KARYA ILHAM PUBLISHING
CONNECTING MINDS, BRIDGING IDEAS, INSPIRING READERS

Journal of Advanced Research in Computing and Applications

Journal homepage:
https://karyailham.com.my/index.php/arca/index
ISSN: 2462-1927

A Comparative Study of Data Synchronization Frameworks for an Offline-First Survey System

Li Yunshu[1], Nur Nasuha Mohd Daud[1,*], Hou Jiamin[2], Ragil Tri Atmi[3]

[1] Department of Software Engineering, Faculty of Computer Science and Information Technology, Universiti Malaya, 50603 Kuala Lumpur, Malaysia
[2] Department of Library and Information Science, Faculty of Arts and Social Sciences, Universiti Malaya, 50603 Kuala Lumpur, Malaysia
[3] Department of Information and Library Science, Faculty of Social and Political Sciences, Universitas Airlangga, Surabaya, Indonesia

ARTICLE INFO

Article history:
Received 2 July 2026
Received in revised form 28 August 2026
Accepted 31 August 2026
Available online 16 September 2026

Keywords:
offline-first, data synchronization, survey systems, PouchDB, CouchDB, Firebase Realtime Database, conflict handling, weak-network robustness

ABSTRACT

Offline-first behaviour is essential for field survey systems operating under intermittent or weak connectivity. This paper compares two synchronization approaches in a survey-system prototype: Firebase Realtime Database (RTDB) through a gateway-to-RTDB route and PouchDB/CouchDB using local-first replication, with REST retained as an online-oriented baseline. A standardized experimental harness evaluates latency and completion behaviour, conflict handling, data retention and recovery, and weak-network robustness. Results show that all routes are viable under stable connectivity, but differences become clearer as network conditions deteriorate. PouchDB/CouchDB provides stronger offline continuity because data are committed locally before remote synchronization, while its revision-based replication offers clearer conflict visibility. Firebase RTDB provides convenient cloud integration and real-time capabilities, but the evaluated gateway route remains more dependent on network and server availability. The findings also show that local acknowledgement and remote-visible completion should be evaluated separately. Within the tested prototype, workload, and network profiles, PouchDB/CouchDB provides the stronger fit for field survey workflows prioritizing local durability, recovery, and synchronization auditability.

1. Introduction

Mobile survey systems often operate in environments with intermittent or weak connectivity. In such settings, synchronization is a core reliability requirement rather than a secondary performance feature. Survey applications must preserve locally captured records during disconnection and synchronize them safely once connectivity returns. Failures can lead to delayed submissions, incomplete records, duplicate operations, or uncertainty about whether data have been successfully preserved [1, 2].

* *Corresponding author.*
E-mail address: nasuha@um.edu.my

This study compares two synchronization approaches. The first uses Firebase Realtime Database (RTDB) through a gateway-to-RTDB integration (C-1), where Firebase serves as the centralized remote store and offline buffering is handled by the application and evaluation harness. The second uses PouchDB on the client with CouchDB on the server, following a local-first replication model in which data are committed locally before remote synchronization. REST is retained only as an online-oriented baseline. These routes differ in where durability is established, how strongly submission depends on connectivity, and how conflicts are observed and managed.

The evaluation is situated in a workflow derived from the Malaysia Ageing and Retirement Survey (MARS) context, where field data collection may involve unstable connectivity and delayed synchronization. In this setting, safe local storage and immediate server visibility are not equivalent. Conflict handling is also important because concurrent or delayed updates should be assessed by whether information is preserved and conflicting changes remain auditable [4–6].

Accordingly, this paper asks which synchronization approach provides the stronger architectural fit for an offline-first survey system. The comparison focuses on four dimensions: latency and completion behaviour, conflict handling, data retention and recovery, and weak-network robustness. The study provides empirical evidence for architecture selection while distinguishing local durability, remote-visible completion, and conflict governance.

2. Literature Review and Comparative Basis

Offline-first synchronization requires applications to remain usable during network disruption while preserving data for later synchronization. Research on disconnected and replicated systems shows that evaluation should extend beyond transfer speed to include local durability, recovery after reconnection, conflict behaviour, convergence, and performance under degraded connectivity [1]. Studies of electronic field-data collection similarly emphasize reliability and data completeness when digital tools are used under variable connectivity conditions [2, 3]. Additional field-survey evidence shows that digital data quality is also affected by interruption, workload, and completeness of reporting, reinforcing the need for reliable capture and recovery mechanisms in field systems [19, 20].

Distributed synchronization literature further emphasizes convergence and state reconciliation. Jannes et al. examine seamless synchronization for collaborative web services, while Frey et al. study differentiated eventual consistency in large-scale gossip-based systems [9, 10]. Together, these studies suggest that offline-first evaluation should distinguish temporary divergence from eventual convergence and should consider the predictability of synchronization behaviour rather than only whether a transfer eventually succeeds.

Firebase Realtime Database (RTDB) is a managed, centralized real-time data service suited to applications requiring rapid cloud-backed updates [4]. Firebase-based mobile applications have also demonstrated the practical value of real-time backend integration in connected operational settings [18]. Firebase can support offline operation through its native client SDKs. However, this study evaluates the C-1 gateway-to-RTDB route rather than native client-side offline persistence. The results therefore describe the behaviour of the evaluated integration route, especially its dependence on remote connectivity, rather than the full offline capability of Firebase.

PouchDB/CouchDB follows a local-first replication model. Data can be committed locally and later replicated to CouchDB when connectivity is available. Replication-oriented research highlights the value of maintaining availability during disconnection and reconciling distributed states after reconnection [1, 5]. Empirical evaluations of CouchDB and related NoSQL document databases further show that document-oriented storage involves measurable trade-offs in performance,

storage design, and scalability [12, 13]. CouchDB's revision-based model can preserve divergent document histories, allowing conflicts to be detected and resolved later. More broadly, conflict-free replication research emphasizes explicit convergence and conflict-governance mechanisms when concurrent updates occur [6, 11]. This is useful in survey workflows where delayed or concurrent edits may require auditing or reconciliation.

REST is retained as an online-oriented baseline rather than a complete offline-first framework. REST provides an established request-response architecture for distributed web systems [7, 8], but offline durability, deferred submission, retry behaviour, and conflict management usually require additional application-level mechanisms.

Weak-network behaviour is also a distinct evaluation concern. Geo-distributed systems can exhibit long-tail latency even when median response remains acceptable, so tail measures such as p95 are important for identifying operational variability [14]. This is especially relevant to field surveys, where intermittent connectivity can amplify delay, retry costs, and uncertainty about when remotely visible completion has occurred.

Finally, comparative evaluation of distributed systems should be reproducible. Research on fault injection and experimental methodology recommends controlled workloads, explicit failure conditions, repeatable instrumentation, and traceable experiment configurations [15–17]. These principles support the use of a standardized harness and common success definitions when comparing synchronization routes whose acknowledgement and durability semantics differ.

Overall, the literature spans field-data reliability, centralized cloud synchronization, replication and eventual consistency, conflict governance, weak-network latency, and reproducible distributed-systems evaluation. For field survey systems, important questions therefore include where data become durable during disconnection, whether conflicts can be identified and governed, whether records recover and become remotely visible after reconnection, and how performance changes under weak connectivity. These concerns form the basis of the four experimental dimensions used in this study.

3. Methods

This study uses a design-oriented comparative methodology within a MARS-like survey workflow. A standardized experimental harness compares Firebase RTDB through the C-1 gateway-to-RTDB route and PouchDB/CouchDB through local-first replication, while REST is retained as an online-oriented baseline. All routes use the same workflow, payload structure, scenario scripts, logging schema, and success definitions. Each condition is repeated five times with 20 submissions per run, and each run is identified by a unique runId.

Two completion measures are distinguished. ACK latency records when a route accepts or commits a submission according to its own semantics. For PouchDB/CouchDB this is a local durable commit, whereas Firebase C-1 and REST depend on an online path. ACK latency is therefore treated as an auxiliary responsiveness measure. Remote-visible latency records when the submission becomes observable in the remote store and is used with Expected, Found, and Missing counts as the main end-to-end comparison.

Four scenarios are evaluated: S1 measures baseline latency and completion behaviour; S2 introduces concurrent updates to examine conflict visibility and retention; S3 tests offline retention and recovery after reconnection; and S4 evaluates weak-network robustness. Network profiles include P0 with no injected impairment, P1 with 200–500 ms added delay, and P2 with 200–1000 ms delay plus a 10% injected HTTP 503 failure rate.

Conflict handling compares last-write-wins (LWW) with a deterministic harness-level merge policy. Recovery is assessed by separating local durability from later remote visibility. Results are summarized using success and completeness measures, median latency, interquartile range (IQR), and relevant tail behaviour.

Ethics Approval and Informed Consent. The study involved literature analysis, prototype implementation, and system-level experiments only; no human participants or identifiable personal data were involved.

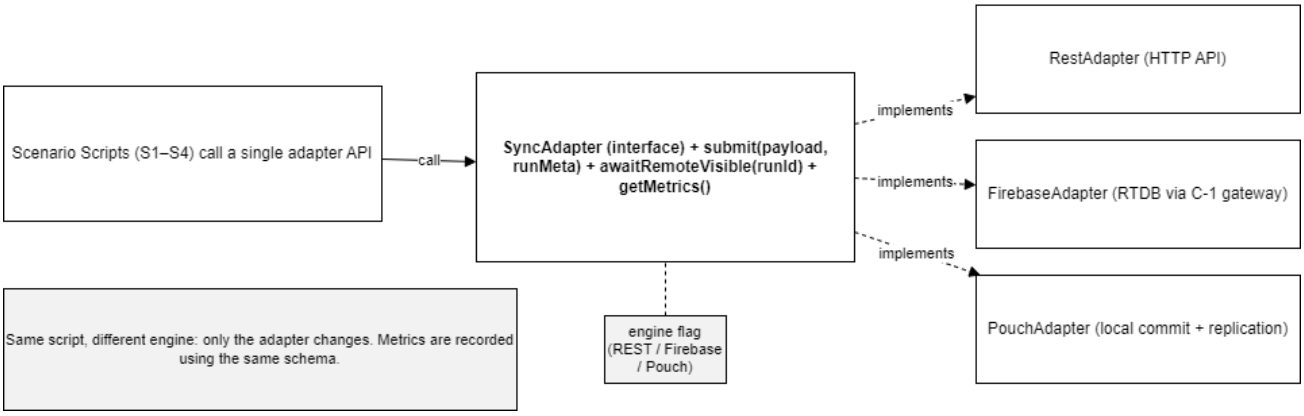

Fig. 1. Standardized synchronization evaluation architecture

Table 1
Comparative basis for framework selection

Criterion	Firebase Realtime Database	PouchDB/CouchDB	REST Baseline
Local durability when offline	Application-level buffering in the evaluated C-1 route; remote durability depends on connectivity	Immediate local commit by design; durability begins on device	Depends on separate queue or retry layer
Conflict visibility	Often application-dependent; risk of silent overwrite without explicit checks	Revision-aware replication can surface divergent histories more clearly	Usually requires application-level versioning
Recovery after reconnection	Good cloud integration, but recovery logic must be observed carefully	Natural fit for eventual replication after reconnection	Possible, but more orchestration burden on the client
Operational complexity	Low infrastructure burden	Higher setup and replication-management burden	Moderate; simple APIs but more custom offline logic

4. Results and Discussion

The experiments show that the evaluated synchronization routes differ less in whether they can operate successfully under stable conditions than in how they establish durability, expose conflicts, and respond to network degradation. These differences are particularly important for field survey systems, where continued data collection and eventual synchronization may be more important than immediate server visibility.

Under the stable network profile (P0), all three routes achieved an ACK success rate and remote-visible success rate of 1.0. PouchDB/CouchDB showed the lowest median ACK latency at approximately 20 ms, followed by REST at 33 ms and Firebase C-1 at 45 ms. Firebase also showed greater tail variability, with a p95 latency of approximately 96.5 ms compared with 43.0 ms for REST and 38.1 ms for PouchDB/CouchDB. However, these ACK values should not be interpreted as a completely equivalent performance ranking. PouchDB/CouchDB acknowledges a local durable commit, whereas Firebase C-1 and REST depend on an online gateway or server path. The result therefore mainly demonstrates the responsiveness advantage of a local-first acceptance boundary rather than a simple database-speed difference.

Conflict handling produced a different pattern. All tested routes completed the conflict workflow successfully, but information retention depended primarily on the resolution policy. Under last-write-wins (LWW), the median retention score for conflicted entities was 0.7143, while the deterministic merge policy achieved full retention of 1.0000. Because the same policy logic was applied at the harness level, this result should not be interpreted as evidence that one synchronization engine inherently performs better merging. Instead, it demonstrates the importance of explicit conflict policy. PouchDB/CouchDB nevertheless provides an architectural advantage in conflict visibility because revision-based replication can preserve divergent histories for later inspection. In Firebase C-1, conflict outcomes depend more heavily on application-level versioning, update structure, and merge logic.

The data-loss-and-recovery experiment further illustrates the distinction between local durability and remote-visible completion. For Firebase C-1 and REST, queue retention reached 1.0 in the observed runs, indicating that offline-buffered submissions were preserved before reconnection. For PouchDB/CouchDB, a separate queue-retention measure is not directly applicable because submissions are committed immediately to persistent local storage. In the extracted runs with complete status coverage, all three routes eventually achieved Expected = Found and Missing = 0. This suggests that the tested routes were able to recover successfully under the evaluated P0 conditions, although they established offline durability through different mechanisms.

The difference became more pronounced under weak-network conditions. Under P1, all three routes still achieved a remote-visible success rate of 1.0, but remote-visible latency increased substantially. Median latency reached 1516.5 ms for REST and 1475.5 ms for Firebase, compared with 823.0 ms for PouchDB/CouchDB. Tail behaviour showed a similar pattern, with p95 latency of 2119.0 ms for REST, 2065.85 ms for Firebase, and 1282.29 ms for PouchDB/CouchDB. These results indicate that the local-first replication route contained latency growth more effectively as network quality deteriorated.

The harsher P2 stress condition, which combined additional delay with injected HTTP 503 failures, further exposed differences in user-facing continuity. Median ACK success was 0.90 for PouchDB/CouchDB and 0.85 for both Firebase and REST. These results are treated as supporting rather than definitive end-to-end evidence because ACK semantics differ across routes, but they are consistent with the architectural expectation that local acceptance reduces immediate dependence on remote connectivity.

Overall, the results favour PouchDB/CouchDB for offline-first field survey workflows where local data safety, recovery, weak-network continuity, and conflict auditability are priorities. Firebase C-1 remains useful where managed cloud integration and real-time services are more important, but the conclusions apply specifically to the evaluated gateway-to-RTDB route and should not be generalized to Firebase's native client-side offline persistence.

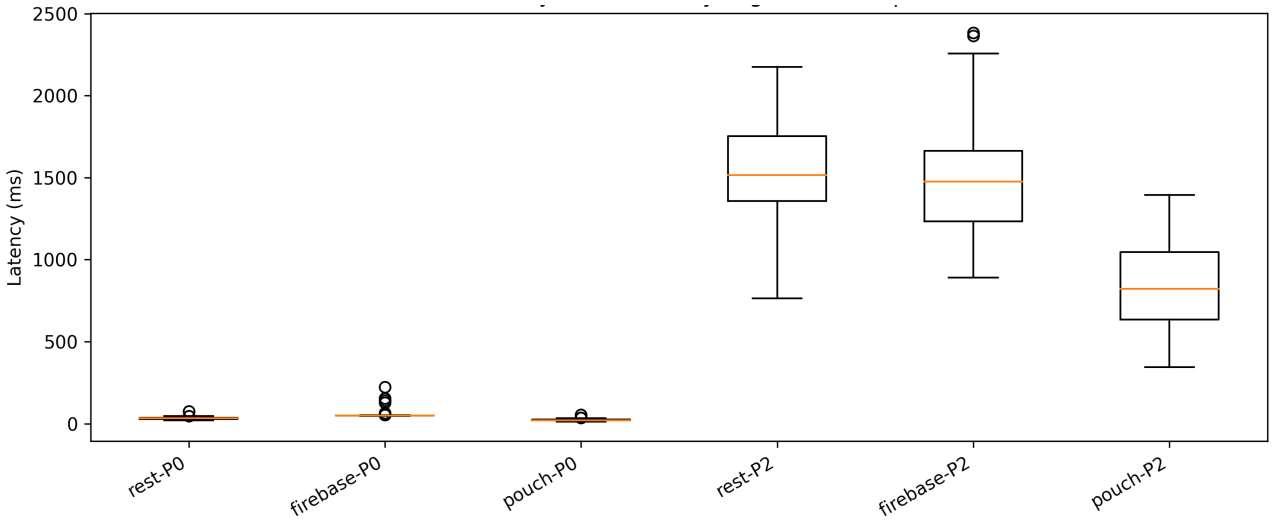

Fig. 2. Remote-visible latency under baseline and weak-network conditions

Table 2
Condensed empirical comparison

Dimension	**REST**	**Firebase C-1**	**PouchDB/CouchDB**
P0 median ACK	33 ms	45 ms	20 ms
P1 remote-visible median	1516.5 ms	1475.5 ms	823.0 ms
P1 remote-visible p95	2119.0 ms	2065.85 ms	1282.29 ms
P1 remote-visible success	1.00	1.00	1.00
Offline recovery	Queue retained	Queue retained	Local durable commit
Conflict visibility	Application-level	Application-level	Revision-aware

5. Conclusion

This study compared Firebase Realtime Database (RTDB) and PouchDB/CouchDB for an offline-first survey system, with REST retained as an online-oriented baseline. The evaluation focused on latency and completion behaviour, conflict handling, data retention and recovery, and robustness under degraded network conditions.

The results show that all evaluated routes can operate reliably under stable connectivity, but their differences become more important when connectivity deteriorates. PouchDB/CouchDB provides the strongest fit for offline-first field survey workflows because data are committed locally before remote synchronization, reducing immediate dependence on network availability. Its replication model also provides clearer conflict visibility and supports more auditable reconciliation. Firebase C-1 offers managed cloud integration and real-time capabilities, but the evaluated gateway-based route remains more dependent on remote connectivity and requires stronger application-level controls for conflict handling and offline buffering.

The study also demonstrates that synchronization should not be assessed using a single latency measure. Local acknowledgement, remote-visible completion, information retention, and eventual

recovery represent different aspects of system reliability and should be evaluated separately. Under the tested conditions, PouchDB/CouchDB showed more stable behaviour under weak-network conditions while maintaining successful recovery and remote convergence.

These findings provide practical guidance for selecting synchronization architectures for field survey systems such as MARS. However, the conclusions are limited to the tested prototype, workload, network profiles, and Firebase C-1 integration route. Future work should evaluate native Firebase client-side offline persistence, larger workloads, longer disconnection periods, and additional operational factors such as storage, battery consumption, and deployment scalability.

References

[1] Mucha, R., B. Balis, C. Grigoras, and J. Kitowski. 2023. "Database Replication for Disconnected Operations with Quasi Real-Time Synchronization." *Computer Science* 24 (3): 401–420. https://doi.org/10.7494/csci.2023.24.3.4831.

[2] Keating, J., et al. 2021. "An Assessment of the Use of Electronic Data Capture Tools for Outbreak Response in Low- and Middle-Income Countries: A Systematic Review." *BMC Public Health*. https://doi.org/10.1186/s12889-021-11790-w.

[3] Thysen, S. M., et al. 2021. "Electronic Data Collection in a Multi-Site Population-Based Survey: EN-INDEPTH Study." *Population Health Metrics*. https://doi.org/10.1186/s12963-020-00226-z.

[4] Namee, K., K. Infueng, J. Polpinij, O. Suwittayapun, P. Promsurapat, and A. Meny. 2022. "Development a Teleconsultation Platform for Outpatients during the COVID-19 Pandemic Based on Cloud Firestore and Realtime Databases." In *2022 14th Biomedical Engineering International Conference (BMEiCON)*. https://doi.org/10.1109/BMEiCON56653.2022.10012077.

[5] Brahneborg, D., W. Afzal, A. Čaušević, and M. Björkman. 2020. "Superlinear and Bandwidth Friendly Geo-Replication for Store-and-Forward Systems." In *Proceedings of the 15th International Conference on Software Technologies (ICSOFT 2020)*, 328–338. https://doi.org/10.5220/0009835403280338.

[6] Ignat, C.-L., V. Elvinger, and H. Ba. 2024. "Synql: A CRDT-Based Approach for Replicated Relational Databases with Integrity Constraints." In *Distributed Applications and Interoperable Systems (DAIS 2024)*, 18–35. Lecture Notes in Computer Science. Cham: Springer. https://doi.org/10.1007/978-3-031-62638-8_2.

[7] Pautasso, C., O. Zimmermann, and F. Leymann. 2010. "RESTful Web Services: Principles, Patterns, and Emerging Technologies." In *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, 1359–1360.

[8] Fielding, R. T. 2000. "Architectural Styles and the Design of Network-Based Software Architectures." PhD diss., University of California, Irvine.

[9] Jannes, K., B. Lagaisse, and W. Joosen. 2022. "Seamless Synchronization for Collaborative Web Services." In *Service-Oriented Computing—ICSOC 2021 Workshops*, 311–314. Lecture Notes in Computer Science 13236. Cham: Springer. https://doi.org/10.1007/978-3-031-14135-5_27.

[10] Frey, D., A. Mostefaoui, M. Perrin, P.-L. Roman, and F. Taiani. 2023. "Differentiated Consistency for Worldwide Gossips." *IEEE Transactions on Parallel and Distributed Systems* 34 (1): 1–15. https://doi.org/10.1109/TPDS.2022.3209150.

[11] Barbosa, M., B. Ferreira, J. Marques, B. Portela, and N. Preguiça. 2021. "Secure Conflict-Free Replicated Data Types." In *Proceedings of the 2021 International Conference on Distributed Computing and Networking (ICDCN 2021)*, 6–15. New York: ACM. https://doi.org/10.1145/3427796.3427831.

[12] Carvalho, I., F. Sá, and J. Bernardino. 2023. "Performance Evaluation of NoSQL Document Databases: Couchbase, CouchDB, and MongoDB." *Algorithms*. https://doi.org/10.3390/a16020078.

[13] Gyorödi, C. A., D. Dumse-Burescu, D. R. Zmaranda, R. S. Gyorödi, G. A. Gabor, and G. D. Pecherle. 2020. "Performance Analysis of NoSQL and Relational Databases with CouchDB and MySQL for Application's Data Storage." *Applied Sciences*. https://doi.org/10.3390/app10238524.

[14] Shithil, S., and M. A. Adnan. 2023. "A Prediction Based Replica Selection Strategy for Reducing Tail Latency in Geo-Distributed Systems." *IEEE Transactions on Cloud Computing* 11 (3): 2954–2965. https://doi.org/10.1109/TCC.2023.3244203.

[15] Amaral, M., M. L. Pardal, H. Mercier, and M. Matos. 2020. "FaultSee: Reproducible Fault Injection in Distributed Systems." In *2020 16th European Dependable Computing Conference (EDCC)*, 25–32. https://doi.org/10.1109/EDCC51268.2020.00014.

[16] Parashar, M. 2021. "Enabling Reproducible Research in Parallel and Distributed Systems." *Computer* 54 (7): 4–5. https://doi.org/10.1109/MC.2021.3055709.

[17] Pan, J., H. Z. Wu, T. Leesatapornwongsa, S. Nath, and P. Huang. 2024. "Efficient Reproduction of Fault-Induced Failures in Distributed Systems with Feedback-Driven Fault Injection." In *Proceedings of the 2024 ACM SIGOPS 30th Symposium on Operating Systems Principles (SOSP 2024)*, 46–62. https://doi.org/10.1145/3694715.3695979.

[18] Tran, T. D., K. T. Huynh, P. Q. Nguyen, and T. N. Ly. 2022. "AttendanceKit: A Set of Role-Based Mobile Applications for Automatic Attendance Checking with UHF RFID Using Realtime Firebase and Face Recognition." In *Future Data and Security Engineering: FDSE 2022*, 432–446. Vol. 1688. Cham: Springer. https://doi.org/10.1007/978-981-19-8069-5_29.

[19] Mboya, D., et al. 2021. "Paradata Analyses from the EN-INDEPTH Study: Typing Speed and Errors, Data Collectors' Impressions on Workload, and Interruption—Findings from an Emergency Phone Survey." *Population Health Metrics*. https://doi.org/10.1186/s12963-020-00241-0.

[20] Dzabeng, F., et al. 2021. "Completeness of Reporting of Births and Stillbirths in Population-Based Surveys: EN-INDEPTH Study." *Population Health Metrics*. https://doi.org/10.1186/s12963-020-00231-2.