CHUKA  UNIVERSITY

FACULTY OF SCIENCE ENGINEERING AND TECHNOLOGY.

DEPARTMENT OF COMPUTER SCIENCE.

TITLE: Design and Implementation of a Secure Crowdfunding System with Phased Fund Release Using Hash-Validated Voting.

VICTOR BARAKA WAFULA.

EB1/61340/22.

DEGREE: BSc. Computer Science

COSC 482
COMPUTER SYSTEMS PROJECT 1.

PROJECT COORDINATOR: Dr. Bernad Osero

ABSTRACT

This project designs a secure mobile-based crowdfunding platform for Kenyan startups, addressing fund misuse, limited transparency, and weak contributor oversight found in existing systems. It integrates a simulated escrow model, hash-validated voting, and phased fund release to ensure that funds are disbursed only after contributors collectively vote on verified milestone progress. Keccak-256 hashing is used to secure votes against tampering and maintain the anonymity of contributors while supporting full auditability, with lightweight digital signatures.

System investigation emphasized the deficiencies of a centralized crowdfunding platform and how the blockchain-based alternatives are not viable due to cost, complexity, and low user accessibility. Following the agile approach for development, the platform integrates a systematic algorithm for fund allocation and an evaluation procedure using Accuracy, Precision, Recall, and F1-Score to validate the correctness of decisions for fund release. The resulting design provides a transparent, adaptive, and secure model for governance that is suitable for the startup ecosystem of Kenya and could be applied to similar developing regions.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

LIST OF ACRONYMS

NF: Third Normalization Form
API: Application Programming Interface
B2C: Business-to-Consumer
BR: Business Registration
BRS: Business Registration Service
CRC: Cyclic Redundancy Check
CT: Campaign Type
D: Project Duration (in months)
Di: Phase Disbursement Percentage
ERD: Entity-Relationship Diagram
F: Total Funding Goal
F': Prototype Ceiling for Normalization
FK: Foreign Key
FRF: Financial Risk Factor
FRFMax: Maximum Financial Risk Factor
FTI: Fundraiser Trust Index
GUI: Graphical User Interface
H: Number of High-Value Projects
KES: Kenya Shillings
L1: Level 1 Industry Category
L2: Level 2 Industry Subcategory
MVP: Minimum Viable Product
NFRT: Non-Financial Risk Term
P: Number of Phases
PK: Primary Key
PoV: Proof of Vote (Consensus Mechanism)
R: Contributor Rating (1–5 scale)
Rm: Remedial Reserve
SHA-3: Secure Hash Algorithm 3 (Keccak-256)
STK: SIM ToolKit Push
T: Tenure on Platform (months)
UI: User Interface
UML: Unified Modeling Language
UUID: Universally Unique Identifier
VSC: Visual Studio Code
Wi: Phase Weight

CHAPTER 1: INTRODUCTION

1.0 Introduction

Crowdfunding has emerged as an alternative method for companies to obtain financial backing for ventures as opposed to the traditional methods of funding such as bank loans, hedge funding, and venture capital funding. Crowdfunding allows for businesses and organizations to aggregate small contributions from many individuals through online platforms and pool these funds to reach specific financial targets without relying on conventional credit mechanisms (Wang et al., 2018).

In Kenya, a struggle that many startups face is securing traditional funding from banks and investors, due to high collateral requirements, limited lender appetite for early-stage risk, as well as prohibitive borrowing costs which hinder growth and innovation. A locally tailored crowdfunding platform seeks to offer an alternative approach by leveraging community support and the existing digital payment infrastructure to promote innovation and mitigate these challenges.

However, most existing crowdfunding systems rely on centralized fund management models that pose risks to backers such as fund misuse, fraud, lack of transparency once the funds are disbursed and unverified project outcomes. To address these problems this project explores the development of a crowdfunding system that integrates enhanced security through a simulated escrow and verification mechanism, milestone-based validation, and phased release of funds enabling transparency and bolstering contributor confidence.

1.1 Background of the Study.

The concept of pooling resources to support a common goal is far from a new phenomenon. Historically, one of the earliest recorded cases of public fundraising tracks back to 1885, when Joseph Pulitzer successfully mobilized over 160,000 contributors through newspaper campaigns, in order to raise funds for building the base of the Statue of Liberty. This early form of collective financing is often recognized as a precursor for modern crowdfunding (Freedman & Nutting, 2015)

In Kenya, a similar culture of communal contribution exists through the practice of Harambee, which translates to “pulling together” Musau (2020), where for decades Kenyan communities have come together to raise money for education, medical bills and community projects, and this deeply rooted social tradition aligns closely with the philosophy of crowdfunding, where a collective effort and trust enable shared progress.

Access to startup financing remains a major challenge facing entrepreneurs in developing economies, including Kenya. Traditional funding institutions such as banks and venture capital firms raise the barrier of entry often by imposing strict eligibility requirements, high interest rates and collateral conditions that many early stage entrepreneurs cannot meet. As a result of the high barrier of entry, numerous innovative ideas fail to break through the conceptual phase and progress into development.

There are primarily four types of crowdfunding, Equity Crowdfunding which is the offering and sale of equity-based private security to all investors (Freedman & Nutting, 2015) , Rewards-based Crowdfunding in which a company with tangible products or services offers certain rights and benefits as a return to investors (Wang et al., 2018), Charity-based or donation crowdfunding which is primarily used for social causes raises funds without requiring contributors to receive anything in return and Debt-based or lending-based crowdfunding models in which contributors lend money to a fundraiser with the expectation that the money will be repaid over a predetermined period with a specific interest .There are mainly 3 types of participants in a crowdfunding platform: fundraisers, platform operators and investors, where the fundraisers are the initiators of a project on the platform, the platform operators are the crowdfunding platform managers that are responsible for examination and display of the projects that are created and submitted for fundraisers and the investors are individuals who select the appropriate projects for their investment through crowdfunding platforms (Wang et al., 2018).

Globally, crowdfunding has managed to anchor itself as a viable alternative to traditional funding models, with the market estimated to have grown to a $16 billion industry experiencing annual growth rates of up to 300%, primarily located in USA, Europe and China (Kiende, 2021). Crowdfunding platforms such as Kickstarter based in Brooklyn New York, Indiegogo in San Francisco, U.S.A ,and Taobao crowdfunding, which is a subsidiary of the Alibaba e-commerce platform, in China are some of the major renditions globally that have enabled millions of individuals and organizations to raise funds for projects in technology, art, education as well as for social causes. These systems establish the effectiveness of online community-based funding in supporting innovation. However, majority of these systems are localized to developed countries and primarily tailored to serve users in their localization and they mainly support payments in international currencies, which in turn limits their accessibility to entrepreneurs in developing regions. The African Crowdfunding Association notes that only 19 platforms a re currently active across the continent, supporting social, creative and entrepreneurial projects with Kiende(2021) reporting that out of 618 projects launched in Africa only 79 were fully funded, representing a 13% comparison to Europe’s 35% within the same period.

In the Kenyan context, however, crowdfunding is still an emergent field and is yet to achieve normative adoption levels. Kiende (2021) notes that campaigns often struggle with limited public awareness, low public visibility and weak digital engagement which significantly reduce the probability of projects achieving their intended goals. These observations illustrate that the effectiveness of crowdfunding efforts is directly proportional to the capacity of a campaign to reach, inform and actively mobilize its intended supporter base.

Regionally, platforms such as M-Changa in Kenya and Thundafund in South Africa have attempted to localize crowdfunding by integrating mobile systems like M-Pesa. While these platforms have been successful in supporting social causes and community initiatives, they are often not optimized for startup funding or scalable business ventures. Furthermore, crowdfunding initiatives tend to be reliant on the credibility of the fundraiser rather than on the

platform governance structures. Kiende (2021) discovered that contributors closely evaluate platform transparency fund-handling clarity and the perceived reliability of a system when deciding on whether to support a campaign.

1.2 Problem Statement.

Despite the growing popularity in crowdfunding as an alternative source of financing, existing platforms continue to face significant challenges related to trust, transparency and accountability in fund management. Once contributions are made, backers have limited visibility to how funds are allocated or whether projects are being executed as promised. The lack of oversight has hence led to numerous cases of fraudulent and failed crowdfunding campaigns, in turn undermining the confidence in the model and deterring potential investors.

Existing research highlights persistent challenges in maintaining trust, transparency and accountability in auditing within crowdfunding systems, particularly in fund allocation and verification leading to fraudulent crowdfunding projects (Xu, et al., 2023).While blockchain-based systems have been proposed as a potential solution to these concerns, their implementation remain resource intensive and complex. Blockchain integration typically requires on-chain transaction publishing, such as to the Ethereum mainnet which in turn incurs high gas and transaction fees and also demand specialized expertise. Moreover each participant must maintain a separate crypto wallet like MetaMask for fund handling verification. This introduces a significant computational and usability overhead, limiting accessibility for users with low literacy in cryptocurrency systems and in turn these challenges hinder the practicality of block-chain based systems implementation in real-time crowdfunding environments, particularly within developing regions.

Consequently, there exists a need for a more accessible, and adaptable governance model that can provide contributors with transparent and verifiable fund oversight throughout a project's lifecycle without the overhead of blockchain technology. Existing platforms lack structured mechanisms for milestone verification or phased disbursement, leaving contributors vulnerable to fund misuse once a campaign reaches its funding goal and is disbursed to the fundraiser. In order to address this gap this project therefore proposes a simulated escrow architecture, hash-based milestone verification mechanism and phased fund release, enabling for funds to be released incrementally only upon contributor approval of documented progress. This approach is designed to strengthen trust, enhance transparency as well as provide contributor assurance within the Kenyan startup fundraising landscape.

1.3 Proposed Solution

To address the challenges of fund misuse, limited transparency in unidentifiable project progress and weak contributor oversight in existing crowdfunding systems, this project proposes a phased fund-release model governed by cryptographically validated voting ensuring that no funds are released without explicit contributor consensus. The proposed solution integrates the following components:

1. Simulated escrow architecture: All contributions are held in a virtual escrow account that prevents fund access until a specific milestone is formally approved. This escrow model replicates the governance logic of existing financial custodianship without the computational overhead of on-chain blockchain systems
2. Hash-validated contributor voting: Contributors cast milestone-approval votes that are locally hashed and salted and are transmitted as tamperproof verification tokens. This ensures voter integrity, prevents manipulation and enforces a one-vote-per-contributor policy without exposing raw voting choices.
3. Phased Fund Release Engine: Project funds are disbursed in structured phases determined by algorithmic calculations (P, W_i , D_i and R_m). Funds are released during a phase only when 75 % of all eligible contributors vote yes and the milestone voting window closes successfully. This mechanism ensures transparency, accountability and measurable fund governance.
4. Contributor governance logic: The system incorporates protocols for contributor voting participation, manages inactive participants, implements suspension for recurrent abstentions without prior waiving of voting right and ensures equitable reimbursement in the event of milestone non-fulfilment.

The integration of these components forms a secure, auditable and governance-driven crowdfunding model that increases investor confidence while minimizing the risk of fraudulent or premature fund release.

Scope of the solution

The study focuses exclusively on donation-based crowdfunding, using it as the demonstrative model for implementing phased fund release and hash-validated milestone voting. Equity-based and reward-based crowdfunding are excluded due to their additional regulatory and operational requirements. However the architectural design is deliberately extensible to support other crowdfunding models in future iterations

1.3.1 Aim of the project

The general objective of this study is to design and develop a mobile-based crowdfunding platform for Kenyan startups that integrates a simulated escrow and hash-based verification mechanism to enhance transparency, security and efficiency in phased fund management.

1.3.2 Specific Objectives of the Study.

The specific objectives of this study are:

1) To define and measure the dependent variable (Y), which is the correctness of fund-release decisions made by the system during milestone verification.

2) To establish the independent variable (X) as the governance mechanism composed of the simulated escrow model, hash-validated voting and milestone approval rules that influence the correctness of fund-release decisions.
3) To design and implement the proposed crowdfunding system integrating the governance mechanism (X) to influence the outcome variable (Y).
4) To validate the system by evaluating the correctness of fund release decisions (Y) using Accuracy, Precision, Recall and F1-Score calculated from simulation results.

The following specific objectives yield the following respective research questions:

1. What measurable outcome best represents the correctness of fund release decisions during milestone validation?
2. How do the escrow model, hash-based voting mechanism and milestone approval rules influence fund-release correctness?
3. How can the system be designed and implemented to operationalize the governance mechanism for reliable phased fund release?
4. How accurately does the system make correct fund-release decisions when evaluated using Accuracy, Precision, Recall and F1-Score?

1.4 Significance of the Study.

The proposed system holds both academic and practical significance in addressing the challenges of transparency, accountability and trust in crowdfunding platforms with a specific interest in the Kenyan startup ecosystem.

The project introduces a secure, locally adaptable solution that minimizes the risk of misuse of and fraudulent activities by introducing a simulated, escrow account system as well as hash-validated voting mechanisms. By allowing for contributors to collectively approve or reject fund disbursement at different phases of a project, the system enhances trust between both investors and fundraisers, thus encouraging a wider participation in the community-driven financing.

Moreover, with integration of existing digital payment platforms such as M-Pesa which are integrated into mobile technology, the platform aliases itself as a cost effective and a more accessible alternative to block-chain based solutions, which are rather too complex and resource intensive especially for developing regions. This localized design ensures inclusivity and practicality for small-scale entrepreneurs who may lack the technical expertise required to adopt into crowdfunding platforms.

Academically the project contributes to the body of research on secure digital finance systems by demonstrating how lightweight cryptographic systems can be applied to enhance verification without the overhead of on-chain transaction verification. It also provides a foundation for future exploration into hybrid verification systems that balance scalability, security and performance in resource constrained environments.

This project offers an overall significance in fostering innovation, transparency and accountability in Kenya's evolving entrepreneurial landscape while also serving as a model for secure, community-driven funding mechanisms in similar developing economies.

1.5 Expected Outcomes/ Deliverables

The expected outcomes of this study are as follows:

1. A functional mobile-based crowdfunding system that enables contributors to make donations, fundraisers to create campaigns and all users to interact with milestone-based project progress.
2. A simulated escrow fund model that securely holds contributions and releases funds only after reaching the predefined contributor consensus threshold of 75%.
3. A hash-validated contributor mechanism where each vote is cryptographically hashed and salted to ensure integrity, tamper-resistance and verifiable milestone approval.
4. An algorithmic fund release model implementing computed values for:
    - Number of Phases (P)
    - Phase Weight (W_i)
    - Disbursement Percentage (D_i)
    - Remedial reserve (R_m)
    - Fundraiser Trust Index (FTI)

Enabling transparent, formula-driven milestone disbursement.

5. A complete backend API service supporting user authentication, campaign creation contributor handling, voting, escrow updates, fund release and refund processing.
6. A documented and normalized PostgreSQL database schema with ERD diagrams, table structures, relationships and partitioning design for contributor and fundraiser accounts.
7. Performance validation results evaluating milestone approval based on: Accuracy, Precision, Recall and F1-Score.
8. A comprehensive project report detailing system analysis, literature review, algorithmic design, implementation and evaluation in accordance with the project guidelines.

1.6 Keywords

Crowdfunding, Escrow System, Hash-Validated Voting, Phased Fund Release, Milestone Verification, Digital Finance, Transparency, Fund Security, Startup Funding.

CHAPTER 2: LITERATURE REVIEW

This chapter explores prior research, established findings, and existing models and platforms in the area of crowdfunding and encrypted transactions, with an accentuation on security, fund management and contributor-controlled mechanisms. Through analysis of the prior research, existent gaps within the crowdfunding ecosystem are identified and avenues to improve on them are also explored.

2.1 Crowdfunding Platforms and Their Limitations.

Kickstarter, one of the early pioneers of online crowdfunding and a behemoth in the space, nonetheless utilizes escrow accounts where, during the funding period, the pledged funds are held in an escrow account until the campaign either succeeds or fails. If the funding goal is met, the total amount is transferred to the project creator via payment processors like Stripe, otherwise the backers are refunded (TinyGrab, 2025). While this escrow model adds a layer of transactional safety, it still remains centrally managed by Kickstarter itself, essentially meaning contributors lack visibility or influence on the release of funds beyond a campaign's seed phase success threshold. The process, though functional, is heavily reliant on institutional trust in Kickstarter's platform rather than distributed trust among the community of backers. Furthermore, the lack of visibility post a campaign's seed-phase could result in the misappropriation of funds since the donors cede their control over the pooled funds until a campaign's fruition.

This limitation lays the groundwork for extension of the crowdfunding model. The conceptualization of the escrow structure in a decentralized, cryptographically secured framework, shifts the process of fund management from a platform-governed system to a community-governed system. In the extended system, the funds are held in a hashed escrow account rather than a centralized pool. Fund disbursement occurs through a Proof of Vote (PoV) mechanism, where contributors collectively verify milestone completion and validate fund release. This design not only preserves the simplicity of Kickstarter's original model but also aggregates transparency, control, accountability and distributed trust, ensuring that each contributor maintains oversight of the whole project during its lifecycle.

2.2 Blockchain-Based Crowdfunding and Smart Contracts

Several blockchain-based crowdfunding platforms have been proposed to enhance transparency and data integrity. (Xu, et al., 2023)Proposed a private Ethereum based crowdfunding platform built on the Go-Ethereum client *geth*. In their proposal an auditor pool is used to verify project authenticity, where the smart contract randomly selects the appropriate auditors to form an auditor committee to ensure fairness and reduce central control. Although the decentralization is effective in promoting accountability, the reliance on auditors introduces partial centralization and may slow down fund release.

Similarly (Hassija, Chamola, & Zeadally, 2020)proposed a unique and secure crowdfunding platform, *BitFund* which was also based on Ethereum smart contracts where investors request a

specific project and different developers bid with varying values for the project ownership. Their proposed platform eliminates the need for any manual negotiation between investors and developers for projects through the integration of a bidding system. However *BitFund's* design still relies on developer honesty and lacks a structured mechanism for contributor approval prior to fund disbursement, hence limiting its ability to prevent misuse of raised funds.

2.3 Voting-Based Consensus Mechanisms

The Proof of Vote (PoV) consensus mechanism introduced by (Li, Li, Hou, Li, & Chen, 2017) presents an innovative approach to achieving a distributed consensus in consortium blockchains through democratic voting rather than computational power or token staking. In the PoV system nodes within a consortium are assigned specific roles such as commissioners, butlers and candidates each responsible for voting, block generation and verification. The separation of voting rights and execution rights enhances fairness, transparency and independence within the network. A vote is deemed valid when it receives a majority vote of ≥51% from the commissioners, ensuring that transaction confirmation occurs without reliance on a central authority. Unlike Proof of Work (PoW) or Proof of Stake (PoS) consensus protocols which consume a lot of computational resources since PoW specifically uses a difficulty variable to maintain a stable block time as miners come and go, which impacts the mining power of the network (hashrate) (Solorio, Kanna, & Hoover, 2019). PoV offers a low latency and less resource intensive approach that prevents blockchain bifurcation while offering a high throughput and security. The models democratic verification principle offers an adaptable framework for community-driven ecosystems such as crowdfunding platforms where collective voting could govern fund release and project validation.

2.4 Fraud Detection, Security and Anomaly Detection Models

In their paper (Zkik, Sebbar, Fadi, Kamble, & Belhadi, 2024) explore the use of Graph Neural Network (GNN) models to protect blockchain-based crowdfunding platforms from cybersecurity attacks, through anomaly detection. Their study categorizes cybersecurity threats into two major groups: cybersecurity attacks targeted at the Smart Contracts and cybersecurity threats targeted at the users, such as fraudulent campaign creation. To establish context (Zkik, Sebbar, Fadi, Kamble, & Belhadi, 2024), reference earlier work by Shafqat et al.(2021) who developed a machine learning system capable of detecting fraudulent crowdfunding campaigns with an 85% accuracy. The study aims to help build trust in the blockchain-based crowdfunding ecosystem hence allowing users to invest their funds safely. While the proposed system effectively detects irregularities, it primarily focuses on identifying fraud rather than establishing governance structures for contributor-controlled fund management.

2.5 Identified Research Gap

In summary, the literature review highlights significant advancements in the integration of blockchain technology, smart contracts, and machine learning to improve transparency, trust and security in crowdfunding ecosystems. However, there is a persistence of centralization within the existing systems -both traditional and blockchain-based- and consequentially limited contributor

participation in fund management with platforms such as Kickstarter depending mainly on institutional trust, while blockchain proposals like BitFund depending on auditor-based systems and not a unanimous consensus. Additionally, existing studies venture more into fraud detection and transaction integrity rather than community-driven governance. This resultant gap establishes the basis for the proposed system, which extends the traditional crowdfunding framework through a decentralized, cryptographically secured escrow model governed by a voting based fund release mechanism. The proposed model aims to achieve a balance between transparency, accountability and contributor autonomy in fund disbursement.

CHAPTER 3: METHODOLOGY

This chapter outlines the methodology adopted in the design and implementation of the proposed crowdfunding platform. It describes the design, development approach, system architecture and evaluation methods used to achieve the study's objectives. The methodology focuses on the integration of escrow-based fund management, hash encryption and voting based fund release mechanism to enhance transparency and contributor control within the crowdfunding environment.

3.1 Research Design

The study uses an applied research design aimed at demonstrating a secure model for crowdfunding that integrates cryptographic hashing and contributor voting for phased-fund disbursement. The prototype simulates an escrow based fund management and democratic decision making within the crowdfunding environment, emulating a blockchain environment's transparency without the overhead of an actual distributed ledger.

3.2 Data Requirements

This study utilizes company registration data as one of the core components of system modeling. However, access to official company records in Kenya is managed by the Business Registration Service (BRS) which is legally mandated to maintain registers, data and records on registrations (Business Registration Service Act, cap. 499B, 2015). Access to these records is restricted to authorized entities. Consequently, the dataset utilized in this study is completely synthetic and is generated to simulate authentic company identifiers and attributes, hence ensuring compliance with data access regulations while maintaining a structural integrity necessary for the simulation and analysis.

3.3 Development Approach

The system follows an agile development model, allowing for continuous refinement of the platform through the designing, testing and improvement of individual components. This approach is well suited for the project since it integrates multiple dependent modules, such as the escrow management module, the hashing module, and the voting-based fund release module, each requiring independent verification before integration. Iterative refinement identifies potential issues in functionality and logic and resolves them before any subsequent iteration, resultantly ensuring overall system stability and accuracy. This model allows for iterative feedback loops and prototype validation at each stage.

3.4 High-Level System Architecture

The system architecture emulates a transparent crowdfunding environment in a structured domain of four main layers: the user interaction layer, the transaction processing layer, the fund management layer and the data persistence layer. The user interaction layer outlines interfaces for contributors and project creators to register, pledge and monitor progress. The transaction layer handles the initiation, validation and encryption of transactions through secure hash functions. The fund management layer simulates escrow operations where contributor funds are

held in a locked state until contributor approval is achieved. In the data persistence layer, user records and transactions are securely stored and retrieved for verification.

3.5 System Modelling

For system modeling and design, Unified Modeling Language diagrams such as use case, data flow and activity diagrams are employed to visualize the relationships between system components, defining user roles, transaction flows and logic governing fund release based on contributor votes. The Proof of Vote (PoV) mechanism is conceptually integrated into the system's workflow, allowing for contributors to participate in decision-making regarding fund disbursement whereby each voting instance is correspondent to a projects milestone, and upon majority approval, the escrowed funds are released to the project creator.

3.6 Evaluation method

To validate the correctness of fund-release decisions (dependent variable Y), a controlled simulation will be conducted. The simulation will generate multiple milestone verification scenarios and the system's decisions against ground-truth outcomes.

3.6.1 Simulation Scenarios

The following scenarios will be generated:

- Normal voting: Most contributors vote correctly; some abstain.
- Missing Voters: A subset of contributors fail to vote.
- Mixed Voting: Voting patterns such as 60/40 or70/30 will test threshold sensitivity.
- Incorrect votes: A subset of votes will intentionally be incorrect.

3.6.2 Simulation Procedure

A ground-truth value will be assigned to each milestone indicating whether it is actually completed and contributor if approval ≥ 75%. Contributor votes will be generated depending on the scenario whether correct, incorrect or missing. The system will compute the approval percentage:

$$approval = \frac{Yes}{Yes + No}$$

The milestone will be approved if approval ≥ 75%. The system's decision will be compared to the ground-truth to determine TP, FP, FN and TN.

3.6.3 Metric Computation

The evaluation metrics will be computed from the following formulas:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy measures the overall correctness of the systems fund release decisions.

$$Precision = \frac{TP}{TP + FP}$$

Precision measures how often the released funds were released correctly.

$$Recall = \frac{TP}{TP + FN}$$

Recall measures how often the system releases funds when it should.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

F1-Score provides a balanced measure when both false releases (FP) and missed releases (FN) must be minimized.

Where:

- TP (True Positive): Correct fund-release when a milestone is actually valid.
- FP (False Positive): Incorrect fund release when a milestone is invalid.
- FN (False Negative): System withholds release when milestone is valid.
- TN (True Negative): Correctly withholds release when milestone is invalid.

3.7 Ethical and Regulatory Considerations

The study will utilize synthetic datasets to avoid handling real company records. No sensitive financial records will be stored and all transactions will be simulated. Hashing mechanisms will ensure data integrity and prevent tapering. The system will operate with regard to relevant data protection and regulatory guidelines.

3.8 Chapter Summary

This chapter presented the methodology adopted in the development of the proposed crowdfunding platform. It identified the applied research design, described the synthetic data sources, and elaborated on the iterative development approach used throughout the project. Furthermore, this chapter introduced the high-level system architecture and presented the conceptual modeling techniques that were followed to guide system structure. The simulation-based evaluation procedure was defined, including scenarios, metrics, and validation logic applicable in assessing the correctness of milestone-based fund release decisions. Also highlighted were ethical and regulatory considerations driving the use of synthetic datasets and secure practices. The methodology thus provides the foundation upon which the system investigation and detailed design in the next chapters are based.

CHAPTER 4: SYSTEM INVESTIGATION, ANALYSIS AND REQUIREMENTS DEFINITION

4.1 Introduction

The system analysis and investigation stage is a crucial stage in the system development lifecycle (SDLC) , as it provides a factual basis for understanding the current existing environment, identifying the gaps within the system and establishing accurate requirements for the proposed solution. This phase is essential in ensuring the proposed solution is grounded in factual evidence rather than assumptions, thereby improving the reliability and accuracy of the final implementation.

For the Secure Crowdfunding Platform with Phased Fund Release Using Hash-Validated Voting, this phase focuses on examining existing crowdfunding platforms and frameworks, identifying both their operational and security constraints. The study revealed persistent issues pertaining fund transparency, centralized fund control and limitations in contributor oversight post fund disbursement. These findings guided the definition of new requirements centered on security, accountability and contributor participation.

The investigation was conducted through document review and secondary data analysis, focusing on published books, academic articles, published research papers, technical documentation of existing crowdfunding systems and credible online sources discussing digital finance systems and secure fund management. The derived insights from these materials provided a comprehensive understanding into the operation of existent crowdfunding systems, the limitations and the avenues for improvement in both transparency and control.

The adopted approach for this stage is an Agile development model, emphasizing iterative development, modular testing and progressive refinement of the system components. This approach was chosen due to the project's integration of multiple independent modules, such as escrow management, hash-based verification, and contributor voting, whereby each requires independent analysis and validation. The suggested agile approach allows for continuous reassessment of system functionality throughout the investigation, therefore ensuring that all requirements involved align with technical findings and project objectives.

4.2 The Current Existing System/Environment

The system investigation analyzed both the local financial landscape for Kenyan Startups and the limitations of contemporary digital funding platforms

4.2.1 The Kenyan Startup Funding Environment

The current environment is characterized by significant barriers to capital access which consequently make traditional and even formal funding methods largely inaccessible to new ventures. Secondary data analysis, specifically from the study on the influence of financial resources on start-up success in Kenya (Ronald, 2017) confirms the critical financial gap:

- Access to finance is the single most pressing hindrance facing new start-up ventures in Kenya.
- An overwhelming majority of survey respondents, 95.7%, acknowledged that access to capital is their greatest obstacle (Ronald, 2017).
- As a direct consequence of this environment, the most preferred sources of capital are personal and informal. The two main sources of capital for start-ups are self-funding and borrowings from family and friends (Ronald, 2017).
- A majority of respondents , 95.7% concurred to have used self-funding as the major source in financing their business (Ronald, 2017)
- Other major operational challenges faced by start-ups are tax burden and competition (Ronald, 2017).
- With Regards to government support, a majority of entrepreneurs, 56.5% disagreed that the government has put in place sufficient programs to support new firms (Ronald, 2017).
- The prevailing environment of inaccessible formal financing and high reliance on personal funds, compounded by the lack of uptake of existing government initiatives such as the Youth Enterprise Development Fund (YEDF) and the Uwezo fund is curbed by a lack of information and awareness. The prevalent environment of inaccessible formal financing and high reliance on personal funds creates a necessity for a specialized, localized and transparent alternative funding mechanism.

4.2.2 Constraints of Existing Crowdfunding Systems.

While global and local platforms offer a framework for capital aggregation, their operational structures introduce critical security and governance inadequacies:

- The existing systems are reliant on centralized fund control (TinyGrab, 2025), where the platform acts as a single and trusted intermediary, maintaining unilateral custody over the funds post a campaign's completion.
- The reliance on intermediaries introduces an Agency Problem, resulting in issues pertinent to fund transparency and limitations in contributor oversight post fund-disbursement. Backers currently lack a formal, auditable mechanism to validate project progression before subsequent funds are released, an insufficiency that necessitates the development of new decentralized trust models (Xu et al., 2023).
- More complex decentralization solutions, while offering cryptographic transparency, often suffer from poor Non-Functional Requirements (NFRs) like high computational overhead, high transaction fees, and a steep learning curve amplified by the need to manage complex crypto wallets, increases the probability of non-adoption by the average user in the market.

Existing platforms only verify the completion of campaigns at the end of the funding period. They do not enforce milestone-level progress checks, nor do they give contributors any

structured governance rights over fund release during project execution. This gap exposes contributors to the misuse of funds and early exhaustion of funds.

4.3 The Intended System

The Secure Crowdfunding Platform with Phased Fund Release, is designed to address the two main gaps identified in 4.2: the inaccessibility of formal finance and the lack of contributor governance that ensures contributors have transparent oversight and fundraisers remain accountable throughout the project lifecycle in digital alternatives. The specifications for this system are categorized into Functional and Non-Functional Requirements.

4.3.1 Functional Requirements

The functional requirements define the key operations that the Secure Crowdfunding Platform with Phased Fund Release Using Hash-Validated Voting must perform in order to fulfill its objectives. These requirements facilitate secure, transparent and verifiable crowdfunding operations:

- User registration and authentication: The system shall allow users to register either as contributors or fundraisers through a secure signup interface. Each user must provide valid credentials which are verified before granting access. Authentication is managed through encrypted credentials to ensure secure user identification and prevent unauthorized access
- Campaign creation and management: The system shall enable fundraisers to create crowdfunding campaigns by submitting project details such as title, funding goal, funding deadline, milestones, and duration of the campaign. The platform must also allow fundraisers to update campaign progress at each milestone and upload supporting media in order to enhance credibility.
- Contribution management: The system shall allow contributors to view active campaigns and pledge funds using the M-Pesa payment gateway and all payments are recorded in the system and reflected in real-time within the campaign dashboard. A confirmation message shall be sent to the contributor upon successful transaction.
- Escrow fund holding: All received funds shall be held in an escrow account until milestone completion. The system shall restrict fund withdrawal by the fundraiser, rather requiring contributor consensus before disbursement for the phased release. This feature ensures that funds remain secure and traceable and also enforces contributor oversight over the project's progress.
- Hash-validated voting: The system shall allow contributors to participate in milestone verification through a voting process. Each contributor's vote shall be hashed using the Keccak-256 algorithm to avoid tampering with votes and ensure integrity in the voting process. A milestone is approved once it achieves a predefined voter consensus of at least 75%.
- Phased fund release: Upon achievement of the voting consensus, the escrow module shall automatically release funds to the fundraiser using the simulated M-Pesa B2C API. In

cases where the milestone fails to achieve the minimum contributor consensus threshold, the systems shall initiate a refund simulation, of the funds held in a simulated escrow account to contributors, ensuring accountability and trust.

- Transaction logging and audit trail: The system shall maintain a detailed log of all transactions votes and fund releases, whereby each log must include timestamps, hashed identifiers, and transaction statuses to facilitate auditability and traceability for all stakeholders.
- Progress monitoring and notifications: The system shall provide both contributors and fundraisers with real-time updates on campaign progress, voting results and transaction statuses through in app notifications. This feature enhances transparency and engagement throughout the campaign lifecycle.

4.3.2 Non-Functional Requirements

The non-functional requirements describe the system performance, security, usability and reliability standards that the Secure Crowdfunding Platform with Phased Fund Release Using Hash-Validated Voting must meet in order to ensure stability and user trust. These requirements describe how the system operates rather than its purpose, ensuring that functional features perform efficiently and consistently across varying conditions:

- Performance and responsiveness: The system shall maintain low-latency transactions and handling during voting operations, achieving an average response time of at least two seconds. Caching through Redis minimizes query delays during high frequency events.
- Scalability: The platform shall support interactions from multiple users without degradation of performance through containerization using Docker and asynchronous processing in FastAPI allowing for seamless scaling with an increase in the number of active campaigns.
- Security and data integrity: All user credentials and votes shall be encrypted. The Keccak-256 hashing algorithm ensures collision-resistant vote transaction and validation. No centralized entity shall have write privileges over contributor data, preserving decentralization and implementing tamper resistance.
- Reliability and fault tolerance: The system shall recover gracefully through state tracking mechanism which logs each transaction and milestone transition with a cryptographic hash reference, allowing for the backend to restore the most recent valid state in the event of a failure, eliminating the need of a full database redundancy. Each voting session operates within a predefined time window of 24-48 hours, during which contributors cast their votes and the system automatically tallies the cast votes. Once the window closes, uncast votes are automatically suspended and excluded from quorum calculation. Suspended voters are re-prompted to either confirm continued participation or voluntary withdrawal before subsequent milestones. This approach maintains democratic fairness and introduces an asynchronous aspect to the voting consensus mechanism, therefore preventing stalling from inactive participants.

- Consensus integrity: Each contributor shall be assigned a unique hash token upon contribution, which is mirrored within the simulated escrow ledger. During milestone validation, the hash is submitted alongside the vote and the escrow verifies the submitted hash against the stored record to authenticate the vote and prevent vote duplication. Votes submitted outside the window are automatically suspended and excluded from aggregation. If a contributor votes *No* their escrowed contribution balance is flagged for refund simulation through the M-Pesa B2C API. This mechanism introduces cryptographic accountability into the consensus process, ensuring tamper-resistant validation independent of an external blockchain.
- Usability and accessibility: The Android interface shall feature an intuitive dashboard that allows easy navigation between campaign viewing, contribution and voting. The design adheres to Flutter's material3 guidelines to support accessibility for users with minimal technical literacy and adapts seamlessly to varying screen sizes.
- Transparency and auditability: All fund flows votes and milestone transitions shall be visible through a public campaign ledger available to all campaign participants. Each recorded action shall include a timestamp and a cryptographic signature, allowing for end to end traceability without exposing contributor identities. For milestone verification, fundraisers shall be required to upload or link supporting materials such as photos, reports, or prototype demonstrations. Each uploaded file shall be hashed using the Keccak-256 algorithm at the time of submission, and the hash stored alongside the milestone record to prevent post-submission modification. This ensures any alteration post submission changes the original hash, maintaining authenticity and accountability of the submitted proof of progress.
- Authenticity validation: To prevent the use of fabricated, stock or AI-generated media as milestone evidence, the platform integrates a minimal multi-layer authenticity validation mechanism, whereby each uploaded material undergoes automatic EXIF metadata extraction to confirm its device origin, creation timestamp and where available its geo-location. Files lacking verifiable metadata or edited records are automatically flagged for contributor review. Additionally, contributors can perform peer authenticity voting to approve or reject milestone materials before the fund release voting phase. This lightweight approach preserves integrity while minimizing system overhead and implementation complexity.
- Maintainability and extensibility: The system architecture shall utilize modular service components for the escrow engine, hash-validation service and the voting module. Each module can be independently updated or extended to incorporate additional payment gateways, verification mechanisms or authentication features without requiring a complete structural overhaul.
- Availability: The system shall target a minimal uptime of 99% in testing, supported by lightweight background services that automatically restart failed modules to maintain operational continuity.

- Privacy and compliance: User data shall be stored according to secure-data-handling practices limiting access to pseudonymized identifiers. Sensitive financial information, such as actual M-Pesa credentials, shall not be retained within the system.

4.4 Chapter Summary

This chapter examined the existent crowdfunding ecosystem, identifying the barriers that Kenyan startups face in accessing equitable financing and the governance limitations coupled within the traditional platforms. Through system investigation and analysis, functional and non-functional requirements for the Secure Crowdfunding Platform with Phased Fund Release Using Hash Validated Voting were established. The defined specifications emphasize decentralization, transparency and contributor empowerment through phased fund disbursement, cryptographic validation and authenticity validation of milestone evidence. These requirements collectively form the base upon which the system structure is designed, guiding the architecture and implementation decisions.

CHAPTER 5: SYSTEM DESIGN

5.1 Introduction

The system design phase is crucial to the Software Development Life Cycle (SDLC), because it refines the functional requirements defined in Chapter 6 into a concrete and structural blueprint for the platform. It translates requirements by converting the abstract goals of ensured transparency and contributor control into a logical and physical system architecture. It is the section where the secure model for fund release, incorporating a simulated escrow and hash validated contributor voting is formally structured to prevent fund misuse and fraudulent activities. The relationship between system components, such as user interaction, transaction processing fund management and data persistent layers are also defined in the system design, describing the user roles and transaction flows. The resulting design, visualized using Unified Modelling Language (UML) diagrams such as use case and Data Flow Diagrams, outlines a roadmap for the development process, ensuring the project development aligns with the objectives and performance standards.

This design phase is essential in ensuring that all architectural components, data structures and governance flows are logically validated before implementation, supporting reliability and subsequent evaluation procedures outlined in Chapter 3.

5.1.1 Technology Stack Justification

The development of this crowdfunding platform supports modularity, scalability and secure financial operations. The system integrates technologies for backend processing, frontend interaction, data management, cryptographic operations and payment simulation. The development environment primarily comprises of Visual Studio Code (VSC) as the Integrated Development Environment (IDE), due to its flexibility, integrated debugging, and plugin ecosystem supporting frontend, backend and containerization. Development and testing are conducted locally, with the system containerized for modular deployment and dependency consistency.

The backend is implemented in Python using the FastAPI framework. Python was selected for its simplicity, rapid development capability and extensive libraries supporting cryptographic operations and API integrations. FastAPI was selected over enterprise frameworks such as Spring Boot because of its lightweight nature, flexibility and its native support for asynchronous operations which enable simultaneous handling of multiple voting and transaction events without bottlenecks, this async-first approach ensures low latency performance during voting, tallying and real time campaign updates. The backend handles the core processes such as logical escrow management, voting aggregation and hash generation for transactional integrity.

The frontend development utilizes Dart's Flutter framework, which enables for android development primarily and also supports cross platform development from a single codebase. Flutter's API integration capabilities facilitate seamless integration between contributors, project

creators and the backend services allowing for real-time updates for pledges, votes, and funds and project statuses.

For database management the system utilizes PostgreSQL as the primary relational database due to its scalability, ACID compliance, and support for advanced data types alongside cryptographic extensions. PostgreSQL also supports table partitioning essential for separating the contributor and fundraiser accounts and for handling large volumes of voting and transactional logs with minimal performance degradation.

Redis serves as an in-memory caching layer which in turn enhances query response times during high frequency operations such as voting and transaction validation. Its low latency characteristics prevent performance bottlenecks during simultaneous contributor participation and help maintain the responsiveness expected in real-time voting systems.

5.1.2 Payment Simulation Approach

The M-Pesa Daraja API is integrated as the main payment gateway to simulate financial transactions, where all financial flows are handled through sandbox APIs and callback events to ensure that no real monetary transactions or sensitive financial records are handled by the system. The STKPush API facilitates user pledges and the B2C API handles fund release and refunds, where refunds are simulated via B2C API callbacks triggered upon failed milestone progression votes. While the escrow simulation logic is handled from the backend layer, managing fund states whether it is held, released, or refunded, without reliance on a physical escrow account. The hybrid integration allows for realistic transaction simulation and contributor-controlled fund management.

5.1.3 Cryptographic Method

For cryptography and security, transaction and voter integrity are preserved using Ethereum's Keccak-256 hashing algorithm. Keccak-256 was selected over CRC encryption for its collision resistance in contrast to CRC which only performs error checking, which is essential for security. Secondly it employs modern Sponge Construction, which is inherently more secure and immune to length extension attacks which the SHA-2 algorithms are prone to. The Keccak-256 algorithm is robust highly vetted and represents the core innovation behind the global SHA-3 standard ensuring the transaction uses the same cryptographic foundation that secures modern transactions. The algorithm is robust and extensively analyzed to form the basis of the NIST-standardized SHA-3 family of cryptographic hash functions.

5.2 Process Flow Design

The process design illustrates the integration between different components of the system to fulfill the functional requirements, showing the sequence of actions, decision points and data transition points during core operations such as contribution, voting and phased fund release.

These workflow diagrams outline a clear implementation blueprint aligning the system logic, user interactions and backend mechanisms with the system design.

5.2.1 Use Case Diagram

The use case diagram outlines the primary interactions between the external actors and the core functions of the crowdfunding platform. It identifies the roles of contributors and fundraisers and maps the operations that each actor is a participant in. The diagram provides a high-level view of the systems functional boundaries by illustrating which activities are managed at the User Interface layer, which activities trigger backend actions such as escrow updates or hash validated voting and the coordination between different modules in achieving the milestone verification and phased fund release.

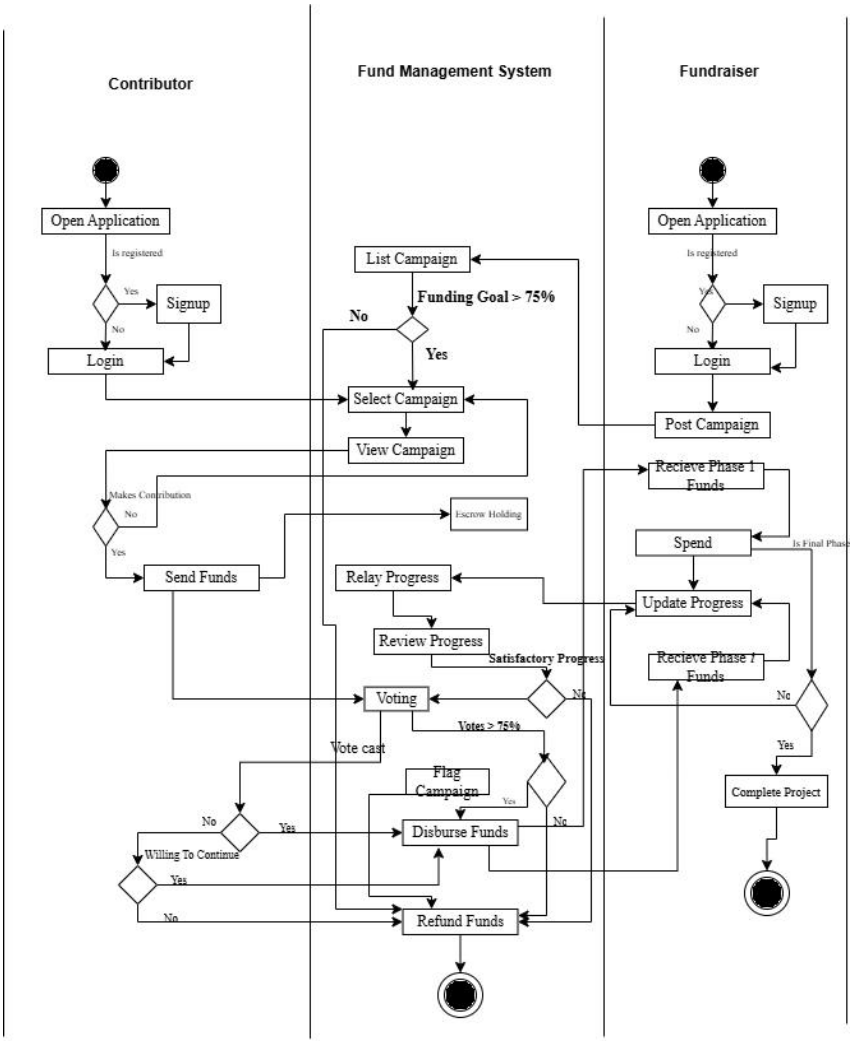
Figure 5- 1: Use Case Diagram for the Secure Crowdfunding System Showing Actor Interactions and Core System Functions

Figure 5.1 illustrates the key actors: Contributor, Fundraiser, Escrow and Voting Engine and their interactions within the crowdfunding system. Fundraisers create and manage campaigns, submit milestone proof and receive phased funds. System driven actions such as fund transfer, escrow updates and disbursement are modelled as included and extended use cases since they occur automatically. The diagram provides a clear overview of how user roles and backend processes are coordinated to achieve milestone-based, governed fund release.

5.2.2 Activity Diagram: Phased Fund Release and Verification

The activity diagram shows the end-to-end workflow of how contributors, fundraisers, and the Fund Management System coordinate in creating a campaign, contributing toward it, verifying the milestones and phased fund release.

Figure 5- 2Activity diagram showing the end-to-end workflow for contribution, verification, and phased fund release.



The diagram illustrates how contributors navigate campaigns, contribute to them, and vote on milestones, while fundraisers create campaigns, upload updates, and receive funds in phases. The Fund Management System handles escrow holding, review of progress, vote counting, funding payout, and refunds upon failure. This constitutes full, governed, and verifiable execution of the fundraising lifecycle.

5.2.3 Algorithmic Fund Allocation

1. **Fundraiser Trust Index (FTI)**

. To incorporate fundraiser reliability into the disbursement logic the system computes a Fundraiser Trust Index (FTI) which accounts for historical performance and behavioral integrity and is bounded between 0 and 1.Its defined as

$$FTI = \min\left(1, 0.4\left(\frac{T}{60}\right) + 0.3\left(\frac{S}{20}\right) + 0.2\left(\frac{H}{10}\right) + 0.1\left(\frac{R}{5}\right)\right)$$

Where:

T = Tenure on the platform in months.

S = number of successful campaigns.

H = Number of high value projects (≥ KES 100,000).

R = Cumulative rating on a 5-star scale.

f(x) = Hybrid normalization function.

The coefficients sum to 1 as each weight reflects the reliability of its corresponding metric with T being assigned the highest weight (40%) as the strongest indicator of operational consistency and S being assigned 30% since successful campaigns are a direct indicator of execution reliability and H being assigned 20% as it demonstrates execution ability under high value environments and R has the lowest coefficient of 10% due to its volatility and subjectivity. For normal operational ranges simple linear scaling is applied but in the event of outliers the model switches to a logarithmic fallback which introduces diminishing returns and normalizes the trust index.

$$f(x) = \begin{cases} \dfrac{x}{normal\_\max(x)}, \\ \dfrac{\ln(1+x)}{\ln(1+cap(x))} \end{cases}$$

With normal_max (T/S/H) being 60, 20 or 10 respectively and cap (T/S/H) being 240, 1000 or 400 respectively denoting 20 years and 1000 successful projects and 400 High cash projects.

2. **Company Risk Factor (C)**
   Unclamped C

$$C_{raw} = \left(l1_{rw} \times l1_{weight}\right) + \{\left(l2_{rw} \times l2_{weight}\right)\}$$

   Constrained risk Factor (C)

$$C = clamp(C_{raw}, 0.30, 0.90)$$

To ensure stability across all subsequent calculations of Phase Count (P) and Remedial Reserve (R_m) the computed company risk factor is constrained within an operational range of $0.30 \leq C \leq$

0.90 using a clamping function preventing low or high risk values from creating disproportionate outputs and aligns with financial risk-modelling practices.

3. Campaign Type (CT) adjustments

It is based on campaign type where: CT∈ {donation, product, equity}, the system loads preset adjustment constants: CT phase_{adj} (-1/ 0/ +1) which affects the number of phases, CT risk_{adj} (-0.03/ 0/ +0.03) which adjusts reserve sensitivity and both these metrics define how campaign type influences risk control. With donation being the lowest since it has a no money back guarantee and equity having the highest constant due to its securities limitations and money back guarantee. Since the project is limited to donation-based crowdfunding, the constants -1 and -0.03 are applied.

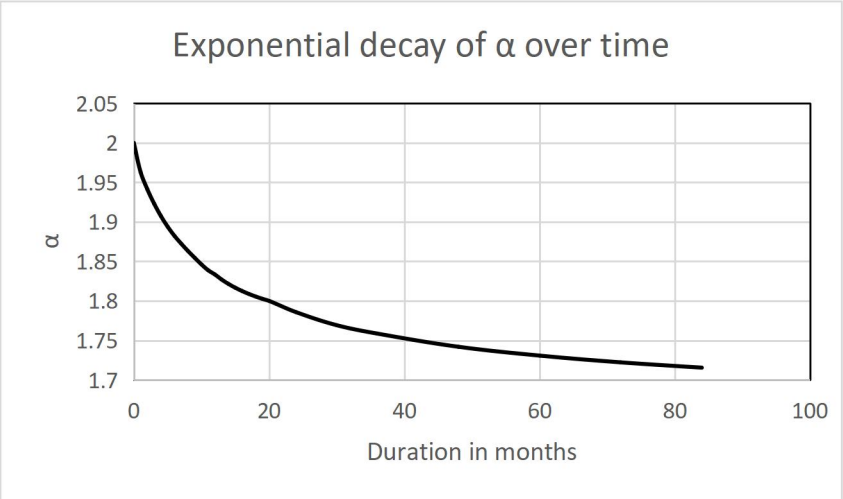
4. Weight Exponent (α)

$$\propto = 1.5 + (0.5 \times \frac{12}{D + 12})$$

Where D = Project Duration in months

The weight exponent controls how the sharpness of the disbursement curve increases across phases such that short projects have a higher α (steeper curve) meaning they will receive D*i* disbursements in earlier phases and funds are released in higher increments in the later phases, ensuring the earlier phases are very small tests and if a project fails in the middle phases contributors retain a higher percentage of the remaining funds. Essentially α mitigates risk for short duration investments. The curve decreases towards 1.5 as D increases, flattening the distribution for longer projects as shown in Figure 5.3.

Figure 5- 3: Exponential decay of α with increasing project duration, showing steeper curves for short projects and flatter curves for long ones.



5. Phase Count (P)

$$P = round(\ FRF + NFRT)$$

$$P = \ clamp(P, 3, 12)$$

Where $3 \leq P \leq 12$ to maintain manageable checkpoints and $D_{ref} = 12$ to provide a fixed baseline of 1 year for the ratio D/D_{ref}. The Financial Risk Factor is scaled to the prototype ceiling F' = KES. 1,000,000 ensuring that the phase count never exceeds maximum number of 12.

Non-Financial Risk Term (NFRT)

$$NFRT = 3 + 1.5\frac{1}{C} + 0.03\frac{D}{D_{ref}} + \ \text{CT\_phase}_{\text{adj}}$$

Maximum Financial Risk Factor (FRF_{Max})

$$FRF_{Max} = \ P_{Max} - NFRT$$

Where the $P_{max} = 12$.

Financial Risk Factor (FRF)

$$FRF = \ \frac{F}{F'} \times FRF_{Max}$$

The FRF ensures that when a campaign reaches a prototype ceiling of F' the financial-risk contribution reaches its allowable maximum FRF_{Max}. For smaller funding goals the financial risk contribution scales down proportionally to maintain stability and prevent financial inputs from overpowering the non-financial risk factors.

Where

F: Total campaign goal in KES

F': Prototype ceiling for normalization (KES 1,000,000)

C: Company Risk factor

D: Project duration

FRF_{Max}: Maximum allowable financial-risk component

FRF: Scaled financial Risk contribution in the phase count

D_{ref}: Duration baseline of 12 months

6. **Seeding phase and Phase Calibration**

The seeding phase is a pre-phase and is consequently left out from the calculation of the phase count P. Its duration is calculated as

$$D_{Seed} = \max(0.1D, 0.1\ months)$$

This ensures that even short-duration campaigns of ≤ 1 month have a meaningful seeding interval of approximately 3 days. Upon conclusion of the seeding phase and the achievement of the funding goal threshold, the active project duration becomes

$D_{active} = D - D_{Seed}$

The system then divides the active duration evenly across all phases

$$D_P = \frac{D_{active}}{P}$$

The first disbursement occurs at the end of the seeding phase, meaning that phase 1 verification and fund release occur immediately after seed phase completion.

7. Final Remedial Reserve (R_m)

The remedial reserve represents the minimal amount of funds a system must retain for potential refunds, milestone safety and early termination scenarios. Its computed in two stages: a base remedial reserve and a final remedial reserve that incorporates campaign type and duration influences.
The base remedial reserve is calculated as:

$$R_{m0} = 0.15 - 0.05(FTI)$$

The 0.15 represents the baseline reserve before any adjustments, ensuring that all campaigns maintain a mandatory safety buffer. The trust based discount of 0.5(FTI) reduces the reserve by up to 5% for highly reliable fundraisers while ensuring that trust never eliminates the need for protective reserves.

Final Remedial Reserve is calculated as:

$$R_m = clamp(\ R_{m0} + CT\ risk_{adj} + 0.02\ \frac{D}{D_{ref}}, 0.05, 0.25)$$

The term shifts the reserve depending on the risk profile of the campaign type and 2% duration term adds a small increase for longer projects due to accumulated uncertainty over a longer period of time ensuring projects retain a larger safety buffer without overwhelming shorter projects whose early-phase protections are handled by the α weighting.

During the prototype implementation, the remedial reserve R_m is kept as the undistributed portion of campaign funds within the platform. It is not transferred to the fundraiser during the campaign and is only released upon final milestone verification. In case one of the milestones fails or the project is terminated, the prototype refunds contributors using the reserve balance. This keeps the prototype operational while maintaining the intended risk-mitigation logic of the model.

8. Phase Weights (W_i)

$$W_i = \frac{i^{\propto}}{\sum_{k=1}^{P} k^{\propto}}, i = 1...p$$

Where $i^{\propto}$ ensures funds are incrementally weighted towards the later stages of a project where i is the phase index and as it increases the disbursed amount gets larger and α is the Weight Exponent based on the project duration ensures for exponential growth in fund disbursement based off of the length of a project's tenure. The denominator $\sum_{k=1}^{P} k^{\propto}$ calculates the total sum of all the weighted phase values ensuring the sum of all W_i is always 1 regardless of the number of phases P or the weight exponent α. This guarantees that the full distributable amount is allocated proportionally across all phases.

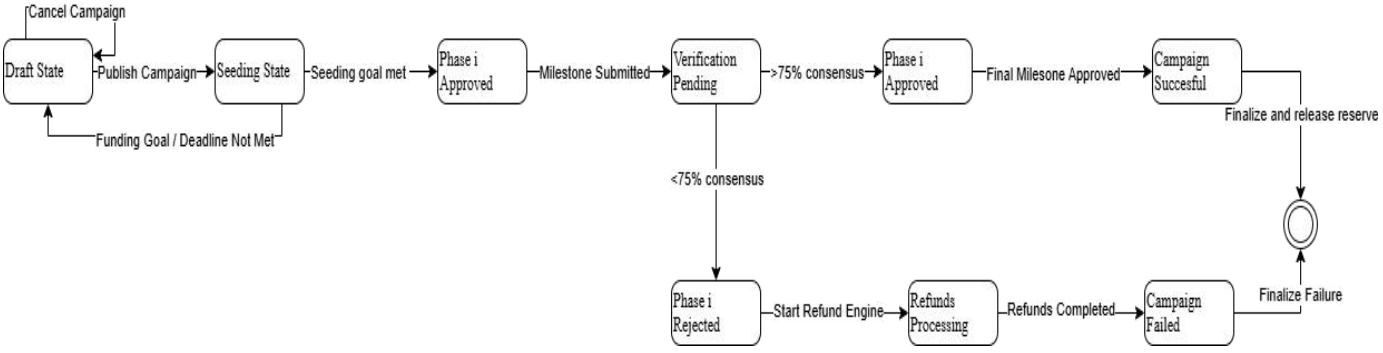
9. Phase Distribution (D_i)

$$D_i = (1 - R_m).W_i$$

Where $(1 - R_m)$ ensures the reserve remedial buffer is deducted first and is multiplied against the weighted distribution to ensure the distributed amount is resultant of the risk weighted requirement of the specific milestone.

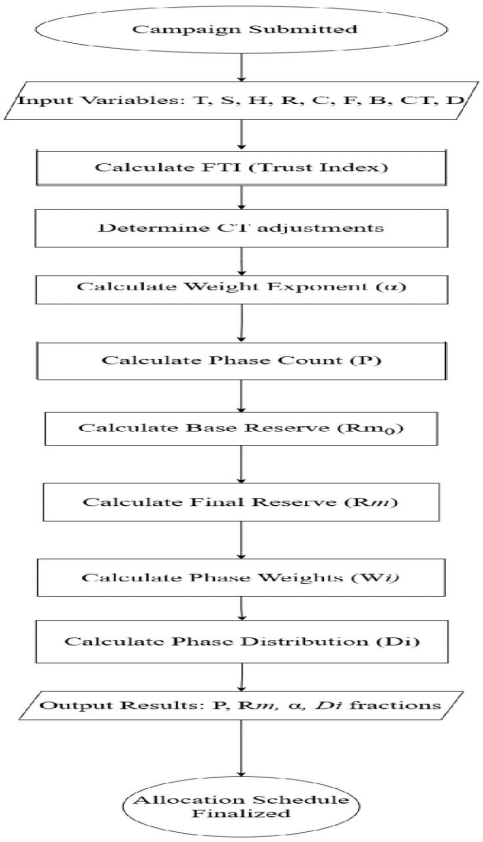
5.2.4 State Diagram

Figure 5- 4State diagram for the Campaign states



5.2.5 Flow Chart

Figure 5- 5Flow Chart for Algorithmic Fund Allocation



5.2.6 Digital Signature-Based Voting

The system employs a lightweight digital-signature mechanism to ensure the milestone approval rating is contributor-controlled and tamper-proof, where contributor devices automatically generate a cryptographic public-private key pair. The private key remains securely stored in the contributor device, while the public key is transmitted to the backend and stored within the campaign's vote token. This binds each contribution to a unique cryptographic identity without requiring user involvement.

When a milestone enters its voting window, the contributor's device generates a minimal voting payload containing the campaign ID, milestone ID, vote token, timestamp, nonce and the contributor's decision whether yes or no. The payload is then hashed using the Keccak-256 algorithm and the resulting hash is digitally signed using the contributor's private key and the signed hash together with the vote is transmitted to the system.

The backend validates the vote by reconstructing the payload, re-computing the Keccak hash and verifying its signature using the public key stored in the vote token. If a signature is valid the embedded vote value hashed in the system is read and the system determines if the contributor

issued an approval or rejection. Due to the vote being a part of the signed data no entity can alter a “yes” to a “no” and only correct signed votes are counted.

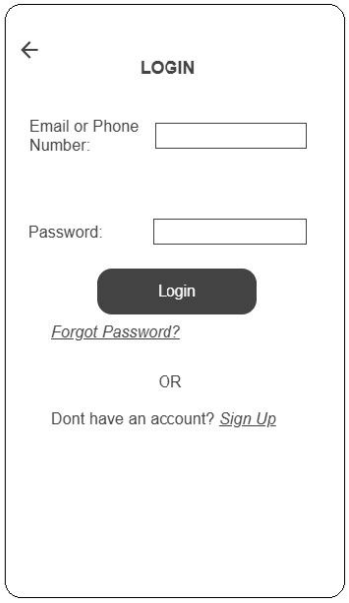
A milestone is only approved when at least 75% of votes are “yes”, triggering the disbursement of the amount from the simulated escrow. In the event the threshold is not attained, the system rejects the milestone and triggers the refund logic. This digital signature approach upholds contributor autonomy, secures the voting process by ensuring its tamper-proof and minimizes system overhead.

5.3 Interface Design

This section presents the interface design of the Secure Crowdfunding Platform, illustrating the layout and navigation of key user screens for both contributors and fundraisers

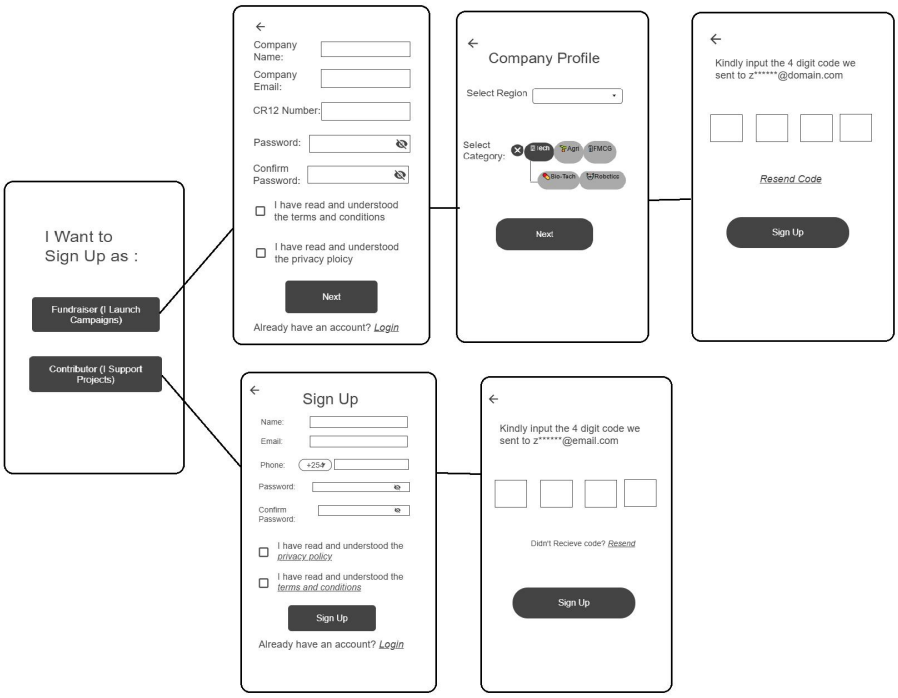
5.3.1 Login Screen Wireframe

Figure 5- 6: Login Page Wireframe created in wireframe.cc



5.3.2 Sign Up Page Wireframe

Figure 5- 7:Sign Up page Wireframe created in wireframe.cc

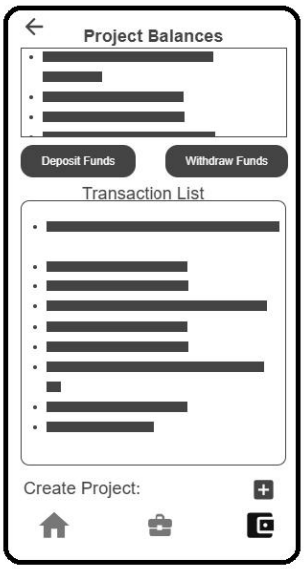


5.3.3 Fundraiser Dashboard Wireframe

Figure 5- 8Dashboard wireframe created in wireframe.cc

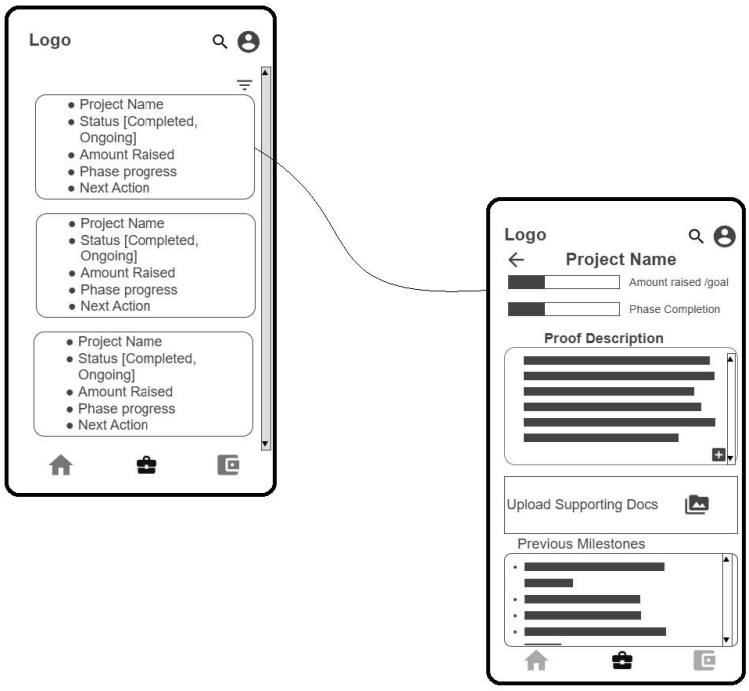
5.3.4 Fundraiser Overview Page Wireframe

Figure 5- 9 Fundraiser overview page created in wireframe.cc



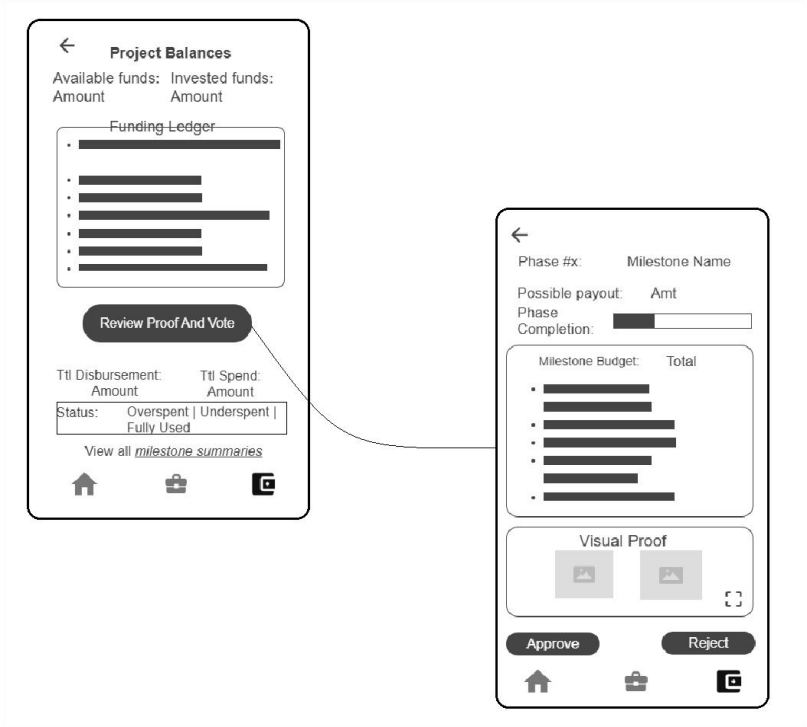
5.3.5 Fundraiser Active Projects List Wireframe

Figure 5- 10Fundraiser active projects created in wireframe.cc



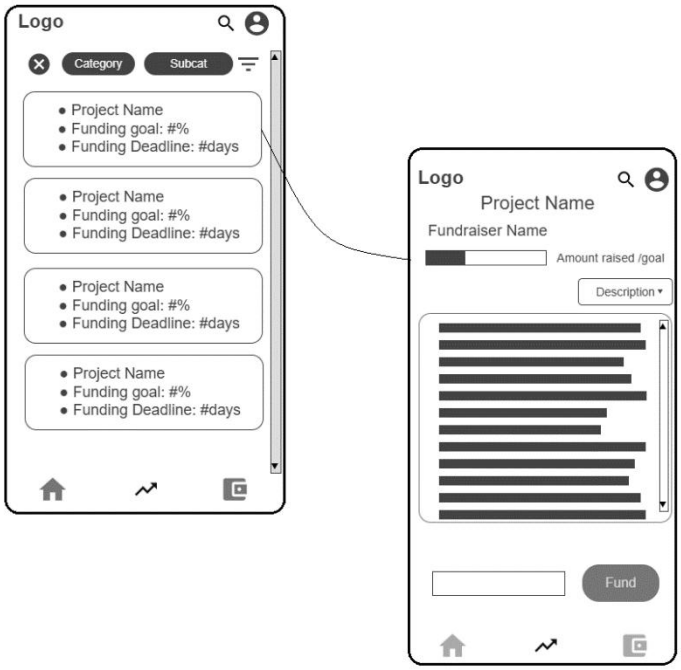
5.3.6 Contributor Overview-Page Wireframe

Figure 5- 11Contriibutor Overview page created in wireframe.cc



5.3.7 Contributor Discovery Page Wireframe

Figure 5- 12Contributor Discovery Page created in wireframe.cc



5.4 Database Design

The database design defines the logical and physical structure of the data used within the Secure Crowdfunding Platform with Phased Fund Release Using Hash-Validated Voting. It ensures data related to users, campaigns and contributions, escrow states, voting records and milestone transition is stored efficiently, consistently and securely. The design supports transparency. This design supports transparency, auditability and integrity of transactions throughout the system.

The database development process follows a structured approach comprising of:

1. Identification of core entities,
2. Defining relationships among entities,
3. Normalization to eliminate redundancy and
4. Creation of well-structured relational tables

PostgreSQL is used as the primary Relational Database Management System due to its ACID compliance, strong relational integrity, JSON support and cryptographic extensions.

5.4.1 Entity Identification

The following entities were identified from the functional requirements.

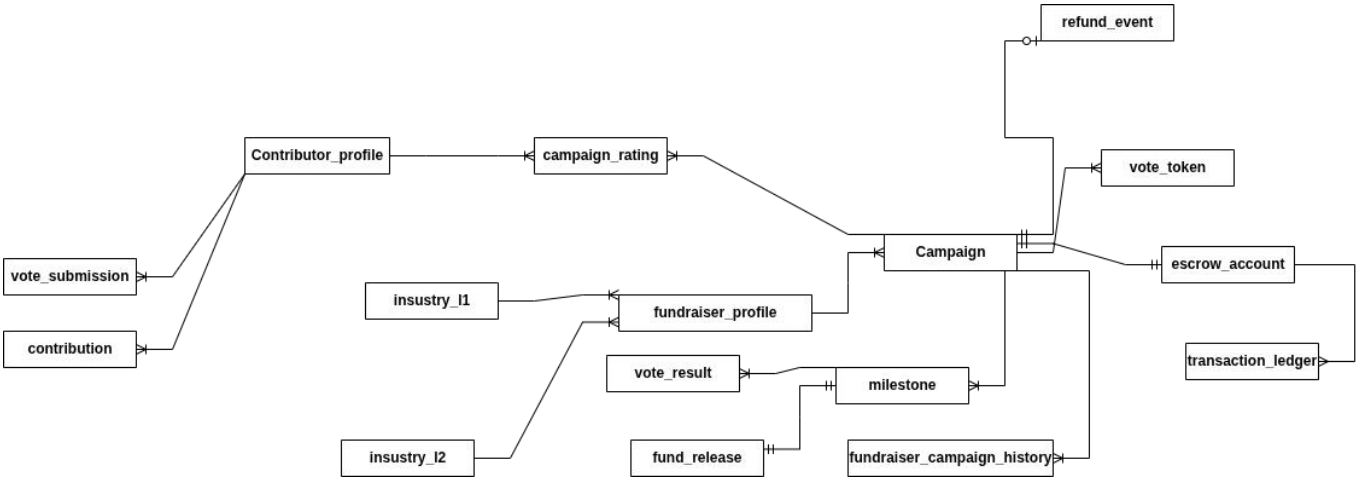
1. User – Stores contributor and fundraiser information.
2. Campaign – Holds campaign details created by the fundraiser.
3. Milestone – Stores project milestones and voting checkpoints.
4. Contribution – Record contributor payments
5. EscrowLedger – Maintains held, released or refunded fund states.
6. Vote – Records hashed contributor votes per milestone.
7. TransactionLog – Stores event History for auditability.

These entities for the core data layer for the platform.

5.4.2 Entity-Relationship Diagram (ERD)

Primary Relationships

Figure 5- 13ERD showing the primary relationships



These relationships collectively ensure referential integrity across the fund-release workflow.

The table structures are further elaborated in 5.4.4 with the key descriptions.

5.4.3 Normalization Process

First Normalization Form (1NF)

All attributes contributes atomic values and no repeating groups are present.

Second Normalization Form (2NF)

All non-key attributes are fully dependent on the primary key and composite dependencies are removed.

Third Normalization Form (3NF)

No transitive dependencies remain.

The final design is fully normalized to 3NF, for minimal redundancy and data consistency.

5.4.4 Table Structures

Below are the main database tables with Primary Keys (PK) and Foreign Keys (FK).

1) Account Tables

These tables store the system's authentication and user identity information.

The account table is portioned by role using PostgreSQL LIST partitioning scheme to logically separate contributors from fundraisers

a) account

Account		
Field	**Type**	**Description**

account_id(PK)	UUID	Unique user identifier for login
email	VARCHAR(120)	User's Unique email address
password_hash	TEXT	Encrypted user password
role	ENUM ('fundraiser', 'contributor)	User role
email_verified	BOOLEAN	True when email OTP is confirmed
created_at	TIMESTAMP	Account creation timestamp

Table 5.1account table structure

b) **account_contributor**:

Partition on the role field where (role = 'contributor').

account_contributor: Partition		
Field	**Type**	**Description**
(Inherits all fields from account)		

Table 5.2Contributor Partition

c) account_fundraiser:

Partition on the role field where (role = 'fundraiser').

account_fundraiser: Partition		
Field	**Type**	**Description**
(Inherits all fields from account)		

Table 5.3Fundraiser Partition

d) Contributor Profile Table

The contributor profile extends the contributor's account record with additional personal details required for transactions.

contributor_profile		
Field	**Type**	**Description**
contributor_id (PK/FK)	UUID	References account_contributor.account_id
uname	VARCHAR(120)	Contributor's name
phone_number	VARCHAR(20)	M-Pesa registered phone number
created_at	TIMESTAMP	Timestamp of profile creation

Table 5.4Contributor Profile Table

e) Fundraiser Profile Table

This table stores company specific identity and industry classification data which links each fundraiser to their verified business entity and associated risk category.

fundraiser_profile		
Field	**Type**	**Description**
fundraiser_id (PK/FK)	UUID	Links to account_fundraiser.account_id
company_name	VARCHAR(200)	Registered business name
br_number	VARCHAR(50)	Business Registration Number
industry_l1_id (FK)	UUID	Broad industry category (Tech, Health, etc.)
industry_l2_id (FK)	UUID (nullable)	Optional subcategory (EdTech, BioMed, etc.)
risk_score_C	DECIMAL(4,3)	Computed risk factor
br_verified	BOOLEAN	TRUE if BR verification successful
created_at	TIMESTAMP	When the profile was created

Table 5.5Fundraiser Profile Table

2) Industry Category Lookup Tables

These lookup tables provide the hierarchical categorization used in computation of the company's risk factor (C), where the L1 category represents broad industry grouping and the optional L2 category refines the classification.

a) Company category L1

Field	**Type**	**Description**
l1_id (PK)	UUID	Category ID
l1_name	VARCHAR(100)	Broad category (Tech, Finance, Health…)
l1_risk_weight	DECIMAL(4,3)	Base risk multiplier

Table 5.6Company Category L1 Table

b) Company category L2

Field	**Type**	**Description**
l2_id (PK)	UUID	Subcategory ID
l1_id (FK)	UUID	Link to parent broad category
l2_name	VARCHAR(100)	Subcategory (EdTech, BioMed, FinTech…)
l2_risk_weight	DECIMAL(4,3)	Adjusts L1 risk weight

Table 5.7Company Category L2 Table

c) Risk Parameter Mapping Table

This table stores configurable algorithmic parameters used in calculating the company risk factor C and it allows for dynamic tuning of the risk model without modifying the backend code.

company_risk_mapping		
Field	**Type**	**Description**
mapping_id (PK)	UUID	-
l1_weight	DECIMAL(3,2)	0.70
l2_weight	DECIMAL(3,2)	0.30
default_l2_adjust	DECIMAL(4,3)	0.06
min_risk	DECIMAL(4,3)	Lower bound
max_risk	DECIMAL(4,3)	Upper bound

Table 5.8Risk Parameter Mapping Table

3) Campaign Table

a) Campaign table

campaign		
Field	**Type**	**Description**
campaign_id (PK)	UUID	Unique identifier
fundraiser_id (FK)	UUID	Links to fundraiser profile
title	VARCHAR(200)	Campaign title
description	TEXT	Campaign description
funding_goal_F	DECIMAL(12,2)	Funding goal
duration_D	INT	Total project duration (months)
campaign_type_CT	ENUM('donation')	MVP campaign type
funding_start_date	TIMESTAMP	When contributions open
funding_end_date	TIMESTAMP	Funding deadline (computed automatically)
category_C	DECIMAL(4,3)	Cached risk factor C
num_phases_P	INT	Computed phase count
alpha_value	DECIMAL(4,3)	Weight exponent α
remedial_reserve_Rm	DECIMAL(4,3)	R_m value
total_contributions	DECIMAL(12,2)	Sum of contributions

total_released	DECIMAL(12,2)	Sum of released funds
status	ENUM('draft','active','funded','in_phases','completed','failed')	Campaign state
created_at	TIMESTAMP	Creation timestamp
updated_at	TIMESTAMP	Last updated

Table 5.9Campaign Table

b) Escrow Account

escrow_account		
Field	**Type**	**Description**
escrow_id (PK)	UUID	Escrow account ID
campaign_id (FK)	UUID	One escrow per campaign
total_contributions	DECIMAL(12,2)	Running total of all inflows
total_released	DECIMAL(12,2)	Running total of all outflows
balance	DECIMAL(12,2)	Current balance = inflow − outflow
updated_at	TIMESTAMP	Last ledger update

Table 5.10Escrow Account Table structure

c) **Contribution Table**

Contribution		
Field	**Type**	**Description**
contribution_id (PK)	UUID	Unique payment record
campaign_id (FK)	UUID	Campaign being funded
contributor_id (FK)	UUID	Who contributed
amount	DECIMAL(12,2)	Amount contributed
payment_status	ENUM('pending','success','failed')	Payment outcome
created_at	TIMESTAMP	Timestamp of contribution

Table 5.11Contribution Table

d) Transaction Ledger

transaction_ledger		
Field	**Type**	**Description**
transaction_id (PK)	UUID	Unique escrow event
escrow_id (FK)	UUID	Which campaign escrow
amount	DECIMAL(12,2)	Positive (in) or negative (out)
transaction_type	ENUM('contribution','release','refund','adjustment')	Ledger classification
reference_id	UUID	FK to contribution OR fund_release OR refund_event
created_at	TIMESTAMP	Timestamp

Table 5.12 Transaction Ledger Table

e) Fund Release

fund_release		
Field	**Type**	**Description**
release_id (PK)	UUID	Disbursement event ID
campaign_id (FK)	UUID	Campaign whose phase was approved
milestone_id (FK)	UUID	Phase that triggered fund release
amount_released	DECIMAL(12,2)	Actual disbursement amount
released_at	TIMESTAMP	Time of release

Table 5.13Fund Release table

f) Refund Event

refund_event		
Field	**Type**	**Description**
refund_id (PK)	UUID	Refund event ID
campaign_id (FK)	UUID	Campaign that failed
contributor_id (FK)	UUID	Person refunded
amount_refunded	DECIMAL(12,2)	Refund amount
refund_reason	TEXT	“Milestone failure”, “Voting timeout”, etc.
refunded_at	TIMESTAMP	Timestamp

Table 5.14Refund Event Table

g) Campaign Rating

Campaign_rating		
Field	**Type**	**Description**
campaign_rating_id (PK)	UUID	Unique rating entry
campaign_id (FK)	UUID	Campaign being rated
contributor_id (FK)	UUID	Contributor who rated
rating_value	INT	Rating value from 1 to 5
comment	TEXT	Optional rating feedback
created_at	TIMESTAMP	When the rating was submitted

Table 5.15Campaign Rating Table structure

h) Fundraiser Campaign History

fundraiser_campaign_history		
Field	**Type**	**Description**
history_id (PK)	UUID	Unique record for each completed campaign
fundraiser_id (FK)	UUID	Links to fundraiser_profile.fundraiser_id
campaign_id (FK)	UUID	Campaign that was completed

is_successful	BOOLEAN	TRUE if campaign completed all milestones
is_high_budget	BOOLEAN	TRUE if campaign exceeded the high-budget threshold
completed_at	TIMESTAMP	Timestamp when campaign was completed

Table 5.16Camapaign History Table structure

4) Milestone Table

milestone		
Field	**Type**	**Description**
milestone_id (PK)	UUID	Unique milestone/phase identifier
campaign_id (FK)	UUID	Links to the parent campaign
phase_index	INT	Phase number (1 to P)
phase_weight_Wi	DECIMAL(6,5)	Normalized weight Wi
disbursement_percentage_Di	DECIMAL(6,5)	$Di = (1 - R_m) \times Wi$
release_amount	DECIMAL(12,2)	Di × funding_goal_F (stored for quick lookup)
vote_window_start	TIMESTAMP	When voting on this milestone opens
vote_window_end	TIMESTAMP	When voting ends
status	ENUM('pending','approved','rejected','released')	Current milestone state
created_at	TIMESTAMP	Timestamp for milestone creation

Table 5.17Milestone Table structure

5) Governance and Hash-Validated Voting Tables

a) vote_result

vote_result		
Field	**Type**	**Description**
result_id (PK)	UUID	Unique tally record
milestone_id (FK)	UUID	Milestone whose votes were tallied
total_yes	INT	Number of YES votes
total_no	INT	Number of NO votes
quorum	INT	Total valid votes counted
yes_percentage	DECIMAL(5,2)	Percentage of YES votes
outcome	ENUM('approved','rejected')	Final milestone decision
tallied_at	TIMESTAMP	Timestamp when final tally was completed

Table 5.18Vote Result table Structure

b) vote_token

vote token		
Field	**Type**	**Description**
token_id (PK)	UUID	Unique identifier for the token entry
campaign_id (FK)	UUID	Campaign for which this token grants voting rights
contributor_id (FK)	UUID	Contributor who is allowed to vote on this campaign
token_hash	TEXT	Hashed token used for vote verification (Keccak-256)
created_at	TIMESTAMP	When the voting token was generated

Table 5.19Vote Token Table Structure

c) vote_submission

vote_submission		
Field	**Type**	**Description**
vote_id (PK)	UUID	Unique identifier for the vote
milestone_id (FK)	UUID	Milestone being voted on
contributor_id (FK)	UUID	Contributor who submitted the vote
vote_hash	TEXT	Hashed (token + vote_choice) used for verification
vote_value	ENUM('yes','no')	The actual vote choice
submitted_at	TIMESTAMP	Timestamp of vote submission

Table 5.20Vote Submission Table Structure

5.4.5 Database Integrity and Constraints

To maintain system trustworthiness:

- Foreign keys enforce relation consistency.
- Check constraints restrict invalid states such as approval percentage ≥ 75%
- Unique constraints prevent multiple votes per contributor for each milestone.
- Triggers log critical events automatically into Transaction_Ledger.
- Hash Validation rules ensure immutability of votes and milestone proof files.

This ensures the system is transparent, auditable and meets security requirements outlined in Chapter 6.

5.4.6 Summary

The database design establishes a secure, normalized and efficient data model for the crowdfunding platform. Integration of relational constraints, cryptographic hashing and escrow state tracking ensures the system maintains a high level of transparency, contributor oversight and traceability across all funding phases. This database schema emphasizes the cryptographically governed phased fund release mechanisms and provides a foundation for system implementation.

REFERENCES

Business Registration Service Act, cap. 499B (Laws of Kenya 2015).

Freedman, D. M., & Nutting, M. R. (2015). *Equity Crowdfunding for Investors A Guide to Risks, Returns, Regulations, Funding Portals, Due Diligence and Deal Terms.* New Jersey: Wiley & Sons.

Hassija, V., Chamola, V., & Zeadally, S. (2020, May 17). *ScienceDirect*. Retrieved from BitFund: A Blockchain-based crowdfunding platform for future smart and connected nation: https://www.sciencedirect.com/science/article/abs/pii/S2210670720301323

Kiende, D. (2021). *Factors that determine success of crowdfunding initiatives in Kenya.* KCA University, Nairobi.

Li, K., Li, H., Hou, H., Li, K., & Chen, Y. (2017). Proof Of Vote: A High-Performance Consensus Protocol Based on Vote Mechanism & Consortium Blockchain. *2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International* (pp. 466-473). IEEE. Retrieved from IEEE Xplore.

Musau, M. M. (2020, October 6). *Harambee: The law of generosity that rules Kenya*. Retrieved from BBC: https://www.bbc.com/travel/article/20201004-harambee-the-kenyan-word-that-birthed-a-nation

Ronald, M. (2017). The Influence of Financial Resources on the Success of Start-up Business in Kenya. *IRA-International Journal of Management & Social Sciences (ISSN 2455-2267)*. doi:http://dx.doi.org/10.21013/jmss.v8.n2.p8

Solorio, K., Kanna, R., & Hoover, D. H. (2019). *Hands-On Smart Contract Development with Solidity and Ethereum From Fundamentals to Deployment.* O'Reilly Media, Inc.

TinyGrab. (2025, June 17). *HOW DOES KICKSTARTER MAKE MONEY?* Retrieved from TinyGrab: https://tinygrab.com/how-does-kickstarter-make-money

Wang, J. G., Ma, H. J., & Chen, Y. Z. (2018). *Financing from Masses: Crowdfunding In China.* Singapore: Springer Nature.

Xu, Y., Li, Q., Zhang, C., Tan, Y., Zhang, P., Wang, G., & Zhang, Y. (2023, August). *A decentralized trust management mechanism for crowdfunding*. Retrieved from ScienceDirect: https://www.sciencedirect.com/science/article/abs/pii/S0020025523005388#se0020

Zkik, K., Sebbar, A., Fadi, O., Kamble, S., & Belhadi, A. (2024, April 27). *Springer Nature Link*. Retrieved from Securing blockchain-based crowdfunding platforms: an integrated graph

neural networks and machine learning approach: https://link.springer.com/article/10.1007/s10660-023-09702-8

APPENDICES

I) RESOURCES REQUIRED/BUDGET

Hardware costs			
Item	**Specification**	**Availability**	**Estimated cost in KES**
Developer Laptop	Intel core i5 13th generation, 2 cores 4 logical cores 2.71GHz CPU, 16GB RAM, 512GB SSD	Available(Personal)	Existing
Electricity	Stable Supply for development	Available(Personal)	Existing
Internet Access	10 Mbps broadband internet	Required	2,000/month
Subtotal			**2000**

Table 5.21Hardware costs

Software Costs		
Software	**Purpose**	**Estimated Cost in KES**
Visual Studio Code	Development Environment for frontend and backend	Free
Flutter SDK + Android Studio Emulator	App Development And Testing Environment	Free
Python 3.10 + FastAPI	Backend Development	Free
PostgresSQL + Redis	Database and caching	Free
M-Pesa Daraja Sandbox	API Payment Simulation	Free
Docker	Containerization and environmental consistency	Free
Git + Github	Version Control	Free
Subtotal		**0**

Table 5.22Software costs breakdown

Human Resources		
Role	**Responsibility**	**Estimated Cost in KES**
Project Developer	System design, development, testing	N/A (student)

Supervisor/ Technical consultant	Guidance and evaluation	N/A (Academic)
Subtotal		**0**

Table 5.23Human resources cost breakdown

Miscellaneous Resources		
Item	**Description**	**Estimated cost in KES**
Printing and binding	Final report submission	320
Stationery	Pens and notebooks	Existing
Subtotal		**320**

Table 5.24Miscellaneous cost breakdown

Summary of Total Estimated Budget	
Category	**Total Cost In KES**
Hardware costs	2000
Software Costs	0
Human Resources	0
Miscellaneous Resources	320
Subtotal	**2320**

The outlined system resources allow for optimal system development, simulation and testing in a realistic environment. The project leverages open-source development tools to minimize cost and provide robust functionality, while the hardware and software requirements facilitate efficient development of the system prototype.

II) TIME PLAN

Figure 5- 14 Gantt Chart for System Investigation Phase

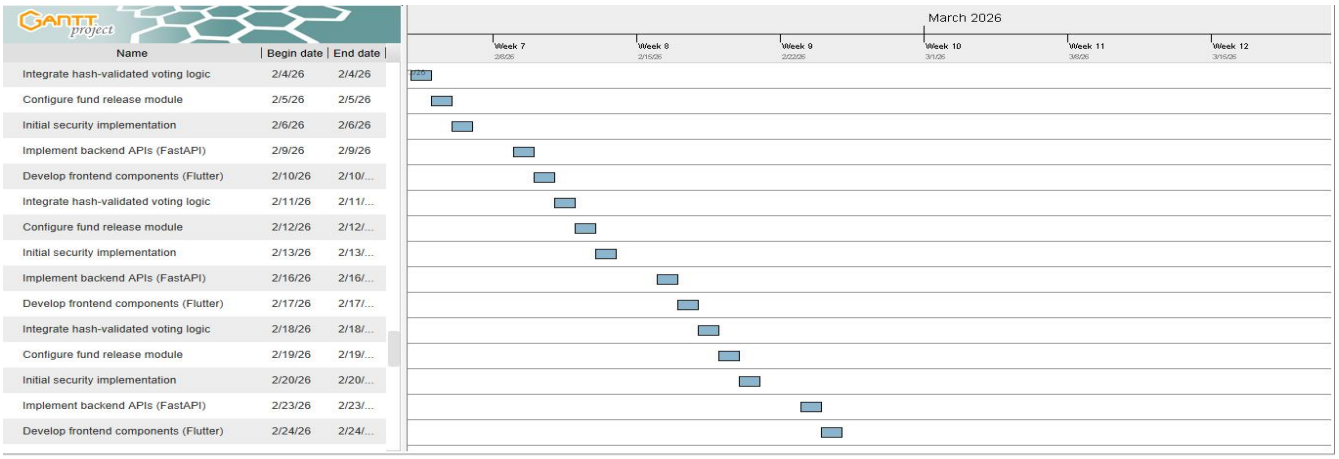
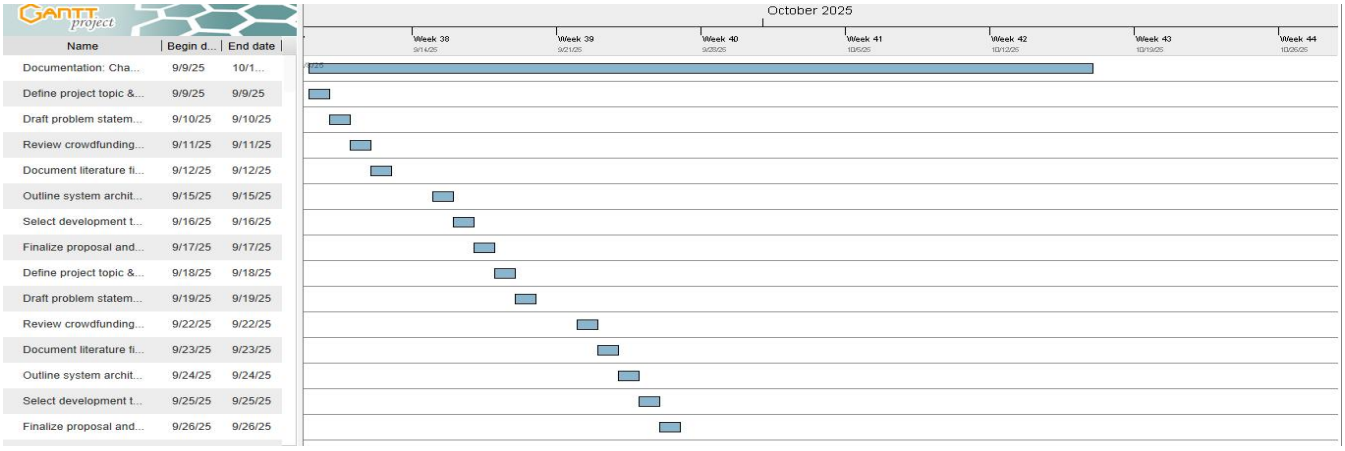
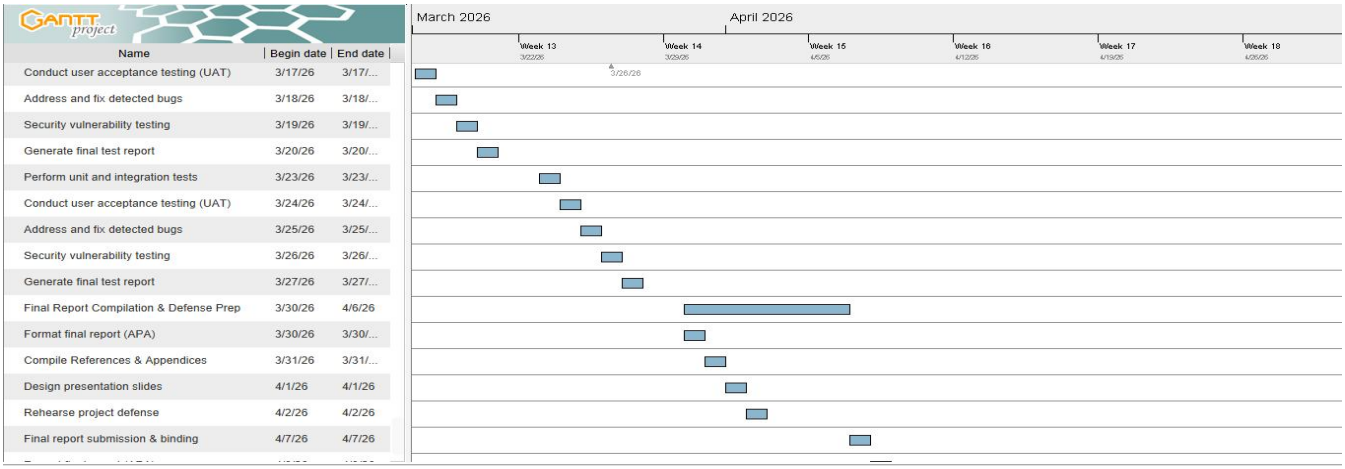


Figure 5- 15 Gantt Chart for System Analysis and Requirements Definition



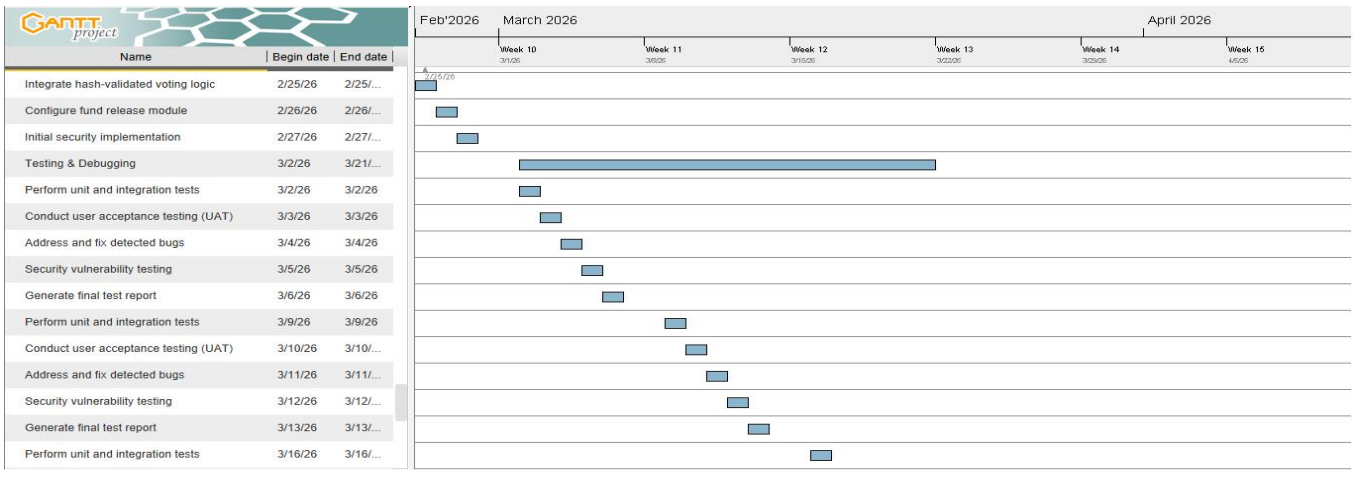
Name	Begin d...	End date
Documentation: Cha...	9/9/25	10/1...
Define project topic &...	9/9/25	9/9/25
Draft problem statem...	9/10/25	9/10/25
Review crowdfunding...	9/11/25	9/11/25
Document literature fi...	9/12/25	9/12/25
Outline system archit...	9/15/25	9/15/25
Select development t...	9/16/25	9/16/25
Finalize proposal and...	9/17/25	9/17/25
Define project topic &...	9/18/25	9/18/25
Draft problem statem...	9/19/25	9/19/25
Review crowdfunding...	9/22/25	9/22/25
Document literature fi...	9/23/25	9/23/25
Outline system archit...	9/24/25	9/24/25
Select development t...	9/25/25	9/25/25
Finalize proposal and...	9/26/25	9/26/25

Figure 5- 16 Gantt Chart for System Design (Phase I)



Name	Begin date	End date
Conduct user acceptance testing (UAT)	3/17/26	3/17/...
Address and fix detected bugs	3/18/26	3/18/...
Security vulnerability testing	3/19/26	3/19/...
Generate final test report	3/20/26	3/20/...
Perform unit and integration tests	3/23/26	3/23/...
Conduct user acceptance testing (UAT)	3/24/26	3/24/...
Address and fix detected bugs	3/25/26	3/25/...
Security vulnerability testing	3/26/26	3/26/...
Generate final test report	3/27/26	3/27/...
Final Report Compilation & Defense Prep	3/30/26	4/6/26
Format final report (APA)	3/30/26	3/30/...
Compile References & Appendices	3/31/26	3/31/...
Design presentation slides	4/1/26	4/1/26
Rehearse project defense	4/2/26	4/2/26
Final report submission & binding	4/7/26	4/7/26

Figure 5- 17 Gantt Chart for Database Design and Entity Relationship Modeling



Name	Begin date	End date
Integrate hash-validated voting logic	2/25/26	2/25/...
Configure fund release module	2/26/26	2/26/...
Initial security implementation	2/27/26	2/27/...
Testing & Debugging	3/2/26	3/21/...
Perform unit and integration tests	3/2/26	3/2/26
Conduct user acceptance testing (UAT)	3/3/26	3/3/26
Address and fix detected bugs	3/4/26	3/4/26
Security vulnerability testing	3/5/26	3/5/26
Generate final test report	3/6/26	3/6/26
Perform unit and integration tests	3/9/26	3/9/26
Conduct user acceptance testing (UAT)	3/10/26	3/10/...
Address and fix detected bugs	3/11/26	3/11/...
Security vulnerability testing	3/12/26	3/12/...
Generate final test report	3/13/26	3/13/...
Perform unit and integration tests	3/16/26	3/16/...

Figure 5- 18 Gantt Chart for System Design (Phase II): Interface Prototyping

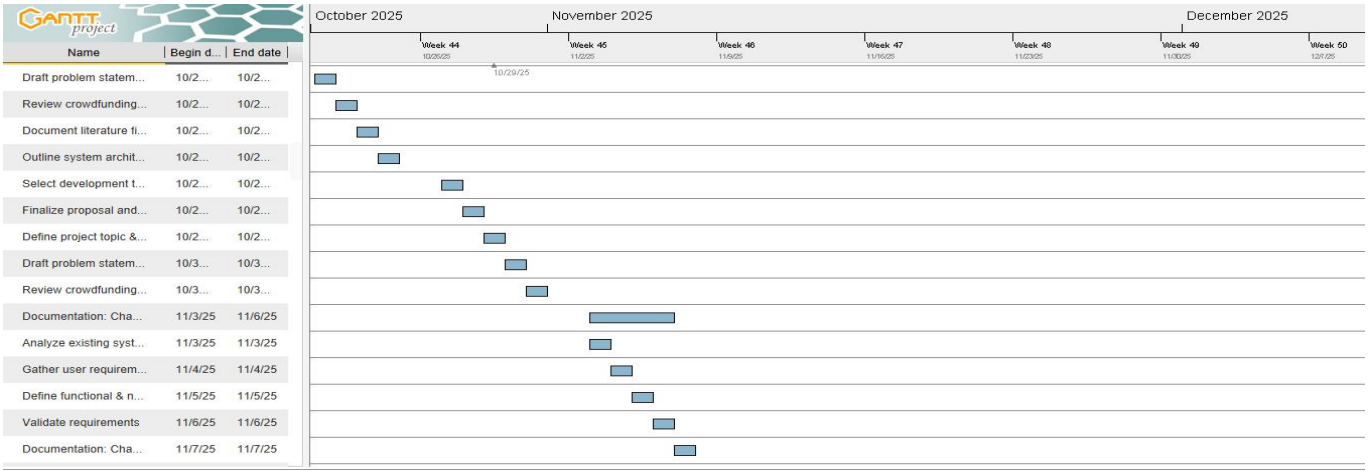


Figure 5- 19 Gantt Chart for Module 1: Authentication and Security Implementation

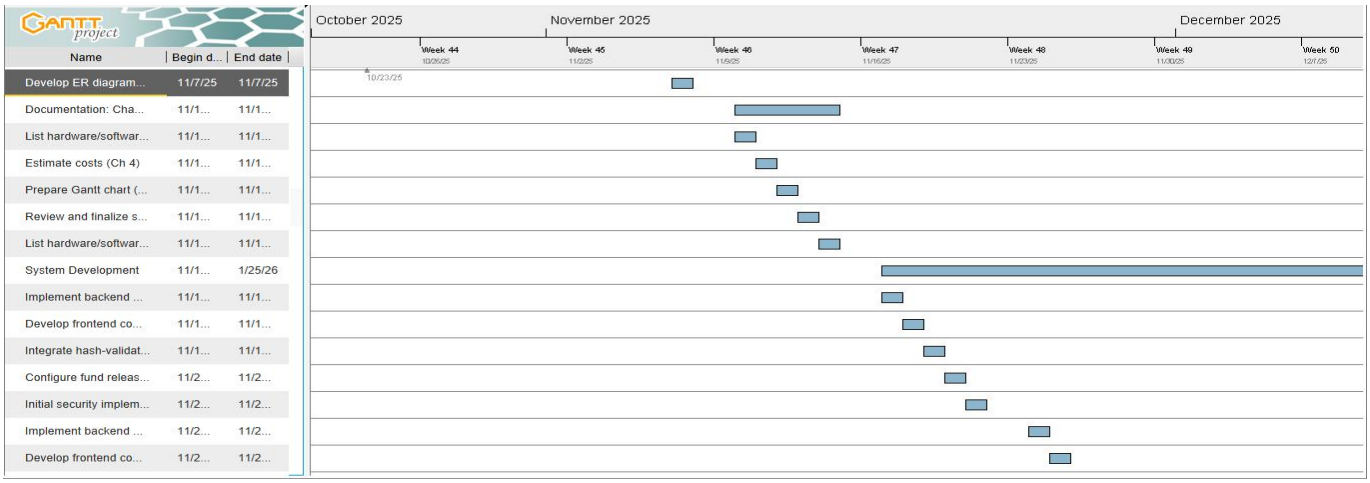


Figure 5- 20Module 2: Backend and Security Development

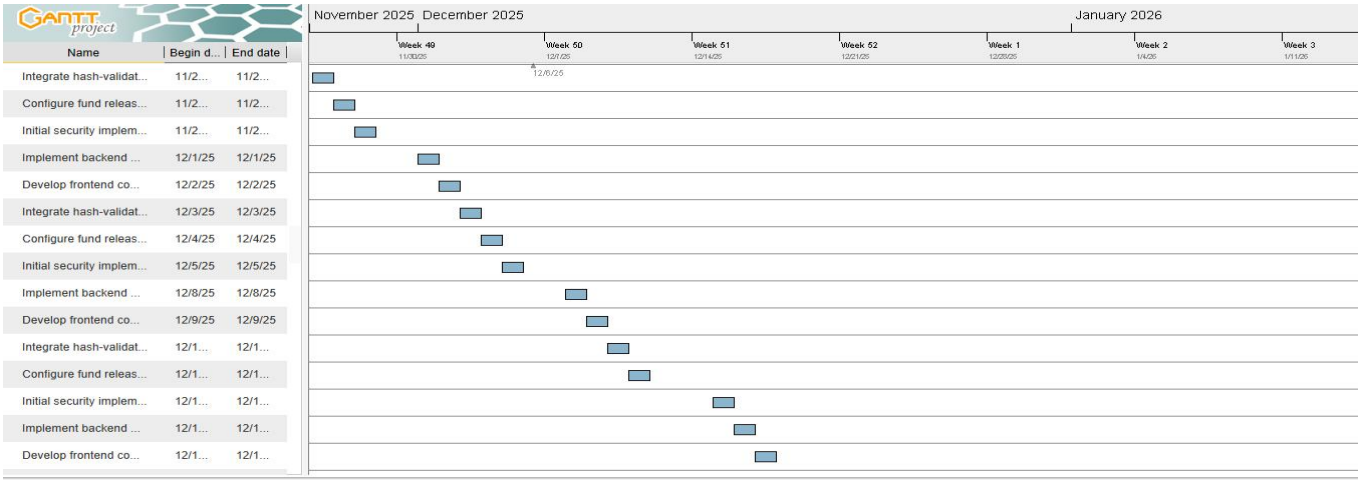


Figure 5- 21Module 3: Extended Backend API Development

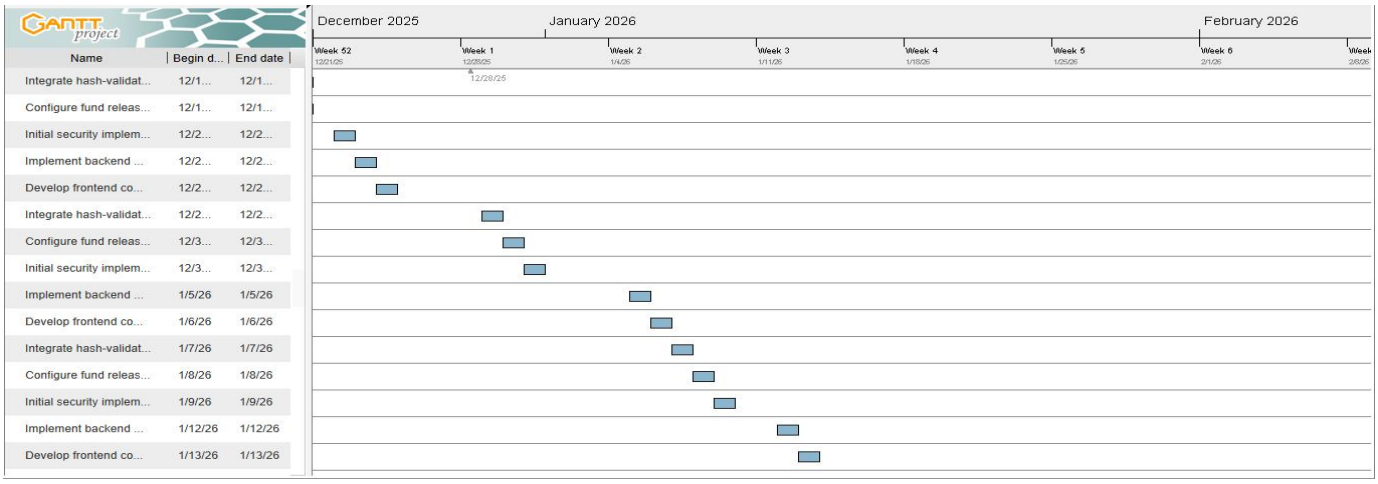


Figure 5- 22Module 4: Full Frontend Development Cycle