Collabits Journal
Vol. 2 No. 2 May 2025 : 123-132
E-ISSN : 1979-5254, P-ISSN : 3062-8601
https://publikasi.mercubuana.ac.id/index.php/collabits

Design And Construction of a Web-Based Crowdfunding Application Using the Laravel Framework

Foezi Arisandi SJ[1*]

[1] Computer Science, University of Politeknik Sukabumi, Indonesia

*Coressponden Author: foeziarisandi@polteksmi.ac.id

Abstract - This research uses RAD analysis research method, diagram design using UML, system coding using PHP programming language and Mysql database. This research produces a web-based ordering system that is connected to a database so that order data can be stored properly. This research aims to design and build a web-based crowdfunding application to address inefficiencies in conventional ordering and donation systems, particularly in industrial environments. The study highlights the increasing importance of digital fundraising and the role of crowdfunding as an effective platform for connecting fund-seekers with potential donors. Using the Laravel framework and MySQL database, the system was developed to improve transparency, accessibility, and transaction management. The research applies to the Waterfall development method and includes structured interviews and observation as part of the requirement-gathering process. The result is a functional, secure, and scalable web application that enables users to create, manage, and donate to campaigns with real-time tracking and integrated payment systems. This system enhances organizational outreach and improves the efficiency of donation and fund distribution processes.

Keywords :
Crowdfunding;
Laravel;
Fundraising;
Donation System;
Web-based Application;

Article History:
Received: 06-04-2025
Revised: 12-05-2025
Accepted: 18-05-2025

Article DOI : 10.22441/collabits.v2i2.35925

1. INTRODUCTION

In the digital age, the advancement of information technology has significantly transformed how individuals and communities interact, collaborate, and solve problems. From education and government to creative industries and social initiatives, digital platforms now serve as powerful tools for engagement and innovation. One such area that has seen rapid growth is crowdfunding a collaborative model where people pool their resources to support causes, products, or projects they believe in.

Crowdfunding platforms bridge the gap between creators and supporters by offering a transparent, accessible, and scalable environment for fundraising. However, despite the widespread use of global crowdfunding services, there remains a growing need for customized, locally relevant web-based solutions particularly those designed to support community-based projects, student initiatives, social movements, and startups with unique needs and cultural contexts.

To address this need, this research focuses on the design and development of a web-based crowdfunding application using the Laravel framework. Laravel, a modern PHP framework known for its elegant syntax and robust features, provides a flexible foundation for building secure and scalable web applications. By utilizing Laravel, the system aims to simplify the process of campaign creation, user contribution, progress tracking, and administrative oversight while ensuring a seamless user experience.

This journal explores the technical and conceptual steps involved in building a crowdfunding platform from the ground up. It discusses the system architecture, development workflow, and core features implemented to support real-time fundraising, campaign verification, and user interaction. Through this project, the study highlights how web technologies can be leveraged not just for commercial gain, but also for fostering social impact and collective action in the digital space.

2. THEORETICAL FOUNDATION

2.1 General Theory

2.1.1 Definition of Crowdfunding

A According to **Belleflamme, Lambert, and Schwienbacher** (2014), “Crowdfunding involves an open call, mostly through the internet, for the provision of financial resources either in the form of donation or in exchange for some form of reward and/or voting rights to support initiatives for specific purposes.”

According to **Mollick** (2014), “Crowdfunding is the

effort by entrepreneurial individuals and groups—cultural, social, and for-profit—to fund their ventures by drawing on relatively small contributions from a relatively large number of individuals using the internet, without standard financial intermediaries."

According to **Ordanini et al.** (2011), "Crowdfunding is a collective effort by people who network and pool their resources to support efforts initiated by other people or organizations. This is usually done via the internet."

According to **Gerber and Hui** (2013), "Crowdfunding enables individuals to support others' creative work through online platforms. It builds community and trust among participants while allowing creators to receive not only financial support but also feedback, attention, and validation."

Based on the above definitions, crowdfunding can be defined as an online-based fundraising method that enables individuals, groups, or organizations to collect small financial contributions from a large number of people—commonly known as "the crowd." These contributions can be donations, investments, loans, or exchanges for rewards or products, and are often facilitated through dedicated online platforms such as Kickstarter, GoFundMe, Indiegogo, or local equivalents.

Crowdfunding not only serves as an alternative financing tool, especially for startups and creative projects, but also creates engagement between creators and supporters by offering a sense of participation, community, and transparency. The success of crowdfunding campaigns is often driven by the power of networks, social media, trust, and storytelling.

2.1.2 Types of Crowdfunding

According to the **European Commission (2015)**, crowdfunding can generally be categorized into four main types, based on the **nature of return expected by contributors**. Each type reflects different motivations and funding structures:

a. Donation-Based Crowdfunding

In this type, contributors donate money to a project or cause **without expecting any material return**. The motivation is typically altruistic, such as supporting social, humanitarian, or charitable causes. According to **Belleflamme et al. (2014)**, donation-based crowdfunding is often used by **nonprofits, individuals in need, or social campaigns**.

Example: A campaign to raise money for disaster relief or medical treatment via platforms like Kitabisa or GoFundMe.

b. Reward-Based Crowdfunding

Contributors provide financial support in exchange for a **non-financial reward**, often a product, service, or exclusive content. As stated by **Mollick (2014)**, this type is **commonly used in creative industries** like film, games, and tech startups where early supporters receive prototypes or merchandise. The reward is often symbolic or tied to the project's outcome.

Example: Kickstarter campaigns offering early access to a new gadget or name credits in a film.

c. Equity-Based Crowdfunding

In equity-based crowdfunding, contributors become **investors** by acquiring shares or ownership stakes in the company or project. They may receive **dividends or capital gains** if the project becomes profitable. According to **Ahlers et al. (2015)**, this model is suitable for **startups and SMEs** looking for seed capital, and requires **regulatory oversight** in most countries due to its investment nature.

Example: Investing in a tech startup through platforms like Seedrs or EquityNet.

d. Lending-Based Crowdfunding (Peer-to-Peer Lending)

Also known as **P2P lending**, this model allows contributors to lend money to individuals or businesses **with the expectation of repayment**, often with **interest**. According to **Agrawal, Catalini, and Goldfarb (2015)**, this type is similar to traditional loans but conducted via **digital platforms** that match borrowers and lenders without intermediaries like banks.

Example: A small business borrowing capital via platforms like Kiva or Funding Societies.

Each type is suitable for different kinds of projects—social, creative, entrepreneurial, or commercial—and involves varying **degrees of risk, regulation, and engagement**. Understanding these types is essential for designing a campaign strategy that matches the goals and audience.

2.1.3 Crowdfunding Platform

Crowdfunding platforms act as digital intermediaries that connect fundraisers with potential backers, enabling transparent and secure financial transactions. According to Ordanini et al. (2011), these platforms provide the structure and tools that facilitate interaction between fund-seekers and funders. Agrawal, Catalini, and Goldfarb (2015) further emphasize that platform features such as visibility, ease of use, and credibility significantly influence the success of crowdfunding campaigns.

2.1.4 Success Factors in Crowdfunding

The success of a crowdfunding campaign is influenced by multiple interrelated factors. According to Mollick (2014), key elements include the quality and clarity of the project presentation, the credibility and background of the initiator, the ability to gain early contributions (early traction), and the effective use of personal networks and social media for promotion. These aspects help build trust and increase visibility, which are crucial in attracting potential backers. Furthermore, Belleflamme, Lambert, and Schwienbacher (2014) add

that the size of the supporting crowd, the design and attractiveness of offered incentives, and the overall reputation of the crowdfunding platform significantly impact campaign performance. Therefore, a successful crowdfunding strategy must integrate strong communication, social outreach, and strategic planning to maximize engagement and funding potential.

2.1.5 Definition of Fundraising

According to Sargeant and Jay (2014), "Fundraising is the process of soliciting and gathering voluntary contributions of money or other resources, by requesting donations from individuals, businesses, charitable foundations, or governmental agencies."

According to Tempel, Seiler, and Burlingame (2016), "Fundraising involves both strategic planning and interpersonal skills to build relationships with donors and secure necessary resources."

Fundraising is a strategic and relational effort to support an organization's mission. It requires planned communication, donor cultivation, and the ability to build trust and shared commitment with supporters.

2.1.6 Digital Transformation in Fundraising

Digital transformation has significantly reshaped the landscape of fundraising. As noted by Saxton and Wang (2014), the rise of the internet and social media has enabled organizations to reach wider audiences more quickly and engage them through personalized appeals. This transformation allows fundraising efforts to be more dynamic and interactive, fostering deeper connections with potential donors.

Similarly, Manchanda and Muralidharan (2020) emphasize the impact of digital tools such as crowdfunding, which have disrupted traditional fundraising models. These tools have not only made it easier for individuals and organizations to seek funds, but also empowered donors to support causes in a more direct and transparent way.

The digital era has enhanced fundraising by making it more efficient, accessible, and far-reaching, with crowdfunding platforms enabling broader participation and aligning with modern communication trends.

2.1.7 Observation in Crowdfunding Campaigns

According to Zhang and Liu (2012), "Observation in crowdfunding includes monitoring campaign performance, donor behavior, and funding trends in real-time to adjust strategies."

According to Lehner (2013), "Analyzing behavioral patterns and feedback during campaigns allows creators to optimize content, rewards, and marketing efforts for better outcomes."

Observation in crowdfunding is a crucial process to analyze donor behavior, campaign trends, and platform dynamics, allowing strategic adjustments for improved fundraising performance.

2.2 Specialized Theory

2.2.1 PHP

PHP (Hypertext Preprocessor) is a widely-used open-source scripting language designed specifically for web development and can be embedded into HTML. It is executed on the server side, generating dynamic content before it is sent to the client's browser.

According to Welling and Thomson (2017), PHP allows developers to create interactive and data-driven websites efficiently. It supports various databases, including MySQL, PostgreSQL, and SQLite, and integrates well with modern frameworks such as Laravel.

PHP is known for its flexibility, simplicity, and active community support. Its syntax is easy to learn, making it a popular choice for beginners and professionals alike. Over the years, PHP has evolved with the release of modern features such as object-oriented programming, improved security mechanisms, and better performance optimizations (Lerdorf, 2007; Suraski & Gutmans, 2020).

2.2.2 MySQL Database

MySQL is a widely used **open-source relational database management system (RDBMS)** that relies on Structured Query Language (SQL) to manage data. It is particularly popular in web application development due to its performance, reliability, and ease of integration with backend frameworks like Laravel. MySQL supports fundamental database operations such as data **insertion, retrieval, updating, and deletion**, and allows developers to design normalized schemas for efficient data management.

In a relational database such as MySQL, **data is organized into tables (also known as relations)** where each table consists of rows (records) and columns (attributes). This tabular structure supports complex queries, relationships between entities, and ensures **data integrity** through constraints such as primary keys and foreign keys.

As stated by Elmasri and Navathe (2016), relational databases provide a formal structure for organizing and manipulating data, which improves **scalability, consistency, and ease of access** to large datasets.

Laravel supports MySQL natively, allowing seamless database integration through the .env configuration file. Laravel's **Eloquent ORM (Object-Relational Mapping)** allows developers to interact with MySQL databases using PHP syntax rather than writing raw SQL queries, thus improving **developer productivity** and **code readability**. Laravel also includes **migration tools**, which make it

easy to version, modify, and share the database schema across development teams.

2.2.3 Laravel Framework

Laravel is a modern, open-source PHP web application framework designed to make common development tasks such as routing, authentication, and caching more streamlined and elegant. It follows the **Model-View-Controller (MVC)** architectural pattern, which separates application logic from presentation, making the codebase more manageable and scalable.

According to Stauffer (2019), Laravel provides a robust set of tools and resources for building modern PHP applications, including **Eloquent ORM**, **Blade templating engine**, **middleware**, and **artisan command-line tool**. The framework emphasizes code readability and developer productivity, which contributes to its popularity in building scalable and secure web applications.

Laravel also includes built-in mechanisms for **database migrations**, **validation**, **session management**, and **API development**, making it suitable for both monolithic and RESTful applications. Integration with **MySQL**, **PostgreSQL**, and other database systems is seamless through Laravel's database abstraction layer.

2.2.4 Web-Based System

A web-based system is an application that operates through a web browser using the internet or an intranet. It allows access to system functionalities without the need for installing software locally on each device. According to Sommerville (2011), web-based systems are advantageous due to their ease of maintenance, platform independence, and centralized data management.

In this research, the system built is web-based to ensure accessibility, real-time interaction, and compatibility across multiple devices. This architecture supports both user convenience and system scalability.

2.2.5 JSON (JavaScript Object Notation)

JSON (JavaScript Object Notation) is a lightweight and human-readable data-interchange format that is widely used in modern web applications. It is structured as key-value pairs and supports data types such as strings, numbers, arrays, and objects. JSON facilitates seamless communication between frontend clients and backend servers, especially in RESTful APIs or AJAX-based requests.

In Laravel, JSON plays a crucial role in handling API responses, allowing asynchronous data transmission between the server and the client. Laravel provides built-in functions such as response()->json() which simplify the process of returning data in JSON format. This enables efficient integration with frontend JavaScript frameworks such as Vue.js or React, and supports mobile app backends or external system integrations (Rahman et al., 2021).

The usage of JSON also supports the implementation of dynamic and real-time web applications. With tools such as Axios or Fetch API on the client side, JSON responses from Laravel backends are used to update page content without reloading, improving user experience and performance (Putra & Santoso, 2020).

2.2.6 UML (Unified Modeling Language)

Unified Modeling Language (UML) is a standardized visual language that provides a set of graphical notations to create abstract models of systems. UML is particularly valuable in object-oriented software development as it allows developers to design and communicate the structure and behavior of a system in a unified and understandable way (Booch, Rumbaugh, & Jacobson, 2005).

In the context of Laravel-based development, UML plays a critical role during the system analysis and design phases. Laravel, being a PHP-based MVC (Model-View-Controller) framework, benefits significantly from the use of UML diagrams to ensure clear planning of its components.

Several UML diagrams are commonly used in Laravel system development:

- **Use Case Diagrams**: Represent the interaction between users (actors) and the system, helping to define functional requirements and understand user needs.
- **Class Diagrams**: Visualize the structure of the application by showing classes, attributes, methods, and the relationships among objects. This is particularly relevant in Laravel for modeling Eloquent ORM-based models and relationships.
- **Sequence Diagrams**: Describe how objects interact in a particular scenario of a use case, detailing the sequence of messages exchanged between system components.
- **Activity Diagrams**: Capture the dynamic aspects of the system by modeling workflows and business logic, which is useful in mapping Laravel controller actions and middleware processes.

Using UML in Laravel projects not only supports efficient system planning and documentation but also improves communication between technical teams and non-technical stakeholders, ensuring shared understanding of system requirements and architecture (Ambler, 2004).

2.2.7 RESTful API

RESTful API (Representational State Transfer) is an architectural style for designing networked applications. It uses HTTP methods like GET, POST, PUT, and DELETE for CRUD operations. RESTful APIs are stateless and provide standard communication between client and server.

Laravel facilitates the creation of RESTful APIs through routes, controllers, and JSON responses. This approach is essential for integrating the system with other platforms or mobile applications.

2.2.8 XAMPP

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB (formerly MySQL), and interpreters for scripts written in the PHP and Perl programming languages. It is widely used for local development and testing of web applications before deploying them to production servers. XAMPP simplifies the setup process for developers by bundling all necessary components in a single installation, making it a popular tool in PHP-based frameworks such as Laravel. With XAMPP, developers can simulate a server environment on their local machine, enabling them to develop and test their applications efficiently without needing a live server (Apache Friends, 2020).

2.2.9 Authentication and Authorization

According Authentication is the process of verifying the identity of users, while authorization determines the permissions granted to authenticated users. Laravel provides a built-in authentication system that supports features such as login, registration, password reset, and session management. Laravel also uses gate and policy mechanisms to implement authorization, which allow developers to define access rules for different user roles or models. These security layers are essential to ensure that only authorized users can access specific resources or perform certain actions (Stauffer & Cox, 2019). Laravel's authentication system is based on guards and providers, making it flexible for applications requiring role-based access control or token-based authentication like API guards using Laravel Sanctum or Passport (Laravel Documentation, 2025).

2.2.10 Visual Studio Code

Visual Studio Code (VS Code) is a lightweight yet powerful source-code editor developed by Microsoft. It supports a wide range of programming languages and frameworks, including PHP and Laravel. With features such as syntax highlighting, IntelliSense, Git integration, and a vast extension marketplace (e.g., Laravel Blade snippets, PHP Intelephense), VS Code significantly improves development productivity. Due to its versatility and rich ecosystem, VS Code is widely adopted among web developers working with Laravel applications (Microsoft, 2021).

2.2.11 MVC (Model-View-Controller) Architecture

MVC is a software design pattern that separates the application into three interconnected components: the **Model**, which handles the data and business logic; the **View**, which manages the user interface; and the **Controller**, which processes user input and updates the model and view accordingly. Laravel is built around the MVC architecture, allowing developers to organize code in a structured and maintainable way. This separation of concerns enhances scalability, facilitates debugging, and improves team collaboration during the development process (Katz & Shutt, 2021).

2.2.12 Routing in Laravel

Routing is a core component of Laravel that defines how the application responds to user requests via specific URLs. Laravel's routing mechanism allows developers to map Uniform Resource Locators (URLs) to corresponding controller actions or anonymous functions. The routes/web.php file handles routes for web interfaces, while routes/api.php is used for API routes. Laravel supports various HTTP methods such as GET, POST, PUT, and DELETE, enabling the development of RESTful web services (Stauffer, 2019).

With features like named routes, route grouping, and middleware assignment, Laravel offers a clean and expressive syntax for defining application behavior. For instance, routes can be protected with middleware to ensure that users are authenticated before accessing certain pages, thereby enhancing the security and maintainability of the application.

Routing not only simplifies the logic for request handling but also plays a crucial role in organizing the application structure within the MVC (Model-View-Controller) architecture. Laravel's routing system enhances productivity and code clarity by effectively separating concerns and allowing route definitions to remain both flexible and readable (Stauffer, 2019).

2.2.13 Payment Integration

In crowdfunding platforms, payment integration is a crucial component that ensures the transaction between donors and campaign creators is smooth, secure, and automated. Laravel supports various payment gateways through third-party packages such as **Laravel Cashier**, **Midtrans**, **Xendit**, or **Stripe**, enabling developers to implement complex transaction systems efficiently. These integrations commonly utilize **webhooks** to handle asynchronous payment notifications, ensuring that donation statuses are updated in real-time. Furthermore, payment validation mechanisms are essential to verify successful transactions before updating the campaign's status and donation amount (Midtrans,

2023; Laravel Cashier Documentation, 2024). Automating these processes reduces human error and builds trust among users by improving the transparency and reliability of the platform.

2.2.14 Testing and Validation

Testing and validation are essential stages in web development to ensure that the system meets its functional requirements and performs reliably under various conditions. In Laravel-based applications, developers often use **PHPUnit**, which is integrated by default, for unit and feature testing. This allows for the simulation of user interactions and system behaviors to verify that each component, such as donation processing or user registration, works as intended (Laravel Documentation, 2024). Additionally, **Postman** and **API testing tools** are commonly employed to validate API endpoints, especially for payment gateways and campaign updates (Restelli & Di Felice, 2023).

Validation is also critical in the context of data integrity, particularly when handling user input and financial transactions. Laravel provides built-in validation rules that help prevent invalid data from entering the database, which is essential for maintaining the reliability and security of crowdfunding operations. Continuous testing through methods like **Test-Driven Development (TDD)** improves code quality and ensures that new changes do not break existing functionalities (Beck, 2002). Through thorough testing and validation, developers can deliver a more robust and trustworthy platform to end users.

2.3 Literature Review

The literature review is a fundamental step in scientific research to understand various concepts, theories, and previous studies relevant to the chosen topic. According to Nasution (2003), a literature study is an essential part of the scientific method, aiming to explore theoretical foundations that support the resolution of the research problem. This process involves examining various documents such as academic journals, books, articles, and credible online sources.

In the context of crowdfunding web development, several prior studies have discussed the implementation of web technologies, including frameworks like Laravel, in building interactive and secure systems. Laravel, as a modern PHP framework, offers features such as routing, middleware, and Eloquent ORM that support application development based on the Model-View-Controller (MVC) architecture.

Furthermore, previous research has shown that the success of a crowdfunding platform is significantly influenced by user interface design, data transparency, and accurate verification systems. Therefore, the choice of framework and system architecture approach becomes a crucial aspect examined within this literature review.

By analyzing various relevant sources, the researcher gains a comprehensive understanding of both technical and non-technical approaches in building an effective and trustworthy crowdfunding platform.

3. METHODOLOGY AND RUNNING SYSTEM ANALYSIS

Research Metodologies

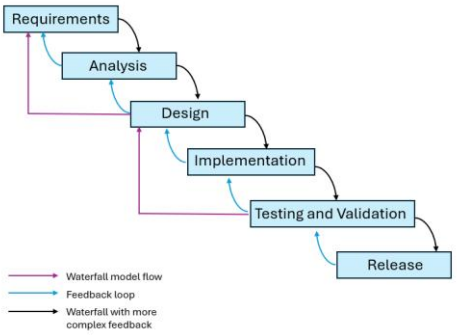

Figure 1. Waterfall Method

This study employs a web-based information system development approach, integrating various supporting methods that encompass the stages of data collection, system design, implementation, and testing. The methodological framework applied in this research is outlined as follows:

1. **Requirements Collecting**

The requirements gathering phase plays a pivotal role within the Waterfall development model, as it establishes a clear foundation for the entire project lifecycle. A thorough and structured analysis of requirements ensures that all stakeholders share a unified understanding of the project's scope, objectives, and expected outcomes.

This stage focuses on collecting detailed information regarding client needs, business goals, and relevant technical specifications. To achieve this, a range of effective techniques are employed to elicit, document, and validate business requirements:

- Interviews: Conducting individual or group discussions with stakeholders to gain in-depth insights into their expectations and needs.
- Surveys and Questionnaires: Distributing structured tools designed to collect both qualitative and quantitative data from a broad range of participants.
- Workshops: Organizing collaborative sessions that encourage active stakeholder engagement, idea exchange, and refinement of requirements.

- Document Analysis: Reviewing existing materials such as business process documents, technical reports, or previous system documentation to extract relevant information.
- Prototyping: Developing preliminary visual models or mockups to help stakeholders better understand system functionality and provide early feedback.

This phase lays the groundwork for all subsequent stages of system development, making clarity and completeness in this process essential for project success.

2. System Design

Types of design documents produced during this phase include:

- High-Level Design (HLD): Provides an overview of the system architecture, including major components and their interactions.
- Low-Level Design (LLD): Details the specific functionalities, data structures, and algorithms to be used.
- User Interface (UI) Design: Outlines the layout, navigation, and visual elements of the web application.
- Database Design: Specifies the database schema, including tables, relationships, and constraints.

3. System Design

The testing method applied in this study is Black Box Testing, which evaluates the system's functional requirements without considering the internal code structure. This approach helps identify issues such as missing or incorrect functions, interface problems, data handling errors, and faults during system initialization or termination. It ensures that the software behaves as expected from the user's perspective.

Existing System Analysis

This analysis aims to examine the current operational workflow related to crowdfunding activities, identify existing limitations or inefficiencies, and determine the necessary requirements for the development of the proposed web-based system.

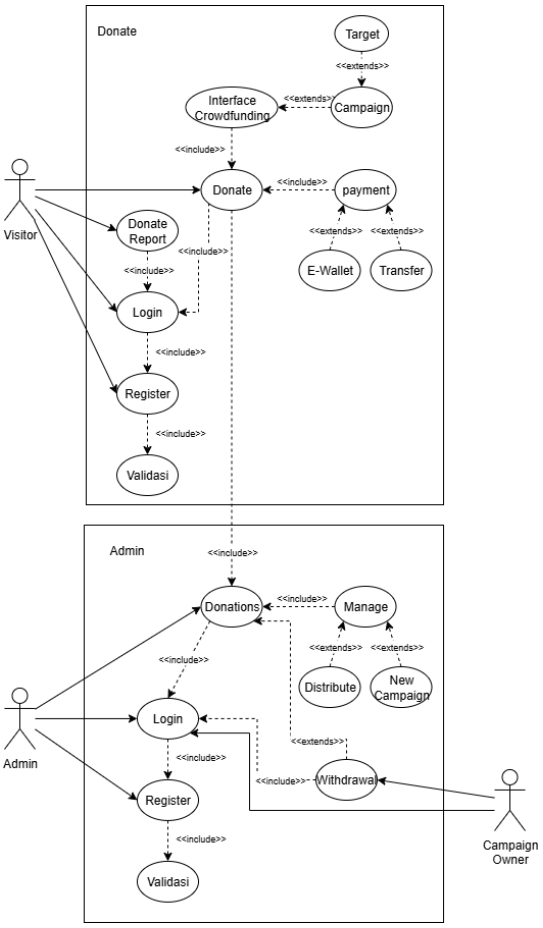

Figure 1. Use Case Diagram of the Running System

The interaction between three main actors: User, Campaign Creator, and Admin within a web-based crowdfunding system. Users can register, log in, validate their accounts, browse campaigns, and donate through various payment methods like E-Wallet or Transfer. They can also view donation reports after contributing. Campaign Creators have their own flow, including account registration, login, and the ability to create new campaigns, manage them, track donations, and initiate withdrawals. The Admin oversees campaign distribution and manages overall system activity. The diagram clearly separates user roles while highlighting shared functionalities such as registration and login.

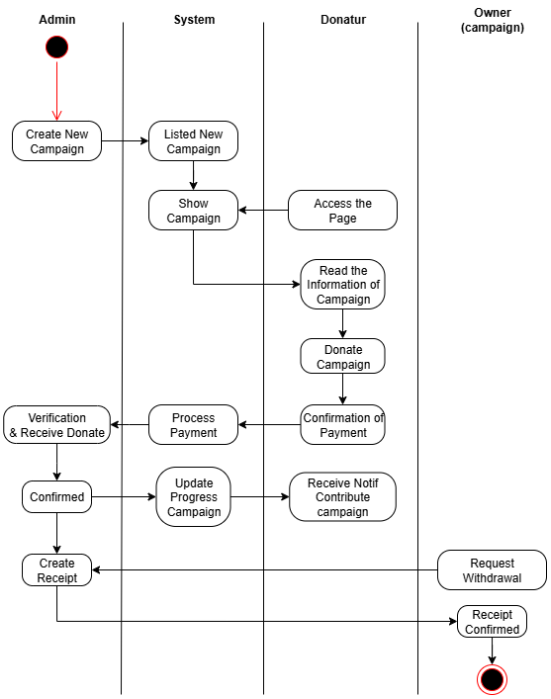

Figure 2. Activity Diagram of the Running System

This diagram illustrates the workflow between all actors involved in a web-based crowdfunding system. It begins with the admin creating a new campaign, which is then listed and made visible by the system. The campaign is shown to potential donors, who access the crowdfunding page and read the detailed information provided about the campaign.

Once interested, the donor proceeds to donate, followed by a payment confirmation step. The system processes the payment, updates the campaign progress, and triggers a notification to the donor to confirm that their contribution has been received. Simultaneously, the admin verifies and receives the donation, marks it as confirmed, and the system generates a receipt.

Campaign creators can then request a withdrawal of the collected funds. Once the request is approved, the system marks the receipt as confirmed, concluding the process. The entire workflow ensures transparency, proper verification, and clear communication between all parties involved.

Sequence Diagram of the Running System.

Login

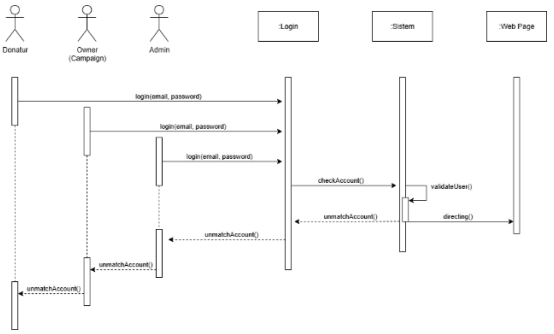

Figure 3. Login Sequence Diagram of the Running System

Admin & Owner (Campaign)

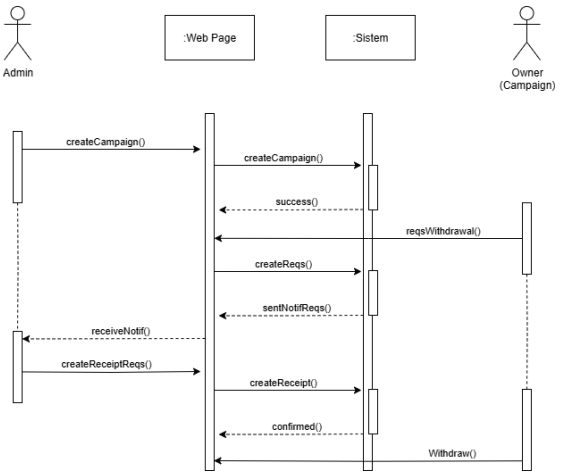

Figure 4. Admin & Owner Sequence Diagram of the Running System

Donatur

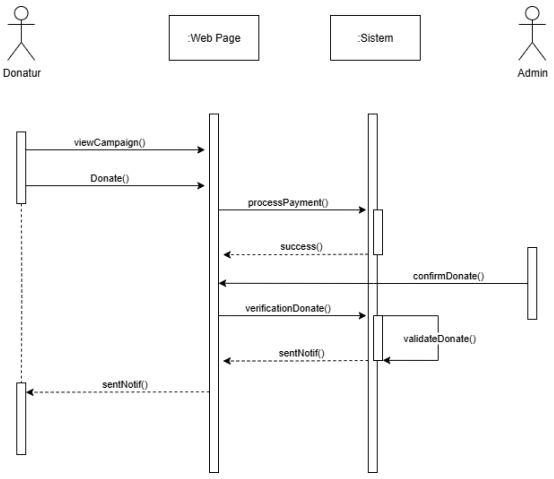

Figure 5. Donatur Sequence Diagram of the Running System

4. RESEARCH RESULTS

- Database Specification

Users

users			
Field	DataType	Length	Desc
id_users	int	11	PK
name	char	50	
email	char	100	
password	varchar	255	
role	enum(donor, admin, owner)		
created_at	timestamp		
updated_at	timestamp		

Campaigns

campaigns			
Field	DataType	Length	Desc
id	int	11	PK
title	char	50	
desc	varchar	255	
target_amount	int	11	
deadline	date		
image	varchar	255	
status	(enum: pending, active, completed, rejected)		
owner_id			FK -> users.id
created_at	timestamp		
updated_at	timestamp		

Donations

Donations			
Field	DataType	Length	Desc
id	int	11	PK
campaign_id	int	11	FK → Campaigns.id
donor_id	int	11	FK -> users.id
amount	int	11	
pay_method	enum(e-wallet, transfer)		
pay_stats	enum(pending, confirmed, failed)		
pay_proof	varchar (nullable)	255	
donated_at	timestamp		

Notifications

Notifications			
Field	DataType	Length	Desc
id	int	11	PK
user_id	int	11	FK -> users.id
message	varchar	255	
is_read	boolean		
created_at	timestamp		

Withdrawal

Withdrawal			
Field	DataType	Length	Desc
id	int	11	PK
campaign_id	int	11	FK → Campaigns.id
owner_id			FK -> users.id
amount_reqs	int	11	
pay_stats	enum(pending, approved, rejected)		
requested_at	timestamp		
approved_at	timestamp		

Receipts

Receipts			
Field	DataType	Length	Desc
id	int	11	PK
donation_id	int	11	FK → Donations.id
receipt_file	varchar (nullable)	255	
status	enum(confirmed, pending)		
issued_at	timestamp		

PaymentTransactions

PaymentTransactions			
Field	DataType	Length	Desc
id	int	11	PK
donation_id	int	11	FK → Donations.id
transaction_ref	text		
pay_gateways	(enum: midtrans, xendit, tripay, manual_transfer)		
amount	int	11	
status	(enum: pending, paid, expired, failed, refunded)		
payment_url	text		
paid_at	timestamp		
raw_response	text		
created_at	timestamp		
updated_at	timestamp		

6. REFERENCES

[1] Agrawal, A., Catalini, C., & Goldfarb, A. (2015). Crowdfunding: Geography, social networks, and the timing of investment decisions. *Journal of Economics & Management Strategy*, 24(2), 253–274.

[2] Alam, M., Rahman, M., & Hossain, M. (2022). An Overview of Laravel Framework and Its Application in Web Development. *International Journal of Computer Applications*, 184(5), 15–18.

[3] Ambler, S. W. (2004). The Object Primer: Agile Model-Driven Development with UML 2.0 (3rd ed.). Cambridge University Press.

[4] Belleflamme, P., Lambert, T., & Schwienbacher, A. (2014). Crowdfunding: Tapping the right crowd. *Journal of Business Venturing*, 29(5), 585–609.

[5] Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (2nd

ed.). Addison-Wesley.
[6] Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of Database Systems* (7th ed.). Pearson Education.
[7] European Commission. (2015). *Crowdfunding: Mapping EU markets and events study*. Publications Office of the European Union.
[8] Gerber, E. M., & Hui, J. (2013). *Crowdfunding: Why people are motivated to participate*. Innovation: Organization & Management, 15(4), 442–460.
[9] Lehner, O. M. (2013). *Crowdfunding social ventures: A model and research agenda*. Venture Capital, 15(4), 289–311.
[10] Lerdorf, R. (2007). *Programming PHP*. O'Reilly Media.
[11] Manchanda, R. V., & Muralidharan, C. (2020). Digital fundraising and its effectiveness in nonprofit organizations. *International Journal of Management*, 11(9), 60–70.
[12] Mollick, E. (2014). *The dynamics of crowdfunding: An exploratory study*. Journal of Business Venturing, 29(1), 1–16.
[13] Ordanini, A., Miceli, L., Pizzetti, M., & Parasuraman, A. (2011). *Crowd-funding: Transforming customers into investors through innovative service platforms*. Journal of Service Management, 22(4), 443–470.
[14] Putra, R. D., & Santoso, B. (2020). Implementation of JSON-Based Data Communication on Laravel and Vue.js. *International Journal of Informatics and Computation*, 2(1), 34–40.
[15] Rahman, A., Nugroho, Y., & Wicaksono, A. (2021). Application of RESTful API using Laravel Framework in Web-Based Information Systems. *Journal of Information Systems*, 15(2), 77–84.
[16] Sargeant, A., & Jay, E. (2014). *Fundraising Management: Analysis, Planning and Practice* (3rd ed.). Routledge.
[17] Saxton, G. D., & Wang, L. (2014). *The social network effect: The determinants of giving through social media*. Nonprofit and Voluntary Sector Quarterly, 43(5), 850–868.
[18] Suraski, Z., & Gutmans, A. (2020). *PHP Manual*. The PHP Group.
[19] Stauffer, M. (2019). *Laravel: Up & Running: A Framework for Building Modern PHP Apps* (2nd ed.). O'Reilly Media.
[20] Sommerville, I. (2011). *Software Engineering* (9th ed.). Pearson Education.
[21] Tempel, E. R., Seiler, T. L., & Burlingame, D. F. (2016). *Achieving Excellence in Fundraising* (4th ed.). Jossey-Bass.
[22] Welling, L., & Thomson, L. (2017). *PHP and MySQL Web Development* (5th ed.). Addison-Wesley.
[23] Zhang, J., & Liu, P. (2012). *Rational herding in microloan markets*. Management Science, 58(5), 892–912.