NUANSA INFORMATIKA
Volume 20 Number 2 July 2026

p-ISSN :1858-3911* , *e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

Design and Implementation of an Offline-First Mobile Grocery Arisan Information System Using Flutter and Firebase

Devina*[1], Putri Eli Sandra Hasibuan[2], Dzikra Azzahra[3], Adetia Anggriani[4], Gunawan[5]
[1,2,3,4,5]Politeknik Negeri Medan, Indonesia
E-mail: *[1]devina@students.polmed.ac.id, [2]putrielisandra05@gmail.com, [3]dzikraazzahra75@gmail.com, [4]adetyaanggriani@gmail.com, [5]gunawan@polmed.ac.id

Abstract

Conventional grocery savings schemes (arisan sembako) managed manually often encounter challenges such as inefficient contribution recording, limited transparency, and difficulties in managing participant data. This study aims to design a mobile-based grocery arisan information system to support the digitalization of community-scale grocery savings management. A descriptive research approach was employed, with data collected through observation and interviews conducted at Mila Sembako.The proposed system was designed using Flutter and Firebase and includes system architecture, use case modeling, Firestore database design, and interface mockup, with an offline-first synchronization mechanism to ensure data accessibility under limited or intermittent internet connectivity. The system supports participant management, grocery package management, contribution recording, winner selection, notification services, and cloud-based data synchronization through Firebase. By centralizing data management in a cloud environment, the system enables more efficient record keeping and improved information accessibility for administrators and participants. The resulting system design provides a structured framework for improving the efficiency, transparency, and reliability of grocery arisan management. The study demonstrates the feasibility of utilizing Flutter and Firebase technologies to support the digital transformation of community-based grocery savings schemes and serves as a reference for future system development and implementation.

Keywords—*grocery savings scheme, mobile information system, Flutter, Firebase, offline-first synchronization*

Submission: May 17, 2026 *Accepted: July 03, 2026* *Published: July 13, 2026*
DOI Article: ***https://doi.org/10.25134/ilkom.v20i2.582***

1. INTRODUCTION

Arisan is a traditional rotating savings activity that has long been embedded in Indonesian society as both a financial mechanism and a means of strengthening social relationships [1]. Along with the development of community needs, various forms of arisan have emerged, including *grocery arisan* (*arisan sembako*), in which participants periodically receive packages of essential goods such as rice, cooking oil, sugar, and other household necessities instead of cash [2]. Compared with conventional cash-based arisan, grocery arisan provides direct economic benefits by helping households fulfill their daily necessities while maintaining a collective savings mechanism.

Despite its benefits, grocery arisan management is still predominantly conducted manually in many small businesses and community organizations. Based on observations and interviews conducted at Mila Sembako, participant registration, contribution collection, payment recording, winner determination, and report preparation are managed manually using notebooks and paper-based records. This approach frequently leads to recording errors, duplicated data, difficulties in monitoring participant payment status, and limited transparency for both administrators and participants. In addition, administrators must spend considerable time verifying payment records and preparing weekly reports, reducing operational efficiency. These challenges indicate the need for a digital information system capable of supporting more accurate, transparent, and efficient grocery arisan management that remains reliable even under limited or unstable internet connectivity, a condition commonly encountered in community-scale operations [3], [4], [5].

The rapid advancement of mobile technology and cloud computing has significantly transformed the development of community-based information systems. Mobile information systems enable users to access services anytime and anywhere through smartphones, while cloud platforms facilitate centralized data storage and real-time

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

information exchange among multiple users. Flutter has become one of the most widely adopted cross-platform mobile application frameworks because of its capability to support a single codebase for multiple operating systems. Furthermore, Firebase provides cloud-based services, including Cloud Firestore, user authentication, and real-time data synchronization, as well as offline data persistence that allows applications to continue functioning without continuous internet access, which simplify mobile application development while ensuring data consistency across devices. These technologies have been successfully implemented in various mobile information systems due to their scalability, development efficiency, and cloud integration capabilities [6], [7], [8], [9].

Several previous studies have explored the digitalization of community financial management through different technological approaches. Research on Rotating Savings and Credit Associations (ROSCAs) has demonstrated the potential of digital technologies to improve transparency, accessibility, and financial inclusion within community-based savings activities [6], [7]. Other studies have implemented Flutter and Firebase to develop mobile information systems with cloud-based data management and real-time synchronization for various application domains, demonstrating the effectiveness of these technologies in supporting cross-platform mobile applications [8], [9], [10], [11]. However, these studies generally focus on monetary savings, financial transactions, or application domains outside grocery arisan management. A comparison of previous studies is presented in Table I.

Ref	Focus	Main Contribution	Research Limitation
[16]	ROSCA digitalization	Comprehensive overview of digital ROSCA development	Does not discuss grocery arisan information systems.
[17]	Electronic ROSCA	Demonstrates the application of digital technology in community savings	Focuses on financial transactions rather than grocery package management.
[18]	Mobile arisan application	Improves usability of mobile arisan applications	Does not integrate cloud-based data synchronization.
[19]	Mobile information system	Demonstrates efficient cross-platform mobile application development	Implemented in a different application domain.
[20]	Real-time mobile system	Supports real-time cloud-based data management	Does not address community-based grocery savings management.
[21]	Mobile cloud synchronization	Improves data consistency across mobile devices	Does not focus on grocery arisan operational processes.
This Study	Mobile grocery arisan information system	Designs a mobile information system for participant management, grocery package administration, contribution recording, offline-first cloud-based data synchronization, and cloud-based data synchronization.	—

Tabel I : Research Gap

Based on the comparison presented in Table I, several research gaps can be identified. First, previous studies have primarily focused on monetary ROSCA systems and general mobile financial applications, whereas limited attention

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

has been given to grocery arisan management. Second, existing studies generally do not integrate participant administration, grocery package management, contribution recording, and cloud-based data synchronization within a single mobile information system. Third, although Flutter and Firebase have been widely utilized in mobile application development, their implementation in designing a mobile grocery arisan information system remains limited. Therefore, this study proposes the design of a mobile grocery arisan information system using Flutter and Firebase to support more efficient, transparent, and well-structured management of community-scale grocery arisan activities.

2. RESEARCH METHODS

This chapter describes the research methodology employed in designing the Mobile Grocery Arisan Information System with Firebase-Based Data Synchronization Using Flutter. The research method encompasses the research approach, data collection techniques, system design methodology, development stages, design tools, and research framework adopted in this study.

2.1 Research Approaches

This study employs a descriptive research approach with a qualitative-dominant orientation. The descriptive approach was selected because the primary objective of this research is to produce a comprehensive system design that accurately describes the structure, behavior, and interactions of the proposed mobile grocery arisan information system [12]. According to Sugiyono [13], descriptive research aims to describe systematically, factually, and accurately the facts and characteristics of a particular population or area. This approach is widely used in information system research where the focus is on understanding existing conditions and designing appropriate technological solutions.

The research was conducted at Mila Sembako, a grocery arisan provider located in the research area. The selection of this location was based on the fact that Mila Sembako exemplifies a typical small-scale grocery arisan operation that still relies on manual administrative processes, making it an ideal case study for digital transformation. The descriptive approach enables researchers to systematically document existing workflows, identify operational pain points, and propose a structured system design that addresses the identified challenges [14].

Furthermore, this research adopts a design science research paradigm as a complementary approach. According to Hevner and Chatterjee [15], design science research in information systems involves the creation and evaluation of artifacts designed to solve organizational problems. The artifact produced in this study is a comprehensive system design encompassing UML diagrams, database schema, system architecture, and interface mockups. This dual approach ensures that the research not only describes existing conditions but also produces a tangible, actionable design artifact that can guide future implementation efforts.

2.2 Data Collection

Data collection in this study was conducted through multiple techniques to ensure comprehensive and accurate understanding of the research context. The data collection methods employed include observation, interview, and literature study, forming a triangulated data collection strategy that enhances the validity and reliability of findings [16].

(1) Observation

Direct observation was conducted at the operational premises of Mila Sembako to understand the existing grocery arisan management workflow. The observation focused on documenting the participant registration process, contribution collection procedures, payment recording methods, winner determination mechanisms, and reporting practices. The researcher observed that all administrative tasks were performed manually using paper-based records, with participant data stored in handwritten notebooks and payment tracking maintained through physical ledgers. The observation method provided direct, unfiltered insights into the operational challenges faced by Mila Sembako administrators on a daily basis [14].

(2) Interview

Semi-structured interviews were conducted with key stakeholders involved in the grocery arisan management at Mila Sembako. The primary informant was the arisan administrator, who provided detailed information about operational procedures, challenges encountered, and expectations for a digital system. The interview protocol covered topics including: (a) participant management procedures, (b) contribution collection and recording workflows, (c) challenges in manual administration, (d) communication methods with participants, and

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

(e) desired features for a digital information system. Interview data was recorded, transcribed, and analyzed to identify functional and non-functional requirements for the proposed system.

(3) Literature Study

A comprehensive literature review was conducted to establish the theoretical foundation and technological basis for this research. The literature study covered three main areas: (a) grocery arisan and ROSCA (Rotating Savings and Credit Associations) management systems, (b) cross-platform mobile development using Flutter framework, and (c) cloud-based data synchronization using Firebase. Relevant academic articles from indexed journals, conference proceedings, and authoritative textbooks were reviewed to identify best practices, design patterns, and technological approaches applicable to this study. Particular attention was given to previous studies published in Nuansa Informatika and reputable international journals indexed in Scopus, IEEE, ACM, and Springer databases.

2.3 System Development Stages

The system development in this study follows a structured design process adapted from the Waterfall model with iterative feedback loops. While this research focuses on the design phase rather than full implementation, the development stages provide a roadmap for future implementation work. The development stages are illustrated in Figure 1. (Research Framework) and consist of the following phases:

Phase 1: Requirements Engineering

This initial phase involves comprehensive requirements elicitation through the data collection methods described in Section 2.2. The output of this phase includes a Software Requirements Specification (SRS) document that catalogs all functional and non-functional requirements identified from stakeholder interviews and operational observations [17]. Requirements were categorized using the FURPS+ model (Functionality, Usability, Reliability, Performance, and Supportability), ensuring comprehensive coverage of all system quality attributes.

Phase 2: System Analysis

The analysis phase transforms raw requirements into structured system models. Existing system workflows were analyzed to identify inefficiencies, bottlenecks, and opportunities for improvement. The current manual system at Mila Sembako was modeled using business process diagrams to establish a baseline for comparison. Gap analysis was performed to identify the functional differences between the manual system and the proposed digital solution, ensuring that the designed system addresses all identified shortcomings.

Phase 3: System Design

This phase constitutes the core contribution of this research. System design activities include architectural design, database schema design, UML behavioral modeling, interface design, and synchronization mechanism specification. The system was designed as a three-tier architecture comprising the presentation layer (Flutter UI), business logic layer (state management and synchronization), and data layer (Firebase Cloud Firestore and local Hive cache). Each layer was designed with clear interfaces and separation of concerns to facilitate future maintenance and scalability [18].

Phase 4: Design Validation

The final stage involves validation of the design artifacts through expert review and walkthrough sessions. Design completeness was verified against the requirements specification to ensure all functional and non-functional requirements were addressed. The design documentation was reviewed for consistency, correctness, and clarity to ensure it can serve as an effective blueprint for future implementation

2.4 System Design Tools

The system design in this research utilized a comprehensive set of tools for modeling, prototyping, and documentation. The selection of tools was guided by their suitability for mobile application design, cloud architecture modeling, and academic documentation standards. Table I presents the design tools used in this research along with their purposes and outputs.

Tool	Category	Purpose
Draw.io/Mermaid	UML Modeling	Use case, activity, and sequence diagrams
dbdiagram.io	Database Design	Firestore collection schema design
Figma	UI/UX Design	Interface mockups and wireframes
Visual Studio Code	Code Editor	Architecture documentation

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

Firebase Console	Cloud Design	Database structure and security rules

Tabel II : System Design Tools

The Unified Modeling Language (UML) diagrams were created using Draw.io, a web-based diagramming tool that supports standard UML notation. Draw.io was selected because of its compliance with UML 2.5 specifications, ease of use, and ability to export diagrams in publication-ready formats [19]. For database design, the schema was modeled using a document-oriented approach appropriate for Cloud Firestore's NoSQL structure, with collections representing logical groupings of data entities.

User interface mockups were designed using Figma, a collaborative interface design tool that enables rapid prototyping and design system creation. Figma's component-based design approach facilitated the creation of consistent interface elements across multiple screens, while its preview capabilities enabled visualization of the user experience flow [20]. The UI design followed Material Design 3 guidelines for Android and Cupertino design patterns for iOS, ensuring native platform aesthetics while maintaining cross-platform consistency through Flutter's adaptive widget system.

2.5 Research Framework (Flowchart)

The research framework provides a visual representation of the systematic process employed in this study, from initial problem identification through final design output. Figure. 1 illustrates the research framework adopted in this study, which consists of four sequential phases with feedback loops to ensure design quality and completeness.

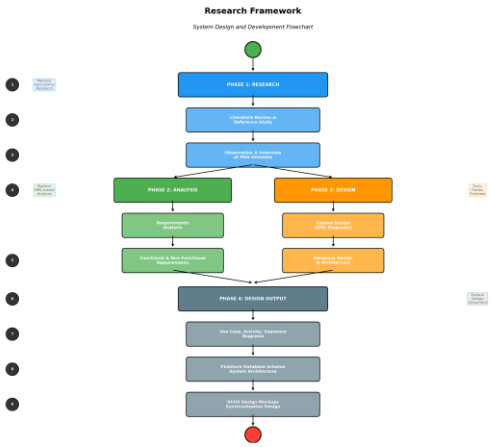


Figure 1. Research Framework

The research framework begins with Phase 1: Research, which encompasses literature review and field observation at Mila Sembako. This phase establishes the theoretical foundation and empirical context for the study. Phase 2: Analysis involves requirements analysis using UML-based techniques, resulting in the identification of functional and non-functional requirements. Phase 3: Design focuses on producing the actual design artifacts using Flutter and Firebase as the designated technology stack. Phase 4: Design Output consolidates all design artifacts into a comprehensive system design document that serves as the primary deliverable of this research [12], [14].

The framework incorporates iterative feedback between phases, allowing design decisions to be refined based on emerging insights. For instance, requirements identified during the analysis phase may necessitate revisiting the literature review to identify suitable design patterns, while architectural constraints discovered during the design phase may require adjustments to the initial requirements specification. This iterative approach aligns with best practices in design science research, where artifact development is inherently iterative and evolutionary [15].

3 RESEARCH RESULTS

This chapter presents the results of the system design process and provides detailed discussion of each design artifact produced. The results encompass the existing system analysis, requirements specification, UML modeling, database design, system architecture, Firebase synchronization mechanism, and user interface design. Each section presents the design decisions, their rationale, and their alignment with the identified requirements.

3.1 Existing System Analysis

The existing grocery arisan management system at Mila Sembako operates entirely through manual processes. Based on direct observation and interviews conducted during the research phase, the current operational workflow was documented and analyzed to identify inefficiencies and areas for improvement. Table II summarizes the comparison between the existing manual system and the proposed digital system.

Aspect	**Manual System**	**Proposed Digital System**

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

Participant Registration	Hand-written forms, manual data entry	Digital registration with form validation
Payment Collection	Door-to-door collection, cash only	Digital payment recording with multiple methods
Payment Tracking	Physical ledgers, prone to errors	Real-time digital tracking with automatic status update
Winner Selection	Manual drawing, limited transparency	Randomized selection with audit trail
Reporting	Manual calculation, hours to prepare	Instant automated reports and analytics
Notifications	Phone calls or WhatsApp messages	Automated push notifications via FCM
Data Storage	Paper records, risk of loss/damage	Cloud-based Firestore with automatic backup
Information Access	Must contact administrator	Self-service access via mobile application

Tabel III : Existing System Analysis

The analysis revealed several critical pain points in the existing system. First, the participant registration process requires administrators to manually record personal information in paper-based forms, which is time-consuming and susceptible to data entry errors. Second, the payment collection process involves administrators visiting each participant's residence to collect weekly contributions, incurring significant time and transportation costs. Third, payment tracking relies on physical ledgers that are difficult to search, prone to calculation errors, and vulnerable to damage or loss [12], [14].

Furthermore, the winner selection process lacks transparency, as participants cannot independently verify the randomness of the selection. Financial reporting requires manual aggregation of payment data, a process that can take several hours depending on the number of participants and the reporting period. Communication with participants is conducted through informal channels such as phone calls or instant messaging applications, which lacks standardization and may result in missed notifications [16].

3.2 Functional Requirements

Functional requirements define the specific behaviors and functions that the system must provide to satisfy stakeholder needs. Based on the analysis of existing system operations and stakeholder interviews, the following functional requirements were identified:

FR-1: User Authentication and Authorization

The system shall provide secure user authentication using email and password credentials managed through Firebase Authentication. The system shall implement role-based access control (RBAC) with two primary roles: Administrator and Participant. Administrators shall have full access to system management functions, while Participants shall be restricted to viewing their own data and performing participant-level actions [17], [21].

FR-2: Participant Management

The system shall enable administrators to register new participants, view participant profiles, update participant information, and deactivate participant accounts. Each participant record shall include personal information, contact details, group membership, and payment history. The system shall enforce data validation rules to ensure data integrity.

FR-3: Arisan Group Management

The system shall support the creation and management of arisan groups, including configuration of contribution amounts, payment schedules, participant rosters, and group status tracking. Administrators shall be able to create multiple groups, each with independent configuration parameters.

FR-4: Package Management

The system shall enable administrators to define grocery packages available for winner distribution. Each package shall include a name, description, list of items, and total price. Administrators shall be able to create, modify, and deactivate packages as needed.

FR-5: Payment Recording and Tracking

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

The system shall support recording of participant contributions, including payment amount, payment date, payment method (bank transfer or cash), and confirmation status. Administrators shall be able to view payment status for all participants, filter by payment status, and generate payment reports. Participants shall be able to view their own payment history and current payment status [14], [16].

FR-6: Winner Selection

The system shall provide a fair and transparent winner selection mechanism using a randomized algorithm with verifiable random seed. The selection process shall consider eligibility criteria (e.g., participants with outstanding payments may be excluded) and maintain a complete audit log of all selections for transparency purposes.

FR-7: Notification System

The system shall send automated push notifications to participants for payment reminders, winner announcements, and general announcements. Notifications shall be delivered through Firebase Cloud Messaging (FCM) and stored in-app for reference. Administrators shall be able to compose and send targeted notifications to specific participants or broadcast messages to all group members [17].

FR-8: Reporting and Analytics

The system shall provide administrators with comprehensive reporting capabilities, including payment collection summaries, participant participation rates, group performance metrics, and financial overview reports. Reports shall be generated in real-time based on Firestore data queries.

3.3 Non-Functional Rquirements

Non-functional requirements specify the quality attributes and constraints that the system must satisfy. These requirements are essential for ensuring the system's usability, reliability, performance, and maintainability. The following non-functional requirements were identified based on stakeholder expectations and industry best practices for mobile information systems [18], [22]:

NFR-1: Performance

The system shall respond to user interactions within 2 seconds under normal network conditions. Database queries shall be optimized to return results within 1 second. The application shall implement efficient caching strategies to minimize redundant data fetching from the cloud database.

NFR-2: Availability and Reliability

The system shall implement an offline-first architecture that allows core functionality to operate without continuous internet connectivity. Data synchronization shall occur automatically when network connectivity is restored, with conflict resolution mechanisms to handle concurrent modifications. The system shall achieve 99.9% uptime through Firebase's managed infrastructure [19], [20].

NFR-3: Security

The system shall encrypt all data in transit using TLS 1.3 and implement Firebase Security Rules for data access control. Passwords shall be hashed using industry-standard algorithms. The system shall implement Role-Based Access Control (RBAC) to ensure data isolation between participants and prevent unauthorized access to administrative functions. Personal data shall be handled in compliance with data protection regulations.

NFR-4: Usability

The user interface shall follow Material Design 3 guidelines for Android and Cupertino design patterns for iOS. The interface shall be intuitive and require minimal training for users with basic smartphone literacy. The application shall provide clear feedback for user actions, informative error messages, and accessible help resources. The target System Usability Scale (SUS) score shall be at least 70 (above average) [14].

NFR-5: Scalability

The system architecture shall support horizontal scaling to accommodate growth in participant numbers and arisan groups. Cloud Firestore's serverless architecture naturally supports scaling without manual intervention. The local database caching mechanism shall be designed to handle increasing data volumes efficiently.

NFR-6: Maintainability

The system shall be designed with clean architecture principles, ensuring separation of concerns between the presentation layer, business logic layer, and data layer. Code shall be modular and well-documented to facilitate future maintenance and feature enhancements. The MVVM architectural pattern shall be employed to ensure testability and

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

maintainability of the Flutter application [21], [23].

3.4 Use Case Diagram

The Use Case Diagram illustrates the functional boundaries of the system and the interactions between actors and system functions. As shown in Figure. 2, the system identifies three primary actors: Admin (Administrator), Participant, and Firebase Cloud Service as a supporting actor.

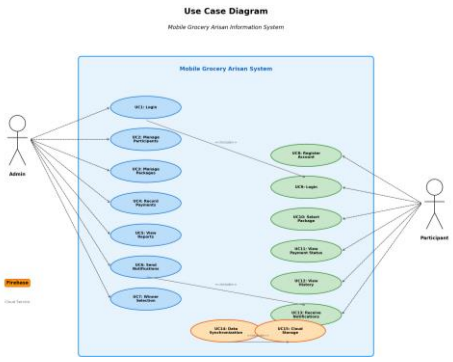


Figure 2. Use Case Diagram

The Admin actor has access to seven primary use cases: (UC1) Login, which includes authentication and session management; (UC2) Manage Participants, encompassing CRUD (Create, Read, Update, Delete) operations on participant records; (UC3) Manage Packages, for configuring available grocery packages; (UC4) Record Payments, allowing administrators to record and confirm participant contributions; (UC5) View Reports, providing analytical dashboards and summary reports; (UC6) Send Notifications, enabling targeted communication with participants; and (UC7) Winner Selection, implementing the randomized drawing mechanism.

The Participant actor interacts with six use cases: (UC8) Register Account for initial system enrollment; (UC9) Login for authenticated access; (UC10) Select Package for choosing preferred grocery packages; (UC11) View Payment Status for self-monitoring of contribution records; (UC12) View History for accessing historical transaction data; and (UC13) Receive Notifications for receiving system-generated alerts and announcements.

The system also includes two backend use cases related to Firebase integration: (UC14) Data Synchronization, which manages bidirectional data flow between the mobile application and Cloud Firestore; and (UC15) Cloud Storage, providing persistent storage for application data with automatic backup capabilities. The include relationships indicate that UC1 (Admin Login) and UC9 (Participant Login) share common authentication logic, while UC6 (Send Notifications) depends on UC13 (Receive Notifications) for message delivery.

3.5 Activity Diagram

The Activity Diagram models the dynamic workflow of the contribution payment process, which is a core business process in the grocery arisan system. Figure. 3 presents the activity diagram with three swimlanes representing the Participant, System, and Admin actors, illustrating the collaborative nature of the payment validation workflow [24], [25].

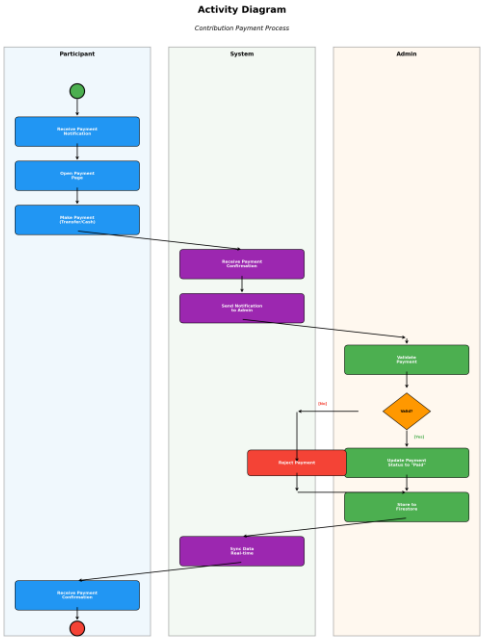


Figure 3. Activity Diagram

The payment process begins when the Participant receives a payment notification from the system, triggered either by the scheduled payment date or by manual initiation from the administrator. The participant opens the payment page within the mobile application, which displays the payment amount, due date, and available payment methods. The participant then proceeds to make the payment using either bank transfer or cash payment method. For bank transfers, the participant is presented with the destination account details and requested to upload proof of transfer. For cash payments, the participant records the intent to pay, which is then processed by the administrator [12].

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

Upon payment submission, the System receives the payment confirmation and forwards a notification to the Administrator for validation. The Administrator reviews the payment evidence and makes a validation decision represented by the decision diamond in the diagram. If the payment is valid (sufficient amount, correct recipient, and authentic transfer proof), the Administrator updates the payment status to "Paid" and the payment record is stored in Cloud Firestore with an automatic timestamp. If the payment is invalid, the Administrator rejects the payment and the system notifies the Participant to resubmit. Once validated, the System synchronizes the payment data in real-time, making the updated status visible to the Participant, who receives confirmation of successful payment processing [20].

3.6 Sequence Diagram

The Sequence Diagram in Figure. 4 illustrates the temporal ordering of object interactions during the user login authentication process. This diagram is particularly important because it reveals the integration pattern between the Flutter application and Firebase services, which is central to the system's data synchronization architecture [17], [21].

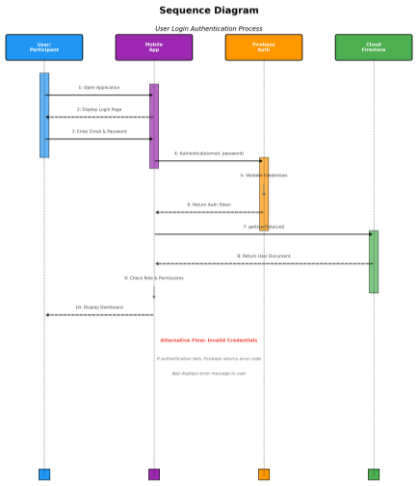


Figure 4. Sequence Diagram

The authentication sequence begins when the Participant opens the mobile application (Step 1). The Mobile App initializes and displays the login page to the user (Step 2). The Participant enters their registered email address and password into the authentication form (Step 3). The Mobile App then calls the Firebase Authentication API with the provided credentials (Step 4). Firebase Auth validates the credentials against its user database and, upon successful validation, returns an authentication token to the Mobile App (Steps 5-6).

Upon receiving the authentication token, the Mobile App requests the user's profile data from Cloud Firestore using the authenticated user's unique identifier (UID) as the document key (Step 7). Cloud Firestore returns the user document, which contains role information (admin or participant), group membership, and profile details (Step 8). The Mobile App then checks the user's role and permissions to determine the appropriate dashboard to display (Step 9). Based on the role, either the Admin Dashboard or Participant Dashboard is presented to the user (Step 10). An alternative flow exists for invalid credentials, where Firebase returns an error code and the app displays an appropriate error message prompting the user to retry [18].

3.7 Firestore Database Design

The database was designed using Cloud Firestore, a flexible, scalable NoSQL cloud database provided by Firebase. Firestore was selected over traditional relational databases because of its native support for real-time data synchronization, offline data persistence, and horizontal scalability. Firestore's document-oriented data model naturally aligns with the hierarchical structure of arisan data, where groups contain participants, participants have payments, and winners are drawn from groups [18].

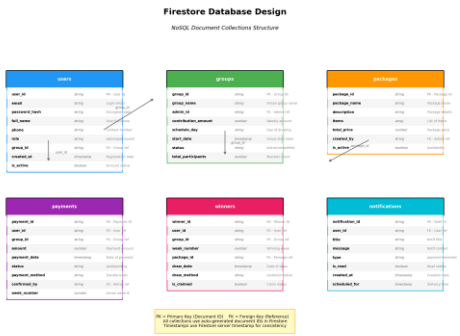


Figure 5. Database Design

Figure. 5 illustrates the six primary collections in the database schema: users, groups, packages, payments, winners, and notifications. The users collection stores authentication and profile information for both administrators and participants, with the role field differentiating access levels. The groups collection defines arisan groups with their configuration parameters including contribution amount, schedule, and membership roster. The packages collection catalogs available grocery packages with item lists and pricing.

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

The payments collection records all contribution transactions, tracking payment amount, date, method, status, and confirmation details. The winners collection maintains a history of winner selections with draw metadata including week number, draw method (random or rotation), and claim status. The notifications collection stores system-generated messages with delivery tracking and read status. Foreign key relationships are implemented using document references and string identifiers, enabling efficient queries while maintaining data integrity [19].

Collection	Primary Purpose	Key Fields	Relationships
users	User accounts and profiles	user_id, email, role, group_id	References groups
groups	Arisan group configuration	group_id, admin_id, contribution_amount	Referenced by users, payments
packages	Grocery package definitions	package_id, items[], total_price	Referenced by winners
payments	Contribution transactions	payment_id, user_id, status, week_number	References users, groups
winners	Winner selection records	winner_id, user_id, package_id, week_number	References users, groups, packages
notifications	System messages	notification_id, user_id, type, is_read	References users

Tabel IV : Firestore Database Design

3.8 System Architecture

The system architecture follows a layered client-server model with cloud backend services. Figure. 6 presents the system architecture diagram, which comprises four main layers: the Client Layer, Connection/Network Layer, Cloud Layer, and External Services Layer. This architecture was designed to leverage the strengths of Flutter for cross-platform mobile development and Firebase for serverless backend infrastructure [21], [23].

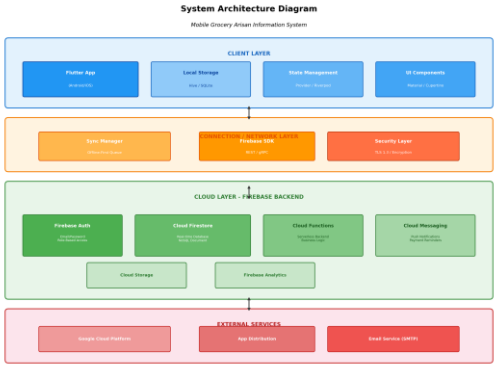


Figure 6. System Architecture Design

The Client Layer encompasses the Flutter mobile application running on Android and iOS devices. This layer includes four components: the Flutter App runtime, Local Storage (Hive/SQLite), State Management (Provider/Riverpod), and UI Components (Material/Cupertino widgets). The local storage component is critical for the offline-first architecture, caching data locally when the device is offline and synchronizing with the cloud when connectivity is restored. The state management component ensures reactive UI updates when Firestore data changes, creating a responsive user experience [20].

The Connection Layer mediates communication between the client and cloud services. The Sync Manager component implements the offline-first queue system, tracking pending operations and managing their execution order. The Firebase SDK provides client libraries for authentication, database access, and messaging. The Security Layer enforces TLS 1.3 encryption for all network communications and validates server certificates to prevent man-in-the-middle attacks.

The Cloud Layer represents the Firebase backend services that power the application. Firebase Authentication manages user identity and access control, supporting email/password authentication with optional multi-factor authentication. Cloud Firestore serves as the primary database, providing real-time data synchronization, offline persistence, and automatic scaling. Cloud Functions enable serverless execution of business logic such as payment validation and winner selection algorithms. Firebase Cloud Messaging (FCM) delivers push notifications to devices. Cloud Storage handles file uploads (payment proofs, profile images), while Firebase Analytics provides usage insights [17], [18].

3.9 Firebase Syncronization Mechanism

p-ISSN :1858-3911 **,** ***e-ISSN : 2614-5405***
https://journal.fkom.uniku.ac.id/ilkom

The Firebase synchronization mechanism is a critical component of the system architecture, enabling real-time data consistency between the mobile application and the cloud database. Figure. 7 illustrates the synchronization architecture, which implements an offline-first pattern to ensure application functionality under varying network conditions. This design is particularly important for the target deployment context, where internet connectivity may be intermittent or unreliable [19], [20].

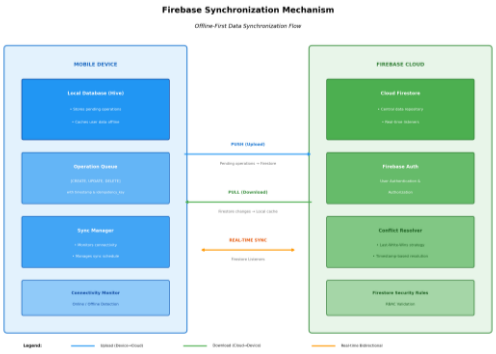


Figure 7. Firebase Sync Mechanism

The synchronization mechanism operates through three distinct modes. First, the PUSH (Upload) mode handles data transmission from the device to the cloud. When the user performs a write operation (create, update, or delete), the operation is immediately recorded in the local database (Hive) and queued in the Operation Queue with metadata including timestamp, operation type, and an idempotency key to prevent duplicate processing. The Sync Manager monitors network connectivity and, when a connection is available, dequeues operations and sends them to Cloud Firestore in chronological order.

Second, the PULL (Download) mode handles data reception from the cloud. Firestore's real-time listeners detect changes in the cloud database and push updates to connected devices. When the mobile application receives an update, it applies the changes to the local cache, ensuring that the device always has the latest data. This mechanism enables multi-device synchronization, allowing users to access consistent data across multiple devices [18].

Third, the Conflict Resolution mechanism handles scenarios where the same data is modified on multiple devices while offline. The system implements a Last-Write-Wins (LWW) strategy using server timestamps as the conflict resolution criterion. Each document update includes a server-generated timestamp, and when conflicts are detected, the update with the most recent timestamp takes precedence. While this approach may result in data loss in certain scenarios, it is appropriate for the grocery arisan domain where the latest payment status or participant information is generally the authoritative version. For critical operations such as winner selection, the system implements additional safeguards including atomic transactions and audit logging to prevent race conditions [20].

3.10 UI/UX Design

The user interface design follows a mobile-first approach, prioritizing usability and accessibility for users with varying levels of digital literacy. The UI design process began with wireframing to establish information hierarchy and navigation flow, followed by high-fidelity mockups that incorporate the visual design system. Figure. 8-10 presents the UI/UX design mockups for the eight primary screens of the mobile application [14].

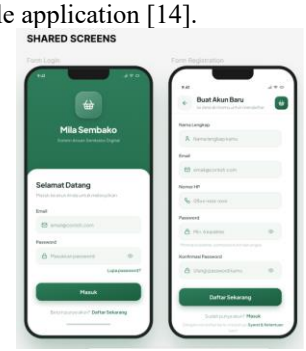


Figure 8. Login & Register Screen

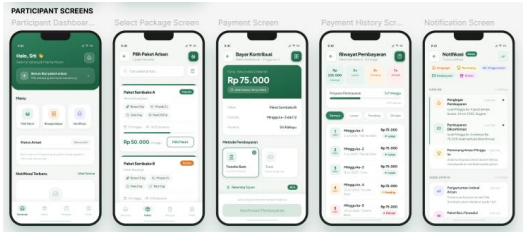
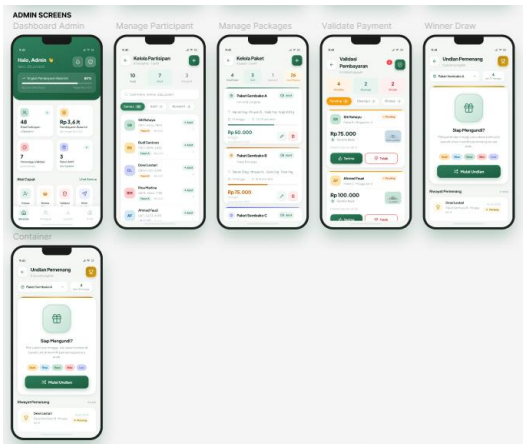


Figure 9. User Screen



p-ISSN :1858-3911* , *e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

Figure 10. Admin Screen

The user interface design follows a mobile-first approach to ensure usability, accessibility, and a consistent user experience across various smartphone devices. The design process began with wireframing to define the application structure and navigation flow, followed by high-fidelity mockups that implement a unified visual design system. As shown in Figure. 8, the application interface is organized into shared screens, participant screens, and administrator screens.

The Shared Screens consist of the Login and Registration interfaces. The Login screen provides a simple authentication interface with email and password fields, accompanied by the application logo and a primary login button to facilitate secure user authentication. The Registration screen enables new users to create an account by entering their personal information, including full name, email address, phone number, password, and password confirmation, ensuring a straightforward onboarding process.

The Participant Screens include the Participant Dashboard, Select Package, Payment, Payment History, and Notifications screens. The Participant Dashboard presents personalized information such as the current contribution status, available menu shortcuts, account summary, and recent notifications. The Select Package screen allows users to browse available grocery packages and select a preferred package for the current arisan period. The Payment screen displays the contribution amount, selected package details, and available payment methods before confirming the transaction. The Payment History screen provides a chronological record of completed and pending payments, enabling users to monitor their contribution history. Meanwhile, the Notifications screen categorizes system notifications, reminders, payment confirmations, and announcements to ensure that participants remain informed about important activities.

The Administrator Screens include the Admin Dashboard, Manage Participants, Manage Packages, Payment Validation, and Winner Draw screens. The Admin Dashboard summarizes key operational information through statistical cards, including participant statistics, payment summaries, and administrative shortcuts. The Manage Participants screen enables administrators to search, view, and manage participant information efficiently. The Manage Packages screen provides facilities for creating, updating, and organizing grocery package data. The Payment Validation screen assists administrators in verifying participant payments before they are recorded into the system. Finally, the Winner Draw screen supports the transparent selection of arisan winners while displaying the draw status and previous winner history.

The interface adopts a consistent visual design based on a green color palette that represents growth, trust, and financial sustainability, complemented by neutral white backgrounds to enhance readability. Secondary colors, including orange, red, and blue, are used selectively to indicate warnings, errors, and informational states, respectively. The application follows Material Design principles by employing consistent typography, rounded card components, adequate white space, and intuitive icons. Interactive elements maintain sufficient touch target sizes to improve usability, while the bottom navigation bar provides persistent access to the application's primary sections, including Home, History, and Profile, enabling efficient navigation throughout the system.

4 DISCUSSIONS

4.1 Alignment Between Design Outcomes and Research Objectives

The objective of this study was to design a mobile-based grocery arisan information system capable of supporting the digitalization of community-scale grocery savings management. The design outcomes presented in Chapter 3 indicate that the problems identified in the existing system analysis (Table II) have been systematically addressed through concrete design artifacts. The manual and error-prone contribution recording process was addressed through the design of the *payments* collection in Firestore (Section 3.7), which stores payment status, method, and timestamp in a structured manner, as well as through the activity diagram of the payment validation workflow (Section 3.5), which formally defines the administrator's role in verifying payment evidence.

Similarly, the lack of transparency in the winner selection process was addressed through FR-6, which specifies a randomized selection mechanism supported by an audit trail stored in the *winners* collection. This demonstrates a clear traceability between the problems identified in the existing system analysis, the requirements formulated, and the resulting UML and database design artifacts. This traceability is worth emphasizing because it serves as an indicator of design validity, even though empirical testing of

p-ISSN :1858-3911* , *e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

a fully functioning system has not yet been conducted at this stage of the research.

4.2 Justification of Key Design Decisions

Several design decisions presented in Chapter 3 warrant further discussion regarding their rationale, particularly within the context of the community-based grocery arisan domain under study.

Selection of Firestore over a relational database. Arisan data is inherently hierarchical groups contain participants, participants have payment histories, and winners are drawn from specific groups. Firestore's document-oriented data model naturally accommodates this structure without requiring the complex joins typically associated with relational databases. In addition, Firestore's built-in support for real-time listeners and offline persistence makes it more suitable than a conventional relational database, which generally requires additional middleware to achieve comparable synchronization capabilities.

Adoption of an offline-first architecture. The decision to implement an offline-first architecture (NFR-2) was made in direct response to the real-world usage context observed at Mila Sembako, where internet connectivity among participants may not always be stable. By storing data locally through Hive prior to synchronization with Firestore, the system is designed so that core functions—such as viewing payment status—remain accessible even without an active connection. This offline-first approach represents one of the central design contributions of this study, distinguishing it from prior grocery arisan applications that assume continuous online access and do not address connectivity limitations common in community-scale deployments [8].

Last-Write-Wins (LWW) conflict resolution strategy. The adoption of LWW as the conflict resolution mechanism (Section 3.9) reflects a deliberate trade-off between implementation simplicity and the requirements of the domain. For data such as payment status or participant profiles, the most recently updated version is generally the most valid one, making LWW an adequate strategy. However, for operations that are sensitive to race conditions, such as winner selection, the design intentionally introduces additional safeguards through atomic transactions and separate audit logging. This indicates that the conflict resolution strategy was not applied uniformly, but rather tailored to the criticality of each business process—an important nuance that supports the reliability of the offline-first synchronization mechanism as a whole.

Adoption of the MVVM pattern. The selection of MVVM (NFR-6) aligns with the need to separate concerns across the presentation, business logic, and data layers, which is particularly relevant given that the system encompasses multiple distinct business processes (participant management, package management, payment recording, and notifications) that will need to be developed and maintained independently during future implementation.

4.3 Positioning of This Study Relative to Prior Research

The research gap comparison in Table I indicates that prior studies have addressed the components of this design only partially. Studies on ROSCA digitalization [6] and electronic ROSCA [7] contribute valuable theoretical foundations for understanding the digital transformation of rotating savings mechanisms, but do not address grocery package management or cloud-based offline-first synchronization architecture, both of which are central to this study. The mobile arisan application study [8] addresses the arisan context directly but does not integrate cloud-based data synchronization, leaving cross-device data consistency unresolved—a limitation that the offline-first design proposed in this study directly addresses.

Meanwhile, studies that focus on real-time and cloud-based synchronization using Flutter and Firebase [9], [10], [11] provide a strong technical foundation for the architecture designed in this study (Sections 3.8 and 3.9), yet were implemented in different application domains and do not address the operational processes specific to grocery arisan, such as weekly contribution recording or grocery package distribution.

The contribution of this study therefore lies in integrating two aspects that have previously been addressed separately in the literature: the community-based grocery arisan domain on one hand, and offline-first, Firebase-based cloud synchronization architecture on the other. The resulting design—encompassing participant management, package management, contribution recording, winner selection, notification services, and offline-first cloud-based data synchronization in a single integrated system positions this study as a direct response to the research gap identified in Table I.

4.4 Practical Implications for Mila Sembako

If implemented, this design is expected to yield several practical benefits. First, administrative workload is anticipated to decrease substantially, as the door-to-door payment collection process can be replaced by

p-ISSN :1858-3911 , e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

remote-verifiable digital recording (Section 3.5). Second, the transparency of the winner selection process is expected to improve, given the availability of an auditable selection history, in contrast to the manual draw process previously used, which could not be independently verified by participants. Third, the offline-first design offers a practical advantage specifically for participants in areas with limited connectivity, a condition commonly observed in small-to-medium-scale community arisan groups such as Mila Sembako.

Nevertheless, these implications remain projections based on the proposed design rather than measured outcomes, and therefore require further validation during the implementation and testing phases of future research.

4.5 Research Limitations

This study has several limitations that should be explicitly acknowledged.

1. First, the research is limited to the design stage and does not include implementation or empirical system testing. As a result, claims regarding the fulfillment of non-functional requirements—such as NFR-1 (response time under 2 seconds) or NFR-4 (a target SUS score of at least 70)—remain design targets and have not yet been empirically validated.
2. Second, although the Last-Write-Wins strategy is appropriate for most use cases, it carries an inherent risk of data loss in rare concurrent-update scenarios, particularly for entities other than payments and winner selection, which have not been assigned additional safeguards in the current design.
3. Third, the evaluation underlying this design is based on observation and interviews conducted at a single research site, Mila Sembako (Chapter 2). Generalization of this design to other grocery arisan communities with different operational characteristics should therefore be approached with caution.
4. Fourth, the system's dependency on the Firebase ecosystem, while offering development efficiency and scalability (NFR-5), also introduces a degree of vendor lock-in that was not extensively examined in this study and may warrant consideration in future, long-term system development.

5 CONCLUSION

This study has presented the design of a mobile-based grocery arisan information system intended to support the digitalization of community-scale grocery savings management at Mila Sembako. Through a descriptive research approach combining observation, interviews, and literature study, the existing manual system was analyzed to identify key operational challenges, including inefficient contribution recording, limited payment tracking accuracy, lack of transparency in winner selection, and time-consuming manual reporting.

Based on these findings, a complete system design was produced, encompassing functional and non-functional requirements specification, use case modeling, activity and sequence diagrams, Firestore database design, system architecture, an offline-first Firebase synchronization mechanism, and user interface mockups. The design demonstrates that Flutter and Firebase can be effectively combined to support participant management, grocery package management, contribution recording, winner selection, notification services, and cloud-based data synchronization within a single integrated mobile information system.

The discussion in Chapter 4 confirmed that the resulting design directly addresses the problems identified in the existing system analysis and is traceable to the functional and non-functional requirements formulated for this study. In particular, the adoption of an offline-first architecture with Last-Write-Wins conflict resolution was shown to be a deliberate response to the connectivity conditions commonly encountered in community-scale arisan operations, distinguishing this study from prior research that either focuses on monetary ROSCA systems, lacks cloud-based synchronization, or addresses synchronization in application domains unrelated to grocery arisan management.

Overall, this study demonstrates the feasibility of designing a structured, transparent, and reliable mobile information system for grocery arisan management using Flutter and Firebase, and provides a design framework that can serve as a reference for the digital transformation of community-based grocery savings schemes, both at Mila Sembako and at similar community organizations.

6 SUGGESTIONS

Based on the design outcomes and limitations discussed in Chapter 4, several suggestions are proposed for future research and development.

1. First, the design produced in this study should be implemented as a functioning mobile application using Flutter and Firebase, followed by functional testing to

p-ISSN :1858-3911* , *e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

verify that the system behaves in accordance with the specified functional requirements (FR-1 to FR-8).

2. Second, empirical testing should be conducted to validate the non-functional requirements defined in this study, particularly system response time (NFR-1) and usability, through System Usability Scale (SUS) measurement involving actual administrators and participants at Mila Sembako (NFR-4).
3. Third, further investigation is recommended regarding the reliability of the Last-Write-Wins conflict resolution strategy under real concurrent-usage conditions, including an assessment of whether additional safeguards, beyond those applied to payment and winner selection processes, are necessary for other data entities within the system.
4. Fourth, future development may consider conducting field trials in multiple grocery arisan communities with varying operational characteristics and connectivity conditions, in order to evaluate the generalizability of the proposed offline-first design beyond the single case of Mila Sembako.
5. Fifth, subsequent studies may also explore architectural alternatives or mitigation strategies for the vendor lock-in risk associated with reliance on the Firebase ecosystem, including the feasibility of hybrid or backend-agnostic synchronization approaches for long-term system sustainability.

Finally, once implemented, the system may be further developed to include additional features such as more advanced financial analytics, integration with digital payment gateways, and multi-language support, in order to broaden its applicability to a wider range of community-based grocery savings schemes.

REFERENCES

[1] A. M. B. Syarbaini, "TRADISI ARISAN UANG," pp. 115–133.

[2] Z. Abidin, "Tinjauan Hukum Islam Tentang Sistem Arisan Sembako," vol. 9, no. 2, pp. 10–19, 2023.

[3] P. Keuangan, D. Desa, R. Sulistyowati, and R. Nataliawati, "Analisis Akuntabilitas , Transparansi , dan Partisipasi," vol. 6, no. April, pp. 1798–1811, 2022.

[4] D. K. Anggraeni *et al.*, "Peningkatan Efisiensi Administrasi Keuangan Melalui Sistem Penagihan Otomatis : Studi Kasus Pada PT Java Energy Semesta," vol. 05, no. 03, pp. 517–522, 2025.

[5] I. Akbar, Z. Niqotaini, and A. R. Fauzi, "Analisis Dan Perancangan Sistem Penjualan Pada Toko XYZ Berbasis Web Dan Mobile Menggunakan UML," vol. 17, no. July, pp. 71–82, 2023.

[6] A. F. Zambrano, L. F. Giraldo, M. T. Perdomo, I. D. Hernández, and J. M. Godoy, "Rotating savings and credit associations: A scoping review," *World Dev. Sustain.*, vol. 3, no. April, p. 100081, 2023, doi: 10.1016/j.wds.2023.100081.

[7] D. Ahn, W. Kang, K.-K. Kim, and H. Shin, "Analysis and design of microfinance services: A case of ROSCA," *Eng. Econ.*, vol. 62, no. 3, pp. 197–230, Jul. 2017, doi: 10.1080/0013791X.2016.1236305.

[8] M. A. Adli and D. P. Lestari, "Designing an arisan mobile application for novice users using user-centered design approach," in *2017 International Conference on Advanced Informatics, Concepts, Theory, and Applications (ICAICTA)*, IEEE, Aug. 2017, pp. 1–6. doi: 10.1109/ICAICTA.2017.8090956.

[9] I. K. Wairooy, I. Dillwyn, K. P. Yonathan, and A. Lay, "Development of Mobile QR Warehouse Management Apllication Based on Flutter and Firebase," *Eng. Math. Comput. Sci. J.*, vol. 6, no. 1, pp. 39–44, 2024, doi: 10.21512/emacsjournal.v6i1.10921.

[10] Y. For, E. C. Chang, and Y. Chang, "a G Aming S Ystem for S Houlder," vol. 22, no. 2, p. 6698, 2015, doi: 10.5121/csit.2026.160418.

[11] R. K. Lomotey, J. Nilson, K. Mulder, K. Wittmeier, C. Schachter, and R. Deters, "Mobile Medical Data Synchronization on Cloud-Powered Middleware Platform," *IEEE Trans. Serv. Comput.*, vol. 9, no. 5, pp. 757–770, Sep. 2016, doi: 10.1109/TSC.2016.2555313.

[12] A. R. Kamila, G. H. Derhass, D. A. Rabbani, and F. S. Lee, "Aplikasi Absensi Berbasis Android Pada Sekolah Boarding Sebagai Transformasi Digital Bidang Pendidikan," vol. 18, pp. 26–34, 2024.

[13] P. D. Sugiyono, *METODE PENELITIAN KUANTITATIF, KUALITATIF DAN R &. D*. Bandung: Alfabeta, 2021.

[14] M. L. Haryanti, F. Fraderic, S. Hartono, and J. Salim, "Cinema e-Ticket Application Design and Usability

p-ISSN :1858-3911* , *e-ISSN : 2614-5405
https://journal.fkom.uniku.ac.id/ilkom

Evaluation Using SUS," vol. 19, pp. 14–22, 2025.

[15] A. Hevner and S. Chatterjee, *Design Research in Information Systems*, vol. 22. in Integrated Series in Information Systems, vol. 22. Boston, MA: Springer US, 2010. doi: 10.1007/978-1-4419-5653-8.

[16] S. Rosaulina and C. M. Sihotang, "Rancang Bangun Sistem Inventory (Studi Kasus : UD . Asia Pratama Pekanbaru)," vol. 18, pp. 35–40, 2024.

[17] K. U. Singh, N. Varshney, P. Gupta, G. Kumar, T. Singh, and S. R. Dogiwal, "Mobile Application Control with Firebase Cloud Messaging," 2024, pp. 527–535. doi: 10.1007/978-981-97-3588-4_42.

[18] M. Ohyver, J. V. Moniaga, I. Sungkawa, B. E. Subagyo, and I. A. Chandra, "The comparison firebase realtime database and MySQL database performance using wilcoxon signed-rank test," *Procedia Comput. Sci.*, vol. 157, pp. 396–405, 2019, doi: 10.1016/j.procs.2019.08.231.

[19] M. Kleppmann, A. Wiggins, P. Van Hardenberg, and M. Mcgranaghan, "Local-First Software : You Own Your Data , in spite of the Cloud".

[20] S. H. Pothineni, "Offline-First Mobile Architecture : Enhancing Usability and Resilience in Mobile Systems," vol. 07, no. 1, 2024.

[21] S. Sharma, S. Khare, V. Unival, and S. Verma, "Hybrid Development in Flutter and its Widgits," in *2022 International Conference on Cyber Resilience (ICCR)*, IEEE, Oct. 2022, pp. 1–4. doi: 10.1109/ICCR56254.2022.9995973.

[22] R. S. Pressman, *Software Engineering*.

[23] D. Zou and M. Y. Darus, "A Comparative Analysis of Cross-Platform Mobile Development Frameworks," in *2024 IEEE 6th Symposium on Computers & Informatics (ISCI)*, IEEE, Aug. 2024, pp. 84–90. doi: 10.1109/ISCI62787.2024.10667693.

[24] M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, Third Edit. Boston, MA: Addison-Wesley Professional, 2004.

[25] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modelling Language User Guide*. 1999.