Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

PVApp: A Cloud-Backed Mobile Framework for Real-Time Consumer Product Verification in Ghana

Ibrahim Tanko

tibrahim@stu.csuc.edu.gh

Christian Service University

Nana Kofi Ahoi Appiah-Badu

Christian Service University

Siddique Abubakr Muntaka

University of Cincinnati

Ajugu Simon

Christian Service University

Research Article

Keywords: Cloud computing, mobile cloud computing, Firebase, Android, real-time database, QR code, product verification, regulatory technology, consumer safety, Ghana

Posted Date: August 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10796490/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Abstract

Counterfeit, substandard, and expired products pose significant consumer-safety challenges in Ghana. Existing verification mechanisms commonly rely on physical inspection, public alerts, manufacturer-supported authentication services, or web-based information channels. These approaches do not necessarily provide consumers with a unified, mobile-accessible mechanism to retrieve product information, verify product identifiers, distinguish product approval from expiry status, and report suspected products in real time. Cloud computing offers an opportunity to address this limitation by enabling centralised data management, real-time synchronisation, scalable service delivery, and access across heterogeneous client applications. This study presents PVApp, a cloud-backed mobile framework for consumer product verification. The system consists of a native Android application, a Firebase Realtime Database cloud data layer, and a web-based administrative panel. The Android client was implemented in Java using Android SDK API 28 or later. Google ML Kit Code Scanner was integrated for QR-code identification, while Firebase Realtime Database provides cloud-hosted product and incident-report storage and real-time synchronisation. The system implements six functional modules: user onboarding, product search, QR-code scanning, three-state product verification, consumer reporting, and an administrative management interface. An iterative Agile-inspired development process was used, followed by black-box functional testing on physical Android devices and Android emulators. Twenty predefined functional test cases were executed across onboarding, product search, product verification, QR-code scanning, consumer reporting, and administrative operations. All 20 test cases passed. The cloud architecture successfully synchronised product records and consumer reports between the mobile application and administrative panel. The three-state verification engine distinguished among approved and valid products, approved but expired products, and products that were neither approved nor valid. Two minor observations were identified during testing: delayed rendering of product records under slow network conditions and limitations associated with storing product images as application resources. PVApp demonstrates the feasibility of combining mobile computing with a managed cloud backend to support real-time consumer product verification. The architecture reduces the need for a dedicated backend server while providing centralised data management and synchronisation between mobile and web clients. The framework provides a practical foundation for cloud-enabled regulatory information systems in resource-constrained environments. Further work is required in authentication, cloud-based image storage, offline operation, direct regulatory database integration, notifications, and large-scale performance evaluation.

1. Background

CNF: Please have a look at courier new font provided for text in article.

Product safety is an important component of public health and consumer protection. In Ghana, the Food and Drugs Authority (FDA) is responsible for regulating the safety, quality, and efficacy of products including pharmaceuticals, food products, cosmetics, medical devices, and household chemicals [1].

Despite the existence of regulatory mechanisms, counterfeit, substandard, and expired products continue to pose risks to consumers, particularly in the pharmaceutical, food, and cosmetics markets.

The consequences of substandard and falsified products can be substantial. The World Health Organisation has documented the public-health consequences associated with substandard and falsified medical products [2]. In addition to direct health risks, counterfeit products can undermine consumer confidence, affect legitimate manufacturers, and complicate regulatory enforcement [3, 4].

Traditional product-safety enforcement depends substantially on physical inspections, regulatory interventions, manufacturer controls, and public communication. These mechanisms remain important but do not necessarily provide individual consumers with immediate access to product-registration information at the point of purchase. The availability of mobile devices and cloud computing creates an opportunity to provide such access through a centralised, continuously synchronised information service.

Ghana has experienced significant growth in internet and mobile connectivity. A 2024 KPMG report cited internet penetration of approximately 70% and mobile connectivity exceeding 100% in Ghana [5]. Android is also widely used among smartphone users in Ghana and across Africa, making an Android client a practical mechanism for consumer-facing access to cloud-based services.

Cloud computing is particularly relevant to this problem because the product registry and consumer reports need to be accessible from multiple clients and updated without requiring redistribution of the mobile application. A cloud-hosted database allows a regulator to modify product information through an administrative interface while making those changes available to mobile users in real time. Such an architecture can also reduce the infrastructure requirements associated with maintaining a dedicated application server.

This paper presents PVApp, a cloud-backed mobile framework developed to explore this approach. PVApp provides a native Android client for consumers and a web-based administrative client for product-registry management. Firebase Realtime Database acts as the central cloud data layer. The mobile application supports product search, QR-code scanning, product-status verification, and consumer reporting.

This work makes five key contributions to cloud-backed product verification and monitoring systems.

Firstly, it introduces a cloud-backed architecture designed to seamlessly synchronise product information between Android mobile devices and web management clients. The system implements a three-state verification mechanism that explicitly separates a product's official approval status from its expiration state to enhance verification precision.

Additionally, the study integrates QR-code product identification with direct cloud-based search and query capabilities, enabling real-time verification. It further bridges consumer oversight and

administrative workflows by embedding consumer-generated incident reports into the unified cloud data environment.

Finally, the practical viability of this integrated design is validated through an empirical functional evaluation comprising 20 distinct test cases.

The remainder of the paper describes related systems, the proposed cloud architecture, implementation, evaluation, discussion, limitations, and future development.

2. Related Work

2.1 Consumer product verification systems

Several product verification systems have been developed in Africa and other regions. Their architectures and verification mechanisms vary according to their intended users and regulatory environments.

NAFDAC's mNafdac application in Nigeria enables consumers to verify medicine registration using product information such as product name and registration number [10]. While such a service provides an important digital verification mechanism, it is designed for the Nigerian regulatory environment and does not provide the complete combination of QR scanning, expiry-aware verification, and consumer reporting implemented in PVApp.

The Ghana FDA's MedAlert initiative provides information concerning recalled, withdrawn, and counterfeit products. It represents an important regulatory communication mechanism but differs from PVApp because its primary function is dissemination of regulatory alerts rather than individual real-time product-record verification.

Mpedigree provides product-authentication services using mechanisms such as scratch codes and barcode technologies [11]. Commercial authentication systems can provide rapid confirmation but generally depend on manufacturer or platform participation.

PharmaTrace, an academic blockchain-based approach, explores distributed-ledger technology for pharmaceutical supply-chain traceability. Blockchain can provide an immutable record of transactions, but its infrastructure and transaction characteristics introduce different design considerations from a lightweight real-time cloud database.

Sproxil provides commercial product-authentication services based on consumer-entered authentication codes and related mechanisms. Its commercial deployment model differs from the open regulatory information model explored in PVApp.

mVerify in Kenya provides medicine verification capabilities targeted primarily toward healthcare workers. Its intended user group therefore differs from the general consumer focus of PVApp.

2.2 Research gap

Existing product-verification systems demonstrate that verification can be implemented across diverse channels such as web services, SMS, barcode systems, and blockchain. The present study focuses on integrating key operational capabilities into a unified, cloud-backed mobile architecture. Central to this design is the combination of QR code-based product identification with real-time cloud database access, enabling instant access to current product data.

This setup powers an approval- and expiry-aware verification engine, giving consumers immediate visibility into product authenticity and shelf-life status directly on their devices. To handle non-compliant or questionable items, the mobile architecture incorporates consumer-generated incident reporting, allowing users to flag issues directly from the point of scan.

Behind the scenes, the architecture bridges user activity with system administration. It provides web-based administrative management alongside automated synchronisation between mobile and administrative clients, ensuring that reported incidents, status updates, and product records stay aligned across the entire platform.

The objective is not to claim that individual components are novel. Rather, the contribution lies in their integration into a practical, cloud-backed framework tailored to the Ghanaian regulatory context.

3. Cloud-Based System Architecture

3.1 Architectural overview

PVApp operates on a three-tier, cloud-backed architecture designed to separate presentation, data management, and core services. At the top of this system is the client tier, which delivers the primary interface through which users interact with the application.

Supporting the application's underlying operations are two cloud-based backend layers. The cloud data tier handles persistent storage, database operations, and data retrieval, while the cloud infrastructure and service tier manage hosting, external API integrations, server-side processing, and platform scalability.

Figure 1 shows the three-tier architecture of the cloud-backed PVApp, essentially how the Android app and web admin panel communicate with Firebase and the supporting services. The client tier contains two application interfaces. The first is the native Android application used by consumers. The second is a web-based administrative panel for managing product records and consumer reports.

The cloud data tier is implemented using Google Firebase Realtime Database. Product records are stored in the/products collection, while consumer-submitted reports are stored in the/reports collection. Firebase represents these collections as JSON-based data structures.

Firebase's managed cloud services provide the infrastructure tier. The architecture therefore avoids the need to develop and maintain a separate application server for the current prototype.

Communication between the Android application and Firebase is handled through the Firebase Android SDK. The administrative interface communicates with Firebase through the Firebase JavaScript Web SDK. QR code scanning is performed locally through Google ML Kit Code Scanner, after which the decoded identifier is used to query the cloud database.

This arrangement allows the mobile application and administrative interface to act as synchronised clients of a common cloud data source.

3.2 Cloud data model

The system relies on two principal data entities to organise product records and user submissions. The first is the Product entity, which stores core product parameters and verification data, including productId, productName, companyName, manufacturingDate, expiryDate, and the overall approved status. Additionally, it maintains links to visual assets and scannable markers via qrCode and imageName.

The second entity, **Report**, tracks user-submitted issues and operational logs. It captures contact and context details, specifically reporterName, contact, productName, companyName, reason, location, and an open-ended message. System metadata is managed through timestamp and status fields to support ongoing tracking and administrative review.

Reports are initially assigned a Pending status. Administrators can subsequently update reports to states such as Reviewed or Resolved.

The use of a centralised cloud data model allows both mobile and web clients to operate on the same underlying product and report records.

3.3 Real-time synchronisation

A key architectural characteristic of PVApp is real-time synchronisation. When an administrator adds or modifies a product record through the administrative panel, the change is written to Firebase. The Android client uses a Firebase listener to observe the /products node. Consequently, changes to product data can become available to the mobile client without requiring a new application release.

Similarly, consumer reports submitted through the Android application are stored in /reports. The administrative panel observes this collection and displays incoming reports.

This design separates application distribution from data management. Product records can therefore evolve independently of the Android application binary.

3.4 Cloud service selection

Firebase Realtime Database was selected because the application requires straightforward synchronisation of relatively structured product and report records between mobile and web clients. It also provides managed infrastructure that reduces the need to deploy and maintain a dedicated backend server.

The choice is therefore based primarily on development simplicity, real-time synchronisation, and suitability for a prototype regulatory information system rather than on a claim that Firebase is universally superior to alternative cloud architectures.

4. Software Implementation

4.1 Development environment

PVApp was developed using Android Studio Ladybug (2024.2.1), Java JDK 8, Android Gradle Plugin 8.6.1, and Gradle Wrapper 8.7.

The Android application targets Android API 34 and supports a minimum API level of 28. Key dependencies include Firebase Android SDK BoM 34.10.0, Google ML Kit Code Scanner 16.1.0, Glide 4.16.0, and Android Navigation Component 2.9.6.

The administrative panel was implemented using HTML, CSS, and JavaScript with the Firebase JavaScript Web SDK version 10.12.5.

4.2 Application architecture

The Android application follows the Model-View-ViewModel (MVVM) architectural pattern. Activities and Fragments manage presentation and user interactions, ViewModels manage presentation logic and application state, while model classes represent domain data.

This separation was intended to improve maintainability and testability by separating presentation concerns from application data and logic.

4.3 Functional modules

4.3.1 Onboarding

The onboarding module introduces first-time users to product search, QR scanning, and product-status checking. A three-slide ViewPager2 interface is used. A Boolean value stored using SharedPreferences records whether onboarding has been completed.

Returning users bypass the onboarding interface and proceed directly to the main application.

4.3.2 Product search

The product search module retrieves product records from Firebase's /products collection. Records are deserialised into Product objects and displayed using a RecyclerView.

Each product entry displays information including the product name, company name, approval status and expiry status.

Text filtering is implemented locally through a TextWatcher. This provides immediate filtering of the records already retrieved by the client.

4.3.3 QR-code scanning

The QR module uses Google ML Kit's Code Scanner. The scanner is configured to recognise QR_CODE formats and uses automatic zoom.

After successful decoding, the scanned value is used to query Firebase:

/products → qrCode → scanned value

If a matching record is found, the corresponding product information is displayed. If no matching record exists, the application provides an appropriate notification to the user.

4.3.4 Three-state verification engine

The verification engine is the main application logic responsible for interpreting product status. The system evaluates two variables: whether the product is approved, and whether its expiry date has passed.

The resulting states are:

State 1 — Not Approved

If approved = = false, the product is classified as **Not Approved**, irrespective of its expiry date.

State 2 — Approved but Expired

If approved = = true and the expiry date has passed, the product is classified as **Approved but Expired**.

State 3 — Approved and Valid

If approved = = true and the expiry date has not passed, the product is classified as **Approved and Valid**.

The expiry date is stored using the yyyy-MM-dd format. The application parses the date using SimpleDateFormat in non-lenient mode and compares it with the device's current date.

A parsing exception is handled to prevent application failure when malformed date data is encountered. Figure 2 shows the decision logic used by the PVApp to determine whether a product is valid. It has three possible outcomes: Not Approved, Expired, or Approved & Valid.

4.3.5 Consumer reporting

The reporting module allows consumers to submit information concerning suspected products.

The form systematically collects essential contact, product, and contextual details from the submitter. It gathers the reporter's name and contact information alongside specific identification details, including the product name and associated company name.

Additionally, the form captures operational context by recording the location and primary reason for reporting. Submitters also have the option to include any relevant additional information to support their claim.

After client-side validation, the report is written to Firebase using a generated unique key. Newly submitted reports receive a Pending status.

The submission button is disabled following successful submission to reduce accidental duplicate reports.

4.3.6 Administrative panel

The web-based administrative panel provides comprehensive management functionality over the cloud database, enabling full oversight of both inventory and user feedback. For product records, administrators have complete lifecycle control, allowing them to create new product entries, view existing listings, update attributes, and delete records as necessary.

In addition to product maintenance, the panel streamlines the management of user-submitted feedback. Administrators can view incoming reports sent by consumers and update the status of each report to track investigations, resolve incidents, and maintain administrative oversight.

Firebase onValue listeners provide real-time updates to the administrative tables.

The administrative panel is intended for controlled internal deployment in the current implementation. Authentication and more restrictive Firebase security rules are required before deployment as a publicly accessible administrative service.

5. Development Methodology

An iterative Agile-inspired methodology was adopted. Development was organised into five principal stages, beginning with a requirements analysis that examined existing product-verification approaches and identified key functional requirements. This initial phase laid the groundwork for the system design stage, where the team developed the cloud architecture, data model, application structure, and user-interface layout.

Following design approval, the project moved into iterative implementation. During this phase, core components were built out, including the Firebase data layer, Android interface and modules, QR scanning functionality, verification engine, reporting module, and administrative panel.

Once implementation was complete, the system underwent functional testing through the execution of predefined black box test cases to ensure all features met operational specifications. Finally, the development process concluded with documentation, focused on preparing both technical guides and user-oriented documentation.

This process allowed individual components to be implemented and tested incrementally. Figure 3 presents the functional requirements of PVApp and their corresponding implementation priorities. The chart shows that ten of the eleven requirements are classified as high priority, covering core functions such as product retrieval, real-time filtering, QR-code scanning, three-state verification, consumer reporting, and administrative operations. FR09, the first-time-user onboarding function, is classified as medium priority, reflecting its supporting role relative to the application's core verification and regulatory functions. Overall, the classification emphasises that reliable product verification, reporting, and administrative management constitute the primary functional objectives of the system.

6. Functional Requirements

Table 1 summarises the eleven functional requirements defined for PVApp and their assigned implementation priorities. The requirements cover the application's core capabilities, including Firebase product retrieval, real-time search, product information display, QR-code scanning and database querying, three-state verification, consumer reporting, and administrative product and report management. Ten requirements are classified as high priority, reflecting their direct contribution to product verification and regulatory operations, while FR09 (user onboarding) is classified as medium priority because it primarily supports first-time user orientation. In summary, the classification indicates that the system prioritises reliable verification, reporting, and administrative functions over supplementary usability features.

Table 1
The principal functional requirements were:

ID	Requirement	Priority
FR01	Retrieve product records from Firebase when the application launches	High
FR02	Filter products in real time through the search interface	High
FR03	Display product name, company, approval, and expiry information	High
FR04	Launch QR-code scanning when requested	High
FR05	Query the cloud database using a scanned QR-code value	High
FR06	Display the appropriate three-state verification result	High
FR07	Allow consumers to submit validated product reports	High
FR08	Store submitted reports with Pending status	High
FR09	Display onboarding to first-time users	Medium
FR10	Provide product CRUD functionality to administrators	High
FR11	Display and update consumer-report status	High

7. Evaluation

7.1 Evaluation methodology

PVApp was evaluated using manual black-box functional testing on physical Android devices and the Android Emulator.

System testing was conducted across Android API levels 28 and 34 to evaluate platform compatibility and operational reliability. A total of twenty test cases were defined and executed to systematically validate core performance across six key functional areas.

The evaluation covered initial user onboarding, product search capabilities, three-state product verification, and QR-code scanning functionality. Additionally, testing assessed the consumer reporting workflow and the administrative panel operations to ensure full functional compliance across both client and backend environments.

Each test case specified an input or user action, an expected outcome, an observed outcome, and a pass/fail status.

7.2 Functional test results

All 20 functional test cases for each model are summarised in Table 2, showing that all test cases achieved the expected outcomes and were marked PASS. The results demonstrate successful operation

across onboarding, product search and verification, QR code scanning, consumer reporting, and administrative functions, indicating 100% success in functional testing.

Representative tests included:

Table 2
The complete original project test log contained 20 functional cases, all of which passed.

Test	Expected result	Observed result	Status
First application launch	Onboarding displayed	Onboarding displayed correctly	PASS
Relaunch after onboarding	Main activity displayed	Main activity displayed directly	PASS
Product search	Matching records displayed	Filtering worked correctly	PASS
Approved, non-expired product	Valid-product result displayed	Correct result displayed	PASS
Approved, expired product	Expired warning displayed	Correct warning displayed	PASS
Not-approved product	Not-approved warning displayed	Correct warning displayed	PASS
Valid QR code	Matching product displayed	Correct product loaded	PASS
Unknown QR code	Product-not-found message	Correct message displayed	PASS
Consumer report	Report submitted	Report submitted successfully	PASS
Admin product creation	Product available to mobile client	Product appeared in real time	PASS
Admin report view	Reports displayed	Reports displayed correctly	PASS

7.3 Observations

Two minor implementation observations were recorded.

Firstly, under slow network conditions, the product list could temporarily appear blank before Firebase data became available. This indicates the need for an explicit loading state.

Secondly, products added through the administrative interface may display the application's default launcher icon rather than a dynamically retrieved product image. This is related to the current implementation's use of APK-embedded drawable resources rather than Firebase Cloud Storage.

8. Discussion

8.1 Contribution of the cloud architecture

The principal technical significance of PVApp is the use of a managed cloud backend to connect multiple application clients to a common product information environment.

The Android application does not require a separate local product database to be distributed with each application release. Instead, product information is maintained in the cloud and retrieved by the application at runtime.

This architecture provides a separation between application logic and regulatory data. Changes to the product registry can therefore be made independently of the Android application release cycle.

The approach is consistent with the broader use of cloud computing for scalable and accessible application services. Existing research in the *Journal of Cloud Computing* has examined cloud data stores, cloud application architectures, cloud service management, and cloud-based systems, demonstrating the relevance of cloud infrastructure to distributed application delivery.

8.2 Real-time synchronisation

Real-time synchronisation is particularly useful in a regulatory environment because product information may change after an application has been distributed.

In PVApp, the administrative panel writes changes to Firebase while the mobile client listens for updates. During testing, products added through the administrative panel became available to the mobile application in real time.

This behaviour demonstrates the practical value of a Backend-as-a-Service architecture for a system where data freshness is important.

However, the evaluation did not quantitatively measure synchronisation latency. The observed real-time behaviour should therefore be interpreted as a functional finding rather than a formal performance benchmark.

8.3 Three-state verification

The three-state verification model extends the conventional binary distinction between valid and invalid products. PVApp separates a product's authorisation status from its shelf life. Since an officially approved product that has passed its expiry date carries a fundamentally different risk profile than a counterfeit or unvetted item that was never authorised, the system replaces traditional binary checks with a three-state classification framework.

Under this model, products are categorised as approved and valid, approved but expired, or not approved. This clear distinction ensures consumers know whether a product is unsafe due to shelf-life degradation or lack of regulatory compliance, while providing regulators with actionable insights into inventory violations and counterfeit threats.

The verification model is implemented at the application layer and depends on the accuracy and freshness of the underlying cloud data.

8.4 Consumer reporting as a cloud service

The consumer reporting module extends the system beyond passive product lookup.

Reports submitted through the mobile application are stored centrally and become available to administrators through the web interface. This creates a simple mechanism for incorporating consumer-generated information into a regulatory workflow.

The current evaluation confirms that reports can be submitted and displayed, but it does not evaluate the quality, reliability, volume, or regulatory effectiveness of real-world consumer reports.

Consequently, the reporting function should be considered a technical capability rather than evidence of improved regulatory enforcement.

8.5 Suitability for resource-constrained environments

A managed cloud backend can reduce the requirement for organisations to operate dedicated application servers. This characteristic may be useful in environments where technical infrastructure and specialist cloud administration resources are constrained.

However, cloud dependence also introduces requirements for reliable network connectivity, service availability, appropriate access control, data protection, and operational governance.

PVApp therefore demonstrates the feasibility of the architectural approach but does not establish its operational suitability for large-scale national deployment.

9. Security, Privacy, and Deployment Considerations

Security is an important consideration for a cloud-backed regulatory application.

The current prototype does not implement user authentication for the consumer application. In addition, the administrative panel is intended for controlled deployment and requires stronger authentication and Firebase security rules before making it available for public use.

The current implementation should therefore not be interpreted as a production-ready regulatory platform.

To ensure robust security, operational integrity, and regulatory compliance, future deployments of the system must prioritise comprehensive authentication and access management. This includes implementing Firebase Authentication alongside role-based access control (RBAC), enforced by restrictive Firebase Security Rules and hard-coded protection for administrative credentials. Restricting

system permissions at both the user and database levels prevents unauthorised data modifications and limits administrative functionality solely to verified personnel.

In addition to access controls, data governance and system auditing require strict security controls. Future updates should integrate automated validation for consumer-submitted reports, comprehensive audit logging to track administrative actions, and robust data protection measures to safeguard personally identifiable information (PII).

Finally, securing API and service configurations, alongside establishing clear data retention policies, will ensure long-term stability, system resilience, and compliance with privacy standards.

These measures are particularly important because the reporting module collects contact information and location information from users.

10. Limitations

The current study has several limitations.

Firstly, the application does not implement comprehensive user authentication. This limits the security of both consumer interactions and administrative operations.

Secondly, meaningful offline functionality has not been implemented. Although Firebase provides default caching behaviour, users operating in areas with poor connectivity cannot be guaranteed reliable product search.

Thirdly, the current product database is populated manually through the administrative panel. There is no direct API integration with Ghana's FDA internal product registration database.

Last but not least, the evaluation focused on functional correctness. It did not include large-scale load testing, cloud cost analysis, quantitative latency measurements, usability studies with consumers, security penetration testing, or long-term operational evaluation.

These limitations mean that the current system should be regarded as a functional prototype and research implementation rather than a production-ready national regulatory infrastructure.

11. Future Work

Future development will focus on seven areas.

11.1 Authentication and authorization

Firebase Authentication and role-based access control will be introduced for both consumer and administrative services. Firebase security rules will be configured to restrict write and administrative

operations.

11.2 Cloud-based image storage

Product images will be migrated from APK resources to Firebase Cloud Storage. This will allow administrators to add and update product images without rebuilding the Android application.

11.3 Offline operation

Firebase offline persistence and a more explicit local caching strategy will be investigated to support product verification under intermittent connectivity.

11.4 Notifications

Firebase Cloud Messaging will be integrated to notify users when the status of a submitted report changes.

11.5 Regulatory database integration

A secure integration mechanism with the FDA Ghana product-registration infrastructure will be investigated. This would replace the current manual data-seeding process and improve the timeliness and reliability of the product registry.

11.6 Cross-platform support

An iOS implementation will be investigated to extend access to users of Apple mobile devices.

11.7 Cloud performance and scalability evaluation

Future experimental benchmarks will focus on critical system and network metrics. Initial performance tests will measure database response latency, data synchronisation latency across mobile and web clients, and overall behaviour under peak concurrent-client traffic. In parallel, researchers will monitor real-world system loads by measuring cloud resource utilisation and total network bandwidth requirements.

Beyond baseline performance under optimal conditions, future studies will evaluate system resilience, availability, and financial sustainability. Testing scenarios will simulate failure recovery protocols and track system uptime to ensure continuous service availability. Ultimately, combining these technical evaluations with operational cost profiling will yield empirical data on the architecture's long-term scalability and viability in real-world deployments.

12. Conclusion

This paper presented PVApp, a cloud-backed mobile framework for real-time consumer product verification in Ghana.

The system combines a native Android client, Firebase Realtime Database, Google ML Kit QR-code scanning, a three-state product-verification engine, consumer reporting, and a web-based administrative interface.

The principal cloud-computing contribution is the use of a managed real-time cloud data layer to separate product data management from application deployment and to synchronise information between mobile and web clients. Functional evaluation using 20 predefined test cases resulted in a 100% pass rate for the tested workflows.

The findings demonstrate the technical feasibility of using a Backend-as-a-Service architecture for a consumer-facing regulatory information system. The approach provides a relatively lightweight mechanism for maintaining a centralised product registry while enabling mobile users to retrieve current information.

Nevertheless, the current implementation remains a prototype. Authentication, security rules, offline functionality, cloud-based image storage, direct regulatory-database integration, quantitative cloud-performance evaluation, and large-scale user testing remain necessary before production deployment.

The work provides a foundation for further research into cloud-enabled regulatory information systems and mobile consumer-safety services in Ghana and other resource-constrained environments.

Declarations

Competing interests

The authors declare that they have no competing interests.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Authors' contributions

Ibrahim Tanko: Conceptualisation, system design, Android implementation, QR-code module, Firebase integration, and testing.

Nana Kofi Ahoi Appiah-Badu: Requirements analysis, manuscript preparation, and administrative-panel implementation.

Siddique Abubakr Muntaka and Ajugu Simon: Data modelling, literature review, and manuscript review.

All authors read, reviewed, and approved the final manuscript.

Data and materials availability

The source code and test artefacts supporting this study are available from the corresponding author upon reasonable request.

Ethics approval and consent to participate

Not applicable. The study did not involve human participants or animals.

Consent for publication

Not applicable.

References

1. Ghana Food and Drugs Authority (FDA) (2023) Annual report on product regulation and enforcement. Ghana Food and Drugs Authority
2. World Health Organisation (2020) A study on the public health and socioeconomic impact of substandard and falsified medical products. World Health Organisation
3. Twumasi YAA (2020) Counterfeiting and Piracy: The Role of IT – The Case of the Ghanaian and Nigerian Fashion and Beauty Industry. IGI Global Scientific Publishing
4. Mantey PK et al (2024) Enhancing Drug Traceability in Ghana to Combat Counterfeit Medicines. Centre for Applied Research and Innovation in Supply Chain-Africa (CARISCA), KNUST
5. KPMG. West Africa Banking Industry Customer Experience Survey (2024) https://kpmg.com/gh/en/insights/consumer/digital-trends.html
6. Kumar R (2023) Emerging Digital Technologies for Pharmaceutical Drug Traceability. Univers J Pharm Pharmacol. :34–40
7. Food Safety and Standards Authority of India (FSSAI) (2025) Scan Before You Eat: What FSSAI's New QR Code Rule Means for You
8. Azubuike C (2025) NAFDAC unveils tools to check fake drugs in North-East. Punch Nigeria
9. Google (2024) Firebase Realtime Database documentation. https://firebase.google.com/docs/database
10. Adeyeye MC et al (2026) NAFDAC launches improved med safety app. NAFDAC. https://nafdac.gov.ng
11. Mpedigree (2026) Mpedigree
12. Marikyan D, Papagiannidis S (2026) Technology Acceptance Model: A review
13. Google ML (2024) Kit. Barcode scanning API guide. Google

14. Mehta A (2022) Comparative Analysis of Backend-as-a-Service Solutions for React Native Mobile Applications: Firebase vs Supabase. Haaga-Helia University of Applied Sciences
15. Moroney L (2017) The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. A
16. World Health Organisation. Substandard and falsified medical products (2024) https://www.who.int/news-room/fact-sheets/detail/substandard-and-falsified-medical-products

Figures

Figure 1. PVApp Three-Tier Cloud-Backed Sysytem Architecture

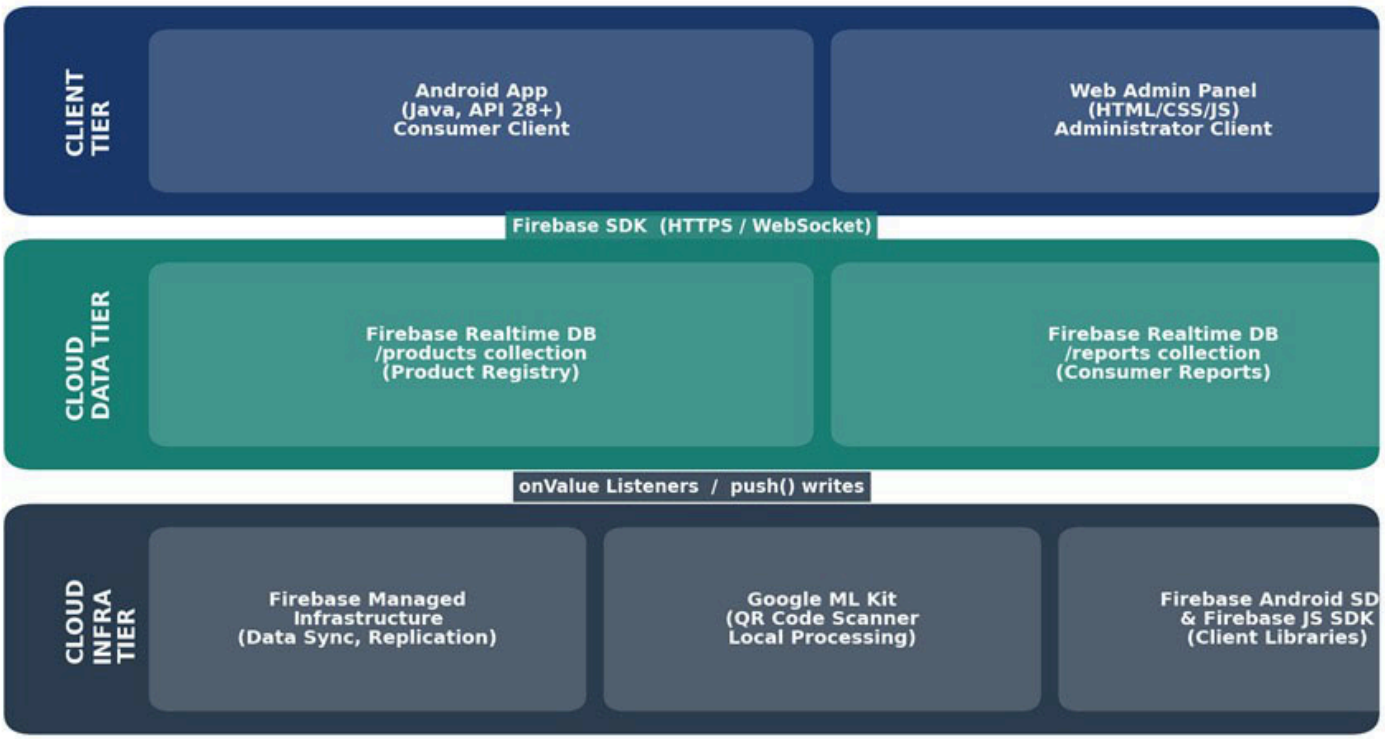


Figure 1

See image above for figure legend

Figure 2. Three-State Product Verification Decision Logic

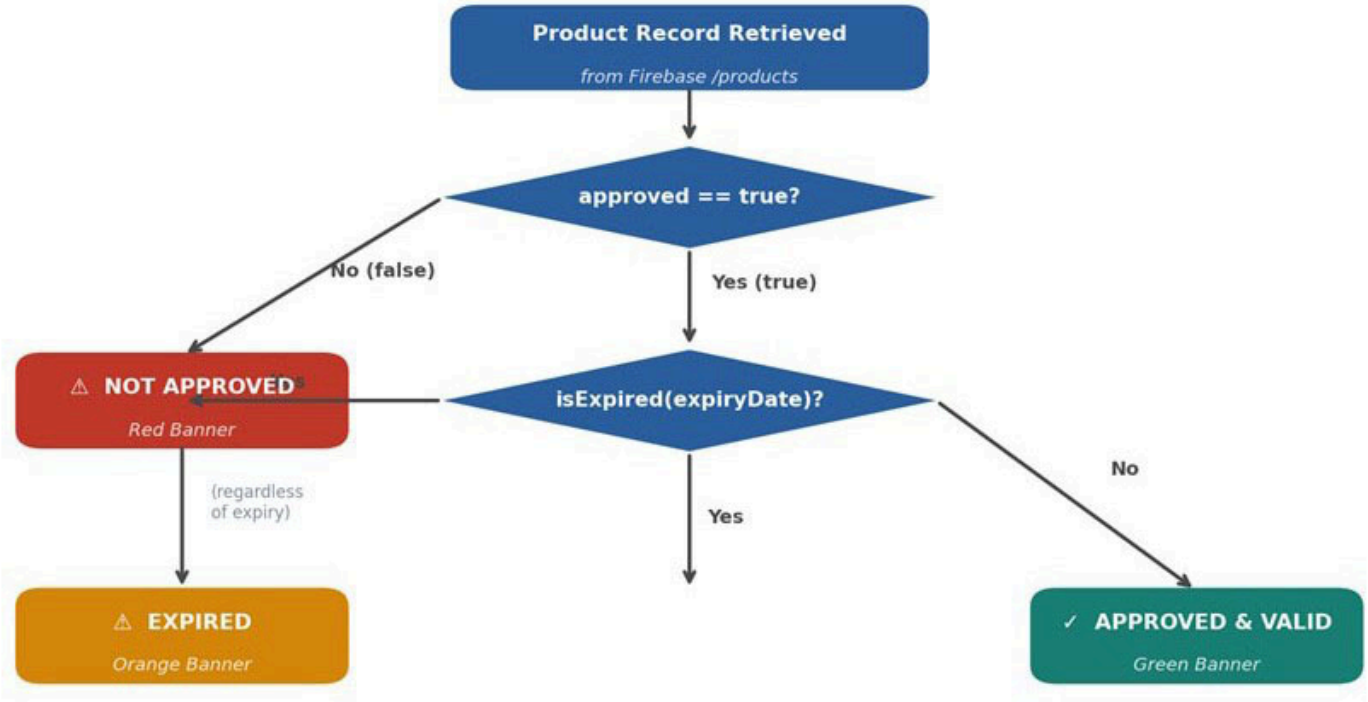


Figure 2

See image above for figure legend

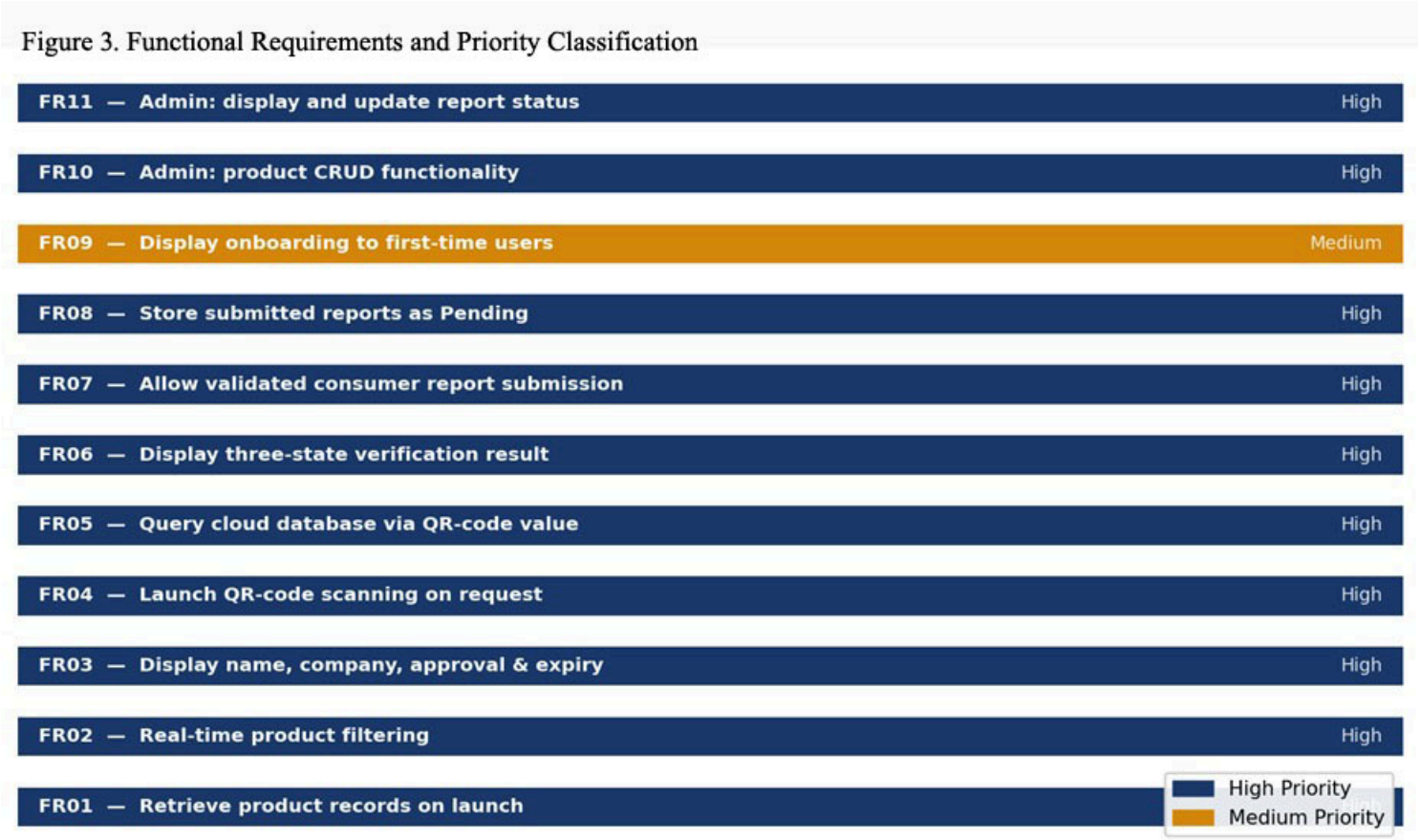


Figure 3

See image above for figure legend

Figure 4. Distribution of Test Cases Across Functional Evaluation Areas

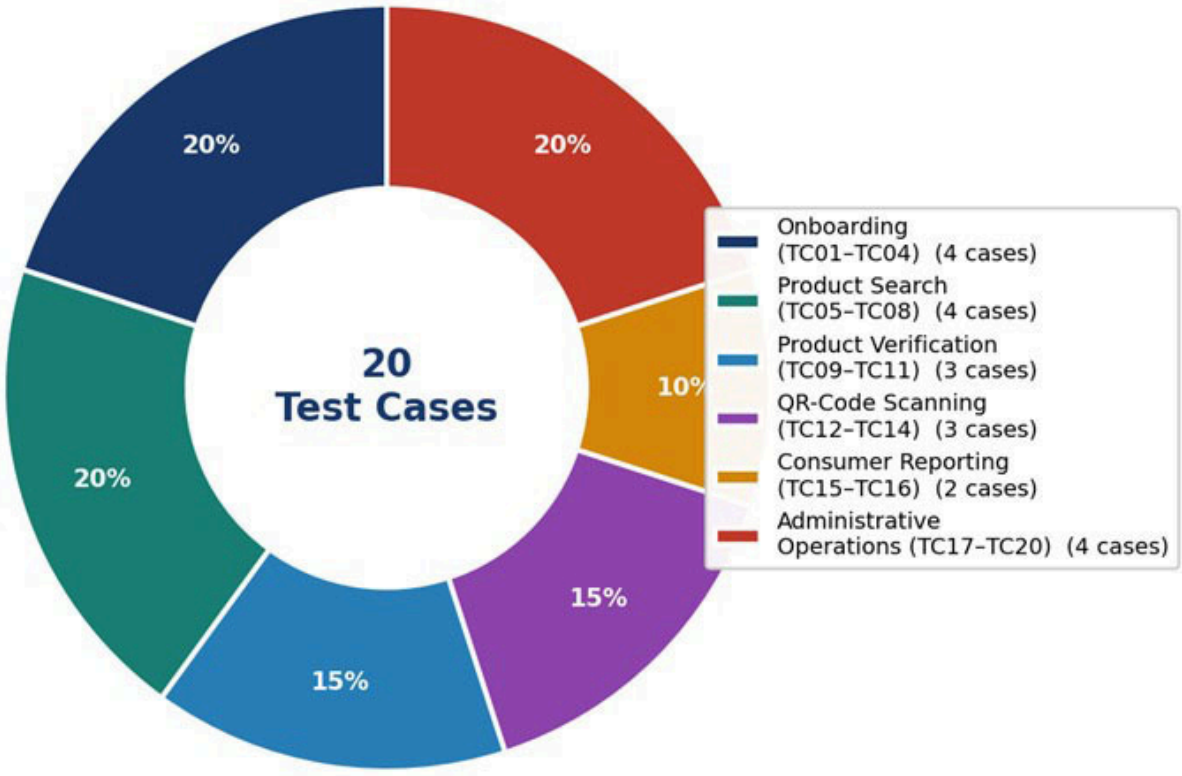


Figure 4

See image above for figure legend

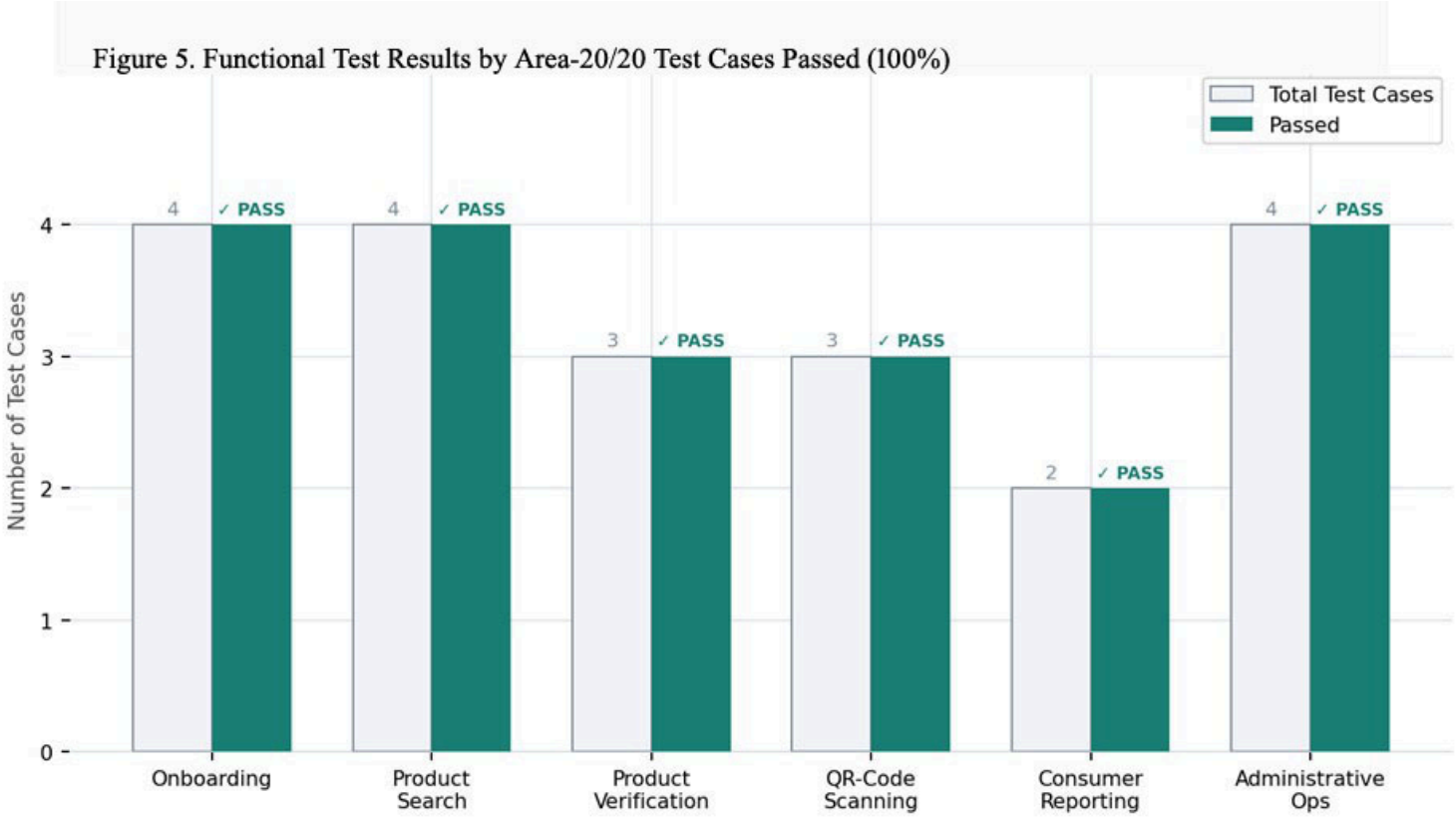


Figure 5

See image above for figure legend

Figure 6. Cloud Performance Evaluation Dimensions:
Current Coverage vs Future Work Target



Figure 6

See image above for figure legend