Web3.js – Ethereum JavaScript API," 2025. [Online]. Available: https://web3js.readthedocs.io [Accessed: Mar. 7, 2025].

[**24**] Tailwind Labs, "Tailwind CSS documentation," 2025. [Online]. Available: https://tailwindcss.com [Accessed: Mar. 8, 2025].

[**25**] Ethereum Foundation, "Solidity documentation," 2025. [Online]. Available: https://docs.soliditylang.org [Accessed: Mar. 8, 2025].

[**26**] Truffle Suite, "Truffle: Smart contract development framework for Ethereum," 2025. [Online]. Available: https://trufflesuite.com [Accessed: Mar. 9, 2025].

[**27**] Nomic Foundation, "Hardhat: Ethereum development environment for professionals," 2025. [Online]. Available: https://hardhat.org [Accessed: Mar. 10, 2025].

[**28**] Ethereum Foundation, "Remix IDE: Web-based Ethereum smart contract development environment," 2025. [Online]. Available: https://remix.ethereum.org [Accessed: Mar. 10, 2025].

[**29**] Truffle Suite, "Ganache: A personal Ethereum blockchain for development," 2025. [Online]. Available: https://trufflesuite.com/ganache [Accessed: Mar. 10, 2025].

[**30**] MetaMask, "A crypto wallet gateway to blockchain apps," 2025. [Online]. Available: https://metamask.io [Accessed: Mar. 11, 2025].

[**31**] Protocol Labs, "IPFS – InterPlanetary File System," 2025. [Online]. Available: https://ipfs.tech [Accessed: Mar. 20, 2025].

[**32**] OpenJS Foundation, "Node.js: JavaScript runtime built on Chrome's V8 engine," 2025. [Online]. Available: https://nodejs.org [Accessed: Mai. 1, 2025].

[**33**] Truffle, "Ganache – Ethereum development blockchain," 2025. [Online]. Available: https://trufflesuite.com/ganache [Accessed: Mai. 2, 2025].