IPTEK The Journal for Technology and Science, 37(2), 2026 (e/pISSN:2088-2033, 0853-4098)

DOI: 10.12962/j20882033.v37i2.9783 | Received 11 May, 2026; Revised 23 May, 2026; Accepted 26 May, 2026

ORIGINAL RESEARCH

Geolocation Microservices Integration Framework for Real-Time Geolocation-Based Immigration Surveillance Support Using UML and the Sidecar Pattern

Siera Samudra Joan Putra Surbakti*[1] | Priati Assiroj[2] | Cakra Trinata[2] | Edward Joseph-Pierce Majewski[3]

[1]Department of Immigration, Polytechnic of Immigration and Corectional Indonesia, Tangerang, Banten, 15119, Indonesia, Email: sierasamudra@gmail.com

[2]Departement of Informatics Engineering, Polytechnic of Immigration and Corectional Indonesia, Ministry of Immigration and Corectional Republic Indonesia, Tangerang, Banten, 15119, Indonesia

[3]College of Arts, Humanities, and Social Sciences, University of Maryland, Baltimore County, Baltimore, MD 21250, United States of America

Correspondence

*Email: sierasamudra@gmail.com

Present Address

Faculty of Immigration, Politeknik Imigrasi dan Pemasyarakatan, Jl. Satria - Sudirman, Kota Tangerang, Banten 15119, Indonesia

Abstract

This study proposes a geolocation microservices integration framework for real-time geolocation-based immigration surveillance support using UML and the sidecar pattern. Here, real-time refers to live acquisition, transmission, and dashboard updating of geolocation data, while positional accuracy remains subject to GNSS, device, and network conditions. The work is positioned as an early-stage prototype, not as a production-ready monitoring platform. The prototype was evaluated through multi-location testing using GNSS-based positional classification, descriptive error metrics, participant-reported device and network-signal conditions, and architectural comparison. Repeated coordinate pivots from the same user within 5 m were merged into one representative observation, while close pivots from different users were retained. Of 50 retained observations, 46 were classified as Very High, three as High, and one as Moderate, with MAE of 15.99 m, RMSE of 33.85 m, median error of 8.87 m, and SD of 30.14 m. The 200 m boundary case shows that repeated sampling and corroboration are required before enforcement. Microservices provide clearer service boundaries and fault isolation than a monolithic baseline, while the sidecar pattern strengthens logging, health-checking, and auditability. The framework may support time-to-awareness only under lawful authority, privacy safeguards, broader sampling, and load testing.

KEYWORDS:

geolocation, immigration surveillance support, microservices, sidecar pattern, UML

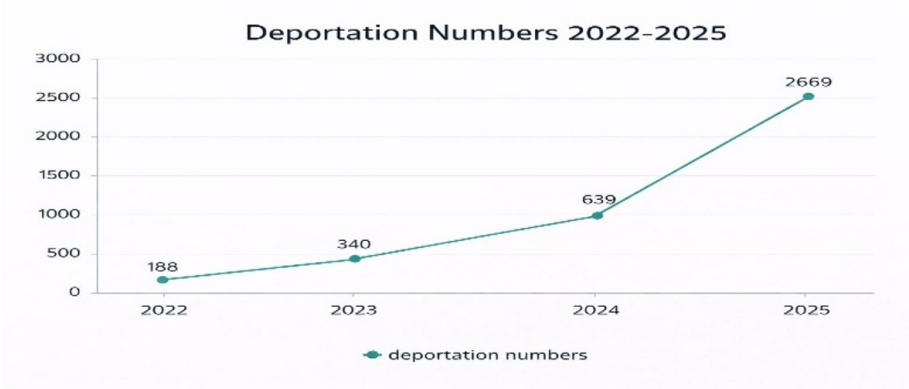

FIGURE 1 Graph of Deportation Cases Involving Foreign Nationals, 2022–2025.

1 | INTRODUCTION

Indonesia is the world's largest archipelago, comprising over 17,000 islands with many official and unofficial entry points. This geography makes immigration control an essential part of national sovereignty, border security, and administrative supervision. The challenge is not limited to checking residence permits. It also includes preventing transnational crime, responding to the mobility of foreign nationals, and reducing dependence on document checks, manual reports, and episodic field surveillance that may miss dynamic cross-border movement. This paper uses the term foreign national for non-Indonesian persons subject to immigration stay and mobility supervision, and Indonesian citizen for citizens of Indonesia. This terminology keeps the manuscript consistent with English academic usage while emphasizing that privacy and personal-data protection remain relevant to every identifiable person whose data may be processed.

Administrative data from the Directorate General of Immigration show an increase in deportation cases involving foreign nationals, from 188 cases in 2022 to 2,669 cases in 2025 (Figure 1). The figure was compiled from Directorate General of Immigration administrative/public-release statistics and institutional recapitulation for 2022-2025, and is cited as descriptive evidence of monitoring pressure rather than as causal evidence that technology alone reduces immigration violations[1]. Because the proposed system processes location data, privacy and legal safeguards are introduced from the outset. Indonesian immigration law provides an institutional basis for supervising the presence and activities of foreign nationals, but it does not justify unrestricted device surveillance or indiscriminate data collection[2],[3]. Identifiable location data must therefore be handled according to lawful basis, purpose limitation, data minimization, access control, retention limitation, security, and accountability principles under the Personal Data Protection Law[4]. The Electronic Information and Transactions framework also indicates that electronic access or extraction cannot be treated as an ordinary administrative shortcut[5]. For this reason, the proposed framework is a scoped geolocation reporting and audit system, not a device-level forensic or spyware-like tracking mechanism. Prior studies show that location-based technologies and microservices have matured across several domains. Cota et al.[6] used IoT and microservices to improve interoperability and maintenance in remote monitoring, while D'Ambrosio et al.[7] demonstrated that independent virtualized microservices can improve adaptability and resilience in security-sensitive systems. These studies support the use of modular service design for distributed monitoring contexts. Security-oriented microservices research is also relevant. Hari et al.[8] showed that microsegmentation under Zero Trust Architecture can reduce attack paths, which is important for systems handling sensitive personal data. Such findings reinforce the need to embed security and access-control considerations in the architecture from the design stage. Research on location performance further supports the technical basis of this study. Knoll et al.[9] validated RTLS accuracy in controlled environments, Chemweno et al.[10] showed the value of RTLS for operational tracking, Mendes et al.[11] discussed hybrid 5G/GNSS location services, and Widhalm et al.[12] demonstrated that GPS-derived mobility data can provide meaningful behavioral information. These studies indicate that location data can support monitoring, although accuracy remains environment-dependent. From the software-engineering perspective, Schwarz et al.[13] classified microservice integration techniques, Meijer et al.[14] showed that architectural pattern selection affects latency and resource use, and Darmoul et al.[15] highlighted modular pattern-based design for resilient cyber-physical systems. These works inform the architectural comparison used in this study. Despite this literature, a gap remains in connecting geolocation microservices with immigration monitoring workflows. Previous work often treats location accuracy, microservice integration,

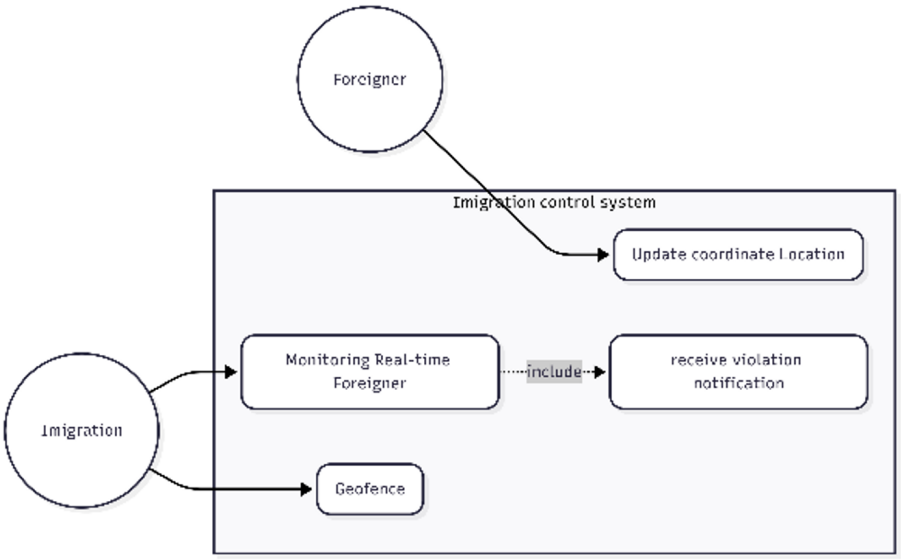

FIGURE 2 Use Case Diagram.

and sidecar-based observability separately, whereas immigration monitoring requires traceable data flows, controlled access, service resilience, audit logs, retention rules, and transparent handling of sensitive personal data. This study addresses that gap by designing and validating a prototype geolocation microservices framework that can support immigration monitoring without modifying the core immigration system. UML models the functional workflow, service interaction, and deployment structure, while the sidecar pattern separates logging, tracing, health monitoring, and observability from the primary geolocation service. The novelty of this research lies in four linked contributions. First, it formulates a domain-specific integration model for immigration monitoring rather than a generic location-tracking application. Second, it links geolocation error interpretation with GNSS-based subject classification and statistical error metrics, so the results can be read as operational evidence rather than as an unsupported percentage accuracy claim. Third, it compares monolithic, microservices, non-sidecar, and sidecar-enabled configurations under the same reporting workflow. Fourth, it positions the sidecar pattern as a privacy- and audit-oriented integration layer for mission-critical public-sector systems, allowing observability and future compliance controls to be added without altering the primary service. Accordingly, the framework is evaluated as a prototype decision-support architecture, not as a complete operational surveillance platform. Its value depends on lawful deployment, controlled data collection, repeated coordinate sampling, anti-spoofing safeguards, and broader load testing.

2 | METHOD

This study uses a microservices-based system design approach to develop a near-real-time location reporting and monitoring-support system that can be integrated into an existing immigration environment without altering its core structure. The method follows five connected stages: UML-based system design, construction of the geolocation microservices architecture, integration through the sidecar pattern, prototype implementation, and empirical testing for accuracy and performance. The sequence is iterative because mission-critical public-sector systems require a careful integration strategy rather than disruptive replacement.

2.1 | System Design Using UML

Unified Modeling Language (UML) was used to document process flows, actor interactions, and service infrastructure in a structured form. UML was selected because it provides a standard notation for communicating system design and aligning the prototype with immigration monitoring workflows. Three UML diagrams were constructed. The Use Case Diagram shows the interaction between the Immigration Officer and the main functions: monitoring foreign-national location, receiving violation notifications, updating coordinate location, and managing geofences. This diagram defines the system scope from the operational user's perspective.

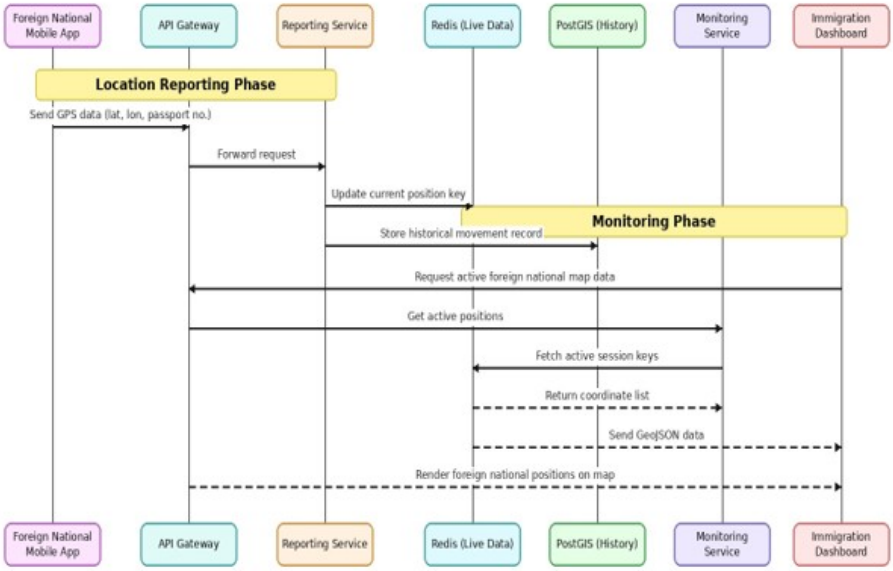

FIGURE 3 Sequence Diagram.

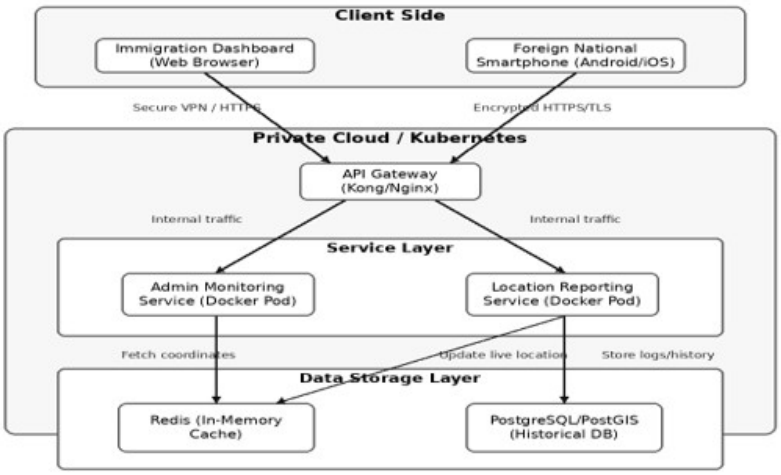

FIGURE 4 Deployment Diagram.

The Sequence Diagram visualizes communication during location data transmission. The Foreign National Mobile Application sends GPS data through the API Gateway to the Reporting Service, which updates current coordinates and records movement data. The Monitoring Service then retrieves active positions and sends them to the Immigration Dashboard for map rendering, making potential latency points visible.

The Deployment Diagram shows the physical and logical architecture. The client side consists of the Immigration Dashboard and a foreign-national smartphone, while the server side uses a Private Cloud/Kubernetes environment with an API Gateway, Admin Monitoring Service, Location Reporting Service, Redis cache, and PostgreSQL/PostGIS historical database. Communication is protected through VPN/HTTPS or encrypted HTTPS/TLS.

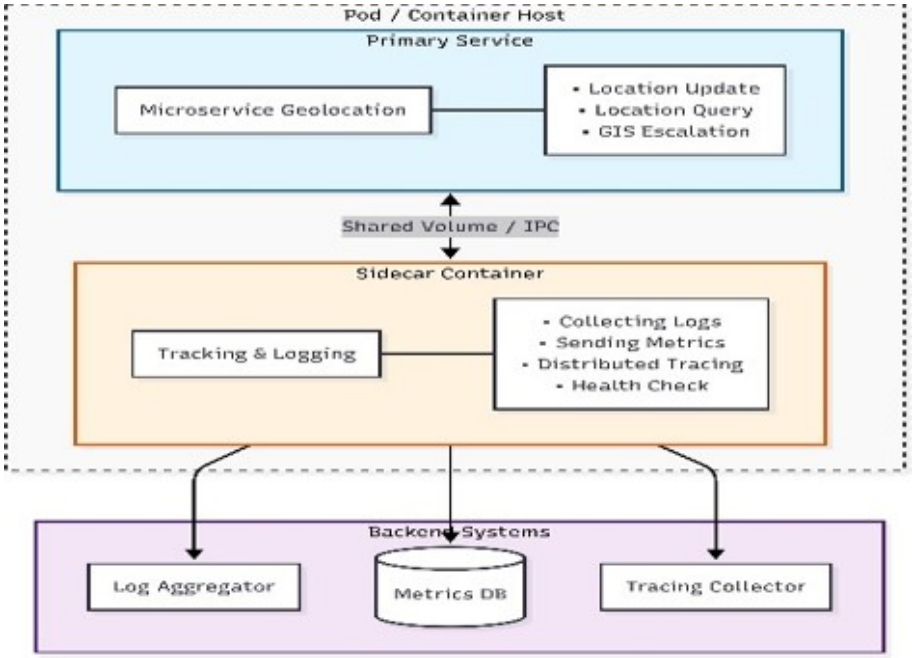

FIGURE 5 Sidecar Pattern.

2.2 | Geolocation Microservices Architecture

The proposed system adopts microservices so each functional service can be developed, deployed, and scaled independently. This approach was chosen over a monolithic structure because it supports modularity, horizontal scalability, fault isolation, and incremental integration with existing systems. The architecture consists of four main components. The API Gateway, implemented using Kong and Nginx, handles routing, authentication enforcement, rate limiting, and load balancing. The Location Reporting Service receives GPS coordinate data, validates the payload, and stores recent and historical records. The Monitoring Service retrieves active coordinates, evaluates geofence rules, and pushes alerts to authorized officers. Redis stores the latest reported coordinate for fast display, while PostgreSQL/PostGIS maintains historical movement data for spatial queries, trajectory analysis, and audit needs. The services are containerized using Docker and orchestrated through Kubernetes. Inter-service communication occurs through a private internal network to reduce exposure of sensitive location data during service-to-service transmission.

2.3 | Integration Using the Sidecar Pattern

The sidecar pattern is used to integrate geolocation reporting without directly modifying the core immigration system. In this pattern, an auxiliary sidecar container is co-deployed with the primary service container in the same Kubernetes Pod, sharing the network namespace and storage volumes while keeping supporting functions separate from the main codebase. In this implementation, the primary container hosts the Microservice Geolocation component for location updates, location queries, and GIS/geofence escalation. The sidecar container handles tracking and logging functions, including log collection, metric transmission, distributed tracing, and health checks. Outputs are sent to a log aggregator, metrics database, and tracing collector to support observability without burdening the primary geolocation service.

This separation is important for immigration systems because observability, security, logging, encryption, compliance auditing, or protocol translation can be added incrementally without increasing regression risk in the primary service.

2.4 | System Hardware and Software Implementation

The prototype used commercial off-the-shelf hardware and open-source software. Sender devices were mainly smartphones with built-in GNSS/GPS or browser-based location services, although several tests were submitted through PC or laptop browsers. Device models were not standardized because the purpose was to evaluate field-oriented geolocation reporting rather than

benchmark one handset. The participant questionnaire recorded device category, device brand/model, and connection signal strength, so the sample could reflect device and network variability. Location data were obtained from device location services and visualized through the Google Maps API for map rendering and reference-point verification. No external GNSS receiver or survey-grade receiver was used; consequently, the accuracy results represent consumer-device positioning rather than geodetic precision. The frontend interfaces were developed using HTML, JavaScript, and CSS, while backend services used RESTful APIs for location updates, session management, and geofence evaluation. Redis was used for recent coordinates and PostgreSQL/PostGIS for persistent spatial storage. Testing used ordinary Wi-Fi or mobile internet at each location; therefore, reported latency should be interpreted as prototype measurement under uncontrolled field-network conditions rather than a production service-level guarantee.

2.5 | Data Collection and Testing

Data collection was conducted through direct prototype testing across Indonesian and international locations. Subjects submitted their location through the Foreign National Mobile Application, after which the system recorded reported GPS coordinates, timestamps, and server-side receipt of the update. The revised analytical sample consists of 50 retained observations: eight first-round observations and 42 additional questionnaire/JSON-matched active-target observations. For the JSON data, near-duplicate coordinate pivots were filtered only within the same user. Coordinates from the same user that were within 5 m were represented by one retained observation, while close pivots from different users were retained independently because they represented different subjects. The geographic spread included multiple Indonesian cities and international test locations in Germany, Saudi Arabia, and Australia. The analyzed variables were positional error, GNSS accuracy classification, output consistency, system stability, participant-reported device type, and signal-based network-latency category. For first-round observations, ground-truth reference coordinates were determined by marking the actual testing point on a digital map and recording its latitude and longitude. For remote participants, the reference point was confirmed through the map interface. For additional browser-based JSON observations, the device-reported horizontal accuracy value was used as an error-distance proxy and matched to the GNSS classification in Table 1 . Each submission was included only after dashboard or database receipt was confirmed.

2.6 | Accuracy Evaluation Method

To evaluate positional accuracy, this study employs the Haversine formula to calculate the great-circle distance between ground-truth coordinates and system-reported GPS coordinates. This method is widely used in geospatial analysis and GNSS-based positioning systems[16],[17]. The Haversine distance is written in Equation (1).

$$d = 2r\arcsin\{\mathrm{sqrt}[\sin^2((\phi_2 - \phi_1)/2) + \cos(\phi_1)\cos(\phi_2)\sin^2((\lambda_2 - \lambda_1)/2)]\} \tag{1}$$

where d is the positional error in meters, r is the Earth radius, φ_1 and φ_2 are the ground-truth and reported latitudes in radians, and λ_1 and λ_2 are the ground-truth and reported longitudes in radians.

In addition to the Haversine formula, this study considers the Equirectangular approximation as a computationally efficient alternative for short-distance calculation. The approximation reduces trigonometric complexity while remaining interpretable for small spatial separations. The distance is calculated using Equation (2).

$$x = (\lambda_2 - \lambda_1)\cos((\phi_1 + \phi_2)/2), \quad y = \phi_2 - \phi_1, \quad d = r\mathrm{sqrt}(x^2 + y^2) \tag{2}$$

While Equation (1) is used as the primary method due to its higher precision, Equation (2) illustrates the trade-off between computational efficiency and distance-estimation precision in real-time systems. All angular values in Equations (1) and (2) are expressed in radians before distance conversion into meters.

To interpret the calculated positional error, this study classifies accuracy levels based on established GNSS performance characteristics. Smartphone-based GPS typically achieves accuracy within 5–20 meters under open-sky conditions and degrades to 20–100 meters in urban environments due to signal obstruction and multipath effects[18],[19].

Based on this classification, positional errors within 0–100 meters are considered reliable for near-real-time monitoring applications, particularly in large-scale monitoring systems where environmental variability cannot be controlled.

TABLE 1 Accuracy Classification

Error Distance (Meters)	Accuracy Level	Interpretation
0-50	Very High	Ideal Condition
50-100	High	Reliable
100-200	Moderate	Acceptable
> 200	Low	Unreliable

2.7 | Statistical Accuracy Metrics

Positional error was evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), median error, and sample standard deviation. MAE indicates average absolute deviation, RMSE penalizes larger deviations, the median shows the typical observation, and standard deviation captures dispersion across heterogeneous device, environmental, and network conditions. Because the retained sample remains limited for operational inference ($n = 50$), these indicators are treated as descriptive validation metrics, not inferential hypothesis tests.

2.8 | Comparative Experimental Configuration

The architectural comparison included two configurations: monolithic versus microservices deployment, and microservices without sidecar versus microservices with sidecar. The same geolocation reporting task, subject data, and dashboard monitoring workflow were used as the functional basis for comparison. The monolithic configuration placed coordinate reporting, monitoring, geofence evaluation, and logging in a single application layer. The microservices configuration separated these responsibilities into an API gateway, location reporting service, monitoring service, Redis cache, and PostgreSQL/PostGIS storage. The comparison used latency, error interpretation, integration risk, service coupling, observability, auditability, recovery implications, and proportionality of data access as indicators. The non-sidecar configuration embedded logging, metrics, tracing, and health checks in or near the primary geolocation service, while the sidecar-enabled configuration delegated these cross-cutting concerns to a co-deployed auxiliary container. Server workload, high-concurrency error rate, service recovery time, and sidecar overhead are treated as next-phase stress-testing dimensions rather than definitive benchmark results.

2.9 | Proposed Framework

The proposed Geolocation Microservices Integration Framework consists of four layers: Data Acquisition, Processing, Integration through the Sidecar Pattern, and Monitoring. The Data Acquisition Layer collects GPS coordinates from mobile devices. The Processing Layer handles validation and geospatial computation using distributed microservices. The Integration Layer enables logging, monitoring, and tracing without modifying the core service, while the Monitoring Layer visualizes real-time location data for authorized officers. Overall, this architecture is intended to support modularity, scalability, and compatibility with legacy systems.

3 | RESULT AND DISCUSSION

This study implemented a prototype near-real-time location reporting system based on microservices architecture. The findings are discussed through subject-level GNSS accuracy, statistical validation, monitoring performance, microservices evaluation, sidecar integration, and comparative architectural evaluation, with claims limited to prototype evidence.

3.1 | System Accuracy Testing

Accuracy testing compared system-reported coordinates or device-reported horizontal accuracy values with the GNSS classification standard in Table 1 . The evaluation covers multiple domestic and international locations and reports each subject's error distance, device category, signal-based latency category, and GNSS accuracy classification in Table 2 . The results are not interpreted as a single percentage-based accuracy score. Instead, each subject is classified according to measured or device-reported error. Table 2 shows that 46 of 50 retained observations are Very High, three are High, and one is Moderate, with no Low case

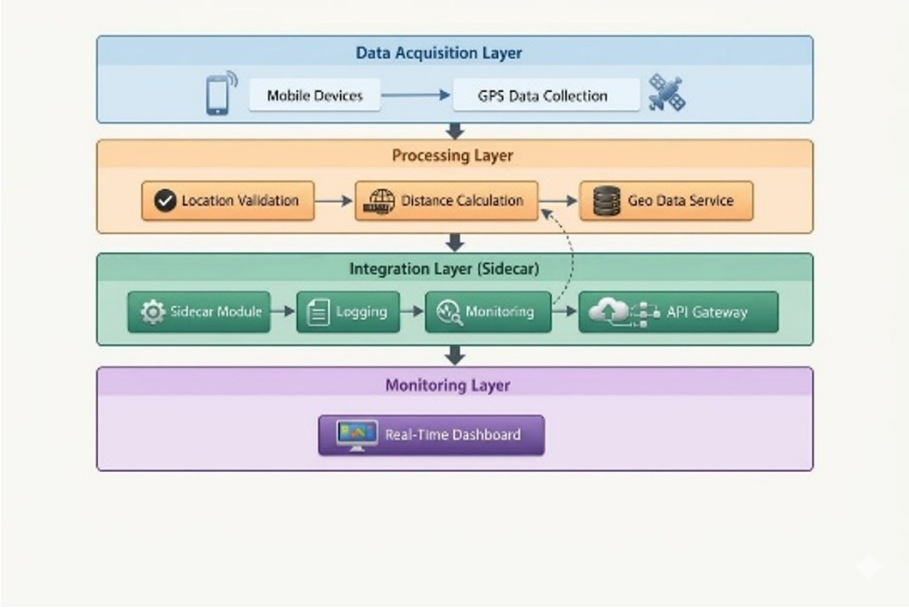

FIGURE 6 Proposed Framework Architecture.

in the corrected retained sample. The statistical summary shows MAE of 15.99 m, RMSE of 33.85 m, median error of 8.87 m, and SD of 30.14 m. RMSE remains higher because it penalizes the 200.00 m boundary case, while the median indicates that the typical observation remains within the Very High class. The dispersion shows that device quality, environment, browser behavior, and network condition still affect the results. Therefore, the system is useful for coarse-to-medium immigration monitoring, but it should not replace physical verification when enforcement depends on exact positioning.

The observed variations are influenced by satellite geometry, obstruction, multipath propagation, device quality, browser geolocation behavior, and network latency. The YM result at 200.00 m is important for geofence-based monitoring because a boundary-level error may incorrectly place a subject inside or outside a restricted zone. High-category browser observations also show that consumer-device reporting can vary even when the dashboard remains online. A safer operational rule is to require repeated coordinate sampling, temporal consistency, geofence tolerance buffers, OSINT or HUMINT corroboration, and officer verification before escalation. Overall, the system provides operationally interpretable positional awareness under heterogeneous field conditions, but the claim remains limited to prototype-scale testing. The latency column in Table 2 is a signal-based category from the participant questionnaire; measured component-level latency is reported separately in Table 6 .

TABLE 2 Results of the Geolocation System Accuracy Test

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
WG	Jakarta	Indonesia	12.18	HP - Samsung Galaxy A55	Low (5G)	Very High
DJ	Jakarta	Indonesia	13.50	HP - Xiaomi Redmi Note 13 Pro	Low (5G)	Very High

Continued on next page...

TABLE 2 – Continued from previous page

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
NFA	Sikka Regency	Indonesia	22.50	HP - Oppo Reno 12	Moderate (4G+)	Very High
YM	Bantul	Indonesia	200.00	HP - Infinix Note 40	Higher (4G)	Moderate
US	Munster	Germany	12.68	HP - iPhone 16 Pro	Low (5G)	Very High
JM	Tangerang	Indonesia	20.00	HP - Vivo V30	Moderate (4G+)	Very High
SH	Medina	Saudi Arabia	61.86	HP - iPhone 16	Low (5G)	High
EA	New South Wales	Australia	4.00	HP - iPhone 16 Pro Max	Low (5G)	Very High
RO	Pematangsiantar	Indonesia	3.71	HP - Xiaomi Redmi Note 13 Pro	Moderate (4G+)	Very High
NES	Tangerang	Indonesia	11.47	HP - iPhone 15	Low (5G)	Very High
AU	Tangerang	Indonesia	4.44	Laptop - MSI Modern 14	Low (5G)	Very High
VG	Yogyakarta	Indonesia	7.33	HP - iPhone 15	Higher (4G)	Very High
WDR	Jakarta	Indonesia	16.14	HP - Vivo V30	Higher (4G)	Very High
BF	Bandung	Indonesia	20.00	HP - Samsung Galaxy A71	Low (5G)	Very High

Continued on next page...

TABLE 2 – Continued from previous page

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
TY	Tangerang	Indonesia	9.21	HP - Xiaomi Redmi Note 13 Pro	Moderate (4G+)	Very High
JG	Tangerang	Indonesia	9.79	HP - Oppo Reno 14F	Moderate (4G+)	Very High
RA	Tangerang	Indonesia	9.79	HP - Samsung Galaxy A55	Low (5G)	Very High
SW	Purwakarta	Indonesia	2.68	HP - Samsung Galaxy A17	Moderate (4G+)	Very High
FA	Tangerang	Indonesia	3.00	HP - iPhone 15	Low (5G)	Very High
ZA	Tangerang	Indonesia	2.90	HP - iPhone 15	Low (5G)	Very High
MAR	Tangerang	Indonesia	60.00	HP - Xiaomi Redmi Note 13 Pro	Moderate (4G+)	High
PP	Tangerang	Indonesia	5.11	HP - iPhone 15	Low (5G)	Very High
WI	Tangerang	Indonesia	60.00	HP - Vivo V30	Higher (4G)	High
L	Tangerang	Indonesia	4.59	HP - iPhone 15	Low (5G)	Very High
TR	Tangerang	Indonesia	12.53	HP - iPhone 15	Low (5G)	Very High
SA	Tangerang	Indonesia	11.45	HP - Samsung Galaxy A15	Moderate (4G+)	Very High

Continued on next page...

TABLE 2 – Continued from previous page

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
NH	Tangerang	Indonesia	10.06	HP - Xiaomi Redmi 14	Moderate (4G+)	Very High
MJ	Jakarta	Indonesia	2.00	HP - Samsung Galaxy A55	Low (5G)	Very High
RF	Jakarta	Indonesia	2.00	HP - Infinix Note 40	Higher (4G)	Very High
MAD	Tangerang	Indonesia	2.06	HP - Samsung Galaxy A15	Low (5G)	Very High
AN	Sidoarjo	Indonesia	3.04	HP - Samsung Galaxy A55	Moderate (4G+)	Very High
SK	Semarang	Indonesia	7.02	HP - Poco X3	Moderate (4G+)	Very High
SA	Jakarta	Indonesia	3.00	HP - Samsung Galaxy A55	Moderate (4G+)	Very High
E	Jakarta	Indonesia	3.02	HP - Infinix Note 40	Higher (4G)	Very High
MS	Jakarta	Indonesia	16.54	PC - Acer Aspire	Low (5G)	Very High
AR	Surabaya	Indonesia	10.32	PC - HP Pavilion	Moderate (4G+)	Very High
AB	Muna Regency	Indonesia	1.30	HP - Oppo Reno 14	Moderate (4G+)	Very High

Continued on next page...

TABLE 2 – Continued from previous page

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
LS	Jakarta	Indonesia	2.57	HP - iPhone 13 Pro	Higher (4G)	Very High
JHW	Jakarta	Indonesia	2.52	HP - Infinix Note 40	Higher (4G)	Very High
ATA	Tanjung Redeb	Indonesia	1.75	HP - Samsung Galaxy M53	Low (5G)	Very High
SCM	Manado	Indonesia	17.60	HP - iPhone 12 Pro	Moderate (4G+)	Very High
SSA	Manado	Indonesia	10.66	Laptop - Acer Aspire	Higher (4G)	Very High
F01	Jakarta	Indonesia	4.38	HP - iPhone 15	Low (5G)	Very High
RC	Manado	Indonesia	5.06	HP - iPhone 11 Pro	Moderate (4G+)	Very High
D	Surabaya	Indonesia	20.25	HP - Samsung Galaxy A54	Moderate (4G+)	Very High
BG	Jakarta	Indonesia	6.19	HP - Xiaomi Redmi Note 13	Low (5G)	Very High
YU	Jakarta	Indonesia	8.32	HP - Oppo Reno 12	Moderate (4G+)	Very High
AF	Jakarta	Indonesia	11.04	HP - iPhone 15	Low (5G)	Very High
RM	Jakarta	Indonesia	39.53	Laptop - Lenovo Idea-Pad	Higher (4G)	Very High

Continued on next page...

TABLE 2 – Continued from previous page

Initial	City	Country	Error Distance (m)	Device	Latency (Signal)	GNSS Accuracy Level
CM	Jakarta	Indonesia	8.53	HP - Samsung Galaxy A55	Low (5G)	Very High

TABLE 3 Statistical Summary of Positional Error

Metric	Result	Meaning
N	50	Retained observations
MAE	15.99 m	Average error
RMSE	33.85 m	Outlier-sensitive
Median	8.87 m	Typical observation
SD	30.14 m	Dispersion
Error range	1.30-200.00 m	Minimum to maximum
GNSS classes	46 Very High; 3 High; 1 Moderate; 0 Low	Retained-observation distribution

Table 3 indicates that the median performance is stronger than RMSE alone suggests because the 200.00 m boundary observation shapes the error profile. The system is adequate for situational awareness and early triage, while enforcement decisions still require repeated sampling and legally authorized assessment.

3.2 | System Monitoring Performance Analysis

The prototype was also evaluated against conventional immigration supervision. Conventional approaches rely on document checks, manual reports, physical checkpoints, and episodic field inspection. The microservices-based prototype allows location data to be submitted, processed, logged, and visualized in near real time under the tested conditions. The phrase near real time is used carefully because the experiment did not yet evaluate high request volumes, unstable cellular handover, peak-load dashboard use, or simultaneous multi-user traffic.

Table 4 shows that the estimated improvement lies in earlier information availability, traceability, and structured escalation. These are operational advantages, not direct evidence that every enforcement case will be resolved faster. Effectiveness is therefore defined as improved visibility and auditability for authorized officers, consistent with prior GEOINT and privacy-governance discussions[20],[21].

3.3 | Microservices Architecture Evaluation

The implementation indicates that microservices provide flexibility and scalability advantages. The API Gateway, Location Reporting Service, Monitoring Service, Redis cache, and PostgreSQL/PostGIS storage can operate as separate components, reducing the risk that changes in one module disrupt the whole system. Production-level performance claims still require large-scale load testing. These findings align with microservices literature, but the relationship is not simply one of direct performance improvement. Schwarz et al.[13] provide a taxonomy of integration techniques, Meijer et al.[14] show that design-pattern selection affects latency and resource utilization, and Xiao[22] notes that resource efficiency should be considered in microservice deployment. Górski[23] and Cavique et al.[24] further support the use of UML-based modeling to clarify integration flows and improve information-system design quality. Together, these studies support the architectural direction of this work while leaving production benchmarking as future work.

TABLE 4 Estimated Operational Improvement

Parameter	Conventional	Proposed
Awareness	Episodic	Continuous GPS
Trigger	Manual report	Geofence/event
Traceability	Officer notes	Timestamped trajectory
Response	Reactive	Earlier triage
Limitation	Spatial gaps	Governance required

3.4 | Sidecar Pattern Integration

The sidecar pattern provides a plausible pathway for integrating logging, metrics, tracing, and health monitoring without modifying the core immigration system. Its main contribution is separation of concerns: observability and future security controls can be added beside, rather than inside, the primary geolocation service. However, sidecar overhead must still be measured under high concurrency through stress testing and failure injection. This finding is consistent with Meadows et al.[25], who proposed sidecar-based path-aware security for microservices without modifying existing software. Their trusted-proxy approach reduces attack surface and is relevant to immigration systems that handle sensitive data. D'Ambrosio et al.[7] similarly support decomposed, virtualized microservices for cyber-resilient operational environments.

3.5 | Implications for Immigration Monitoring and Data Protection

The framework has implications for immigration supervision, national resilience, and personal data protection. Operationally, the generated location data can enrich GEOINT, OSINT, and HUMINT analysis for authorized officers[26]. However, it must remain decision support rather than automated enforcement because location evidence can be affected by signal noise, network disruption, device manipulation, and spoofing. The legal justification for immigration location monitoring must be distinguished from unrestricted device tracking. Immigration law provides a basis for supervising foreign nationals when the purpose is connected to residence permit compliance, presence verification, overstay prevention, or administrative enforcement[2],[3]. At the same time, the Personal Data Protection Law requires identifiable location data to be limited by lawful purpose, necessity, minimization, security, retention, and accountability[4]. The Electronic Information and Transactions framework also cautions against treating electronic access or interception as a routine shortcut[5]. Consequently, the proposed framework is more appropriate for routine immigration monitoring than device-level forensic or spyware-like tools because it collects scoped coordinates, stores auditable timestamps, separates access privileges, and supports retention controls without extracting the full contents of a person's device. This distinction is important because mobile forensic or spyware-like tools may expose communications, files, contacts, credentials, and other data beyond what routine immigration monitoring requires. A microservices geolocation architecture is more governable because the system boundary can be limited to coordinate reporting, timestamped movement logs, access-controlled dashboards, and audit trails. These safeguards support privacy-by-design and governance-oriented deployment[27],[28],[29],[30].

3.6 | Comparative Experimental Evaluation

The comparative experiment covers monolithic versus microservices deployment, sidecar versus non-sidecar integration, and scoped geolocation microservices versus device-level forensic or spyware-like tools. The purpose is not to claim that one architecture universally outperforms another. Instead, the comparison identifies which properties are strengthened by the proposed framework under the same geolocation workflow. Gojek-like ride-hailing systems are used only as architectural analogues because their public documentation describes high-volume microservices and real-time geospatial visualization pipelines[31],[32]. They are not treated as evidence for immigration enforcement outcomes.

Table 5 shows that a monolithic baseline is simpler but less suitable for incremental integration because functional changes concentrate in one application layer.

The microservices configuration offers stronger separation of concerns, independent scalability, and clearer error isolation, although it requires orchestration and service governance. The sidecar-enabled configuration adds value by moving observability and audit functions outside the primary geolocation service. The comparison with forensic or spyware-like tools clarifies the

TABLE 5 Comparative Experimental Evaluation of Alternative Architectures

Experiment	Baseline configuration	Proposed configuration	Comparative finding
Monolithic vs microservices	Single application layer combines coordinate reporting, monitoring, geofence evaluation, and logging.	Separated API gateway, location reporting service, monitoring service, Redis cache, and PostgreSQL/PostGIS storage.	Microservices improve modularity, fault isolation, and independent scaling, but require stronger orchestration and service governance.
Latency and workload interpretation	A single application pipeline makes it harder to isolate whether delay originates from reporting, processing, storage, or visualization.	Component-level measurement separates GPS update, transmission, and backend processing; the current prototype records 950 ms overall average latency.	The result supports prototype-level near-real-time visualization, but not production readiness without request-volume, CPU, memory, and network-variability tests.
Non-sidecar vs sidecar	Logging, metrics, tracing, and health checks remain coupled with the primary geolocation service.	Auxiliary sidecar container handles logging, metrics, distributed tracing, and health checks beside the primary service.	The sidecar improves observability and auditability without modifying the primary service, although overhead must be tested under high concurrency.
Recovery and error isolation	Service failure can affect multiple functions because monitoring, storage, and logging are tightly coupled.	Failure can be localized to a specific service or sidecar function, while logs and health checks support diagnosis.	The architecture is more maintainable for mission-critical systems, but recovery time must be measured through failure-injection tests.
Conventional monitoring vs proposed framework	Manual reports, periodic checks, and fragmented movement evidence.	Continuous geolocation submissions, timestamped movement logs, and dashboard-based monitoring.	The framework improves time-to-awareness and traceability, but enforcement still depends on legal authority and institutional SOPs.
Scoped geolocation microservices vs device-level forensic or spyware-like tools	Device-level extraction or covert tracking may capture communications, files, contacts, and broader personal data.	The proposed framework records scoped coordinate updates, timestamps, logs, and dashboard events through controlled services.	Microservices better support proportional, routine immigration monitoring; intrusive tools should remain limited to narrowly authorized investigations.

proportionality argument: immigration agencies need timely and accountable location evidence, not unrestricted extraction of personal device contents.

TABLE 6 Latency Analysis of System Components

Component	Average Analysis Times (ms)			
	GPS Update	Data Transmission	Data Processing	Total Latency
GPS Module	800	-	-	800
Transmission	-	50	-	50
Backend Processing	-	-	100	100
Overall Average	800	50	100	950

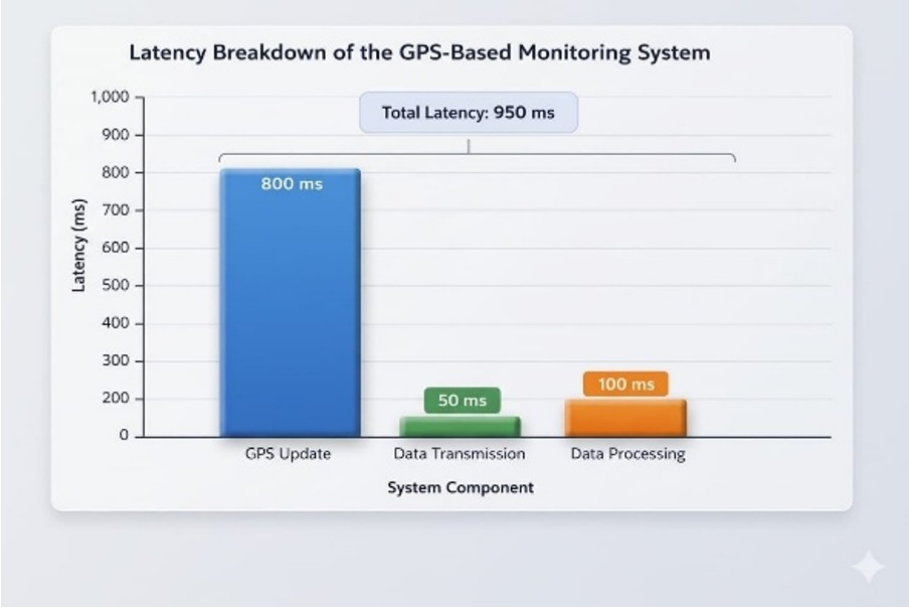

FIGURE 7 Latency Graph.

3.7 | System Performance (Latency)

The prototype achieved an overall average latency of 950 ms across GPS update, transmission, and backend processing. This supports near-real-time dashboard visualization at the prototype level. In the expanded sample table, participant-reported signal strength is used only to categorize expected network-latency conditions and does not replace measured end-to-end latency. Average latency alone is insufficient for production-level claims because the experiment did not report request concurrency, cellular variability, CPU and memory workload, recovery time, error rate, or sidecar overhead under stress. Future testing should add repeated requests per subject, simultaneous users, poor-signal scenarios, controlled network delay, workload monitoring, and failure injection. The latency test was a functional prototype measurement. Request volume was limited to individual submissions and did not simulate many foreign nationals or dashboard users concurrently; therefore, near-real-time capability remains conditional until validated through controlled load and network-variability testing.

3.8 | System Implementation

The implementation demonstrates the feasibility of the prototype as a monitoring-support architecture. The dashboard visualizes location submissions, the e-visa interface illustrates possible integration with existing digital services, and the database structure supports spatial data handling. Production use still requires load testing, access control, retention, and security hardening.

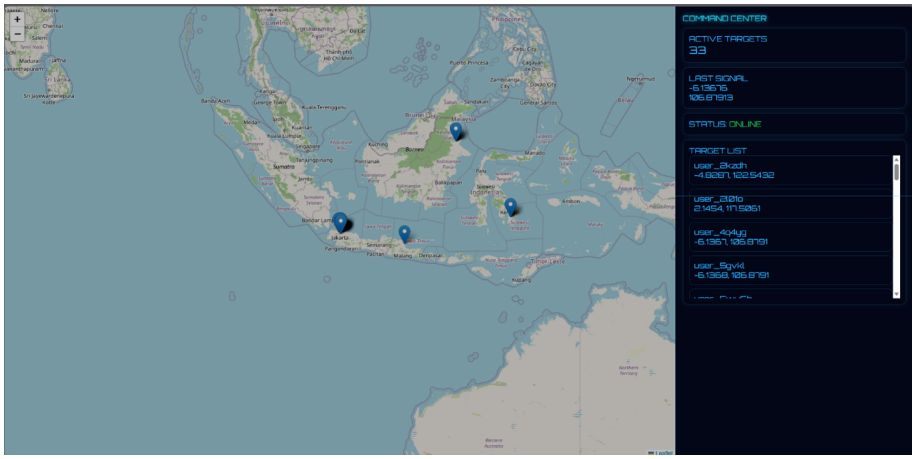

FIGURE 8 Dashboard.

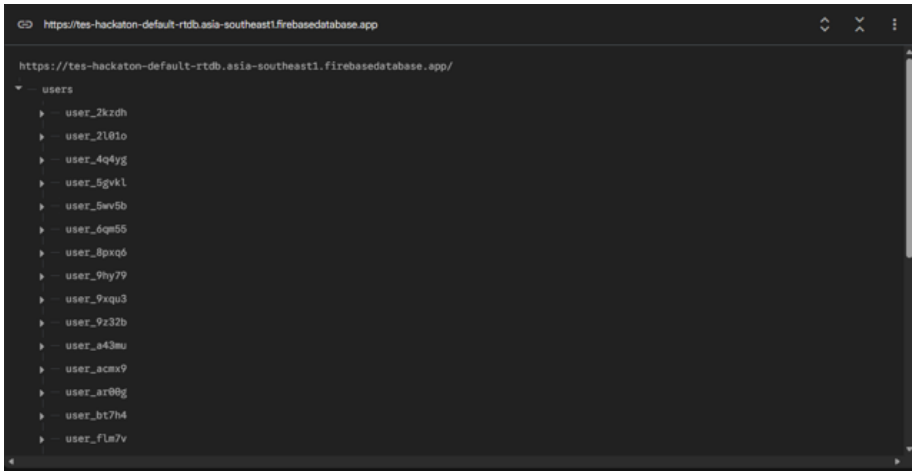

FIGURE 9 Database User.

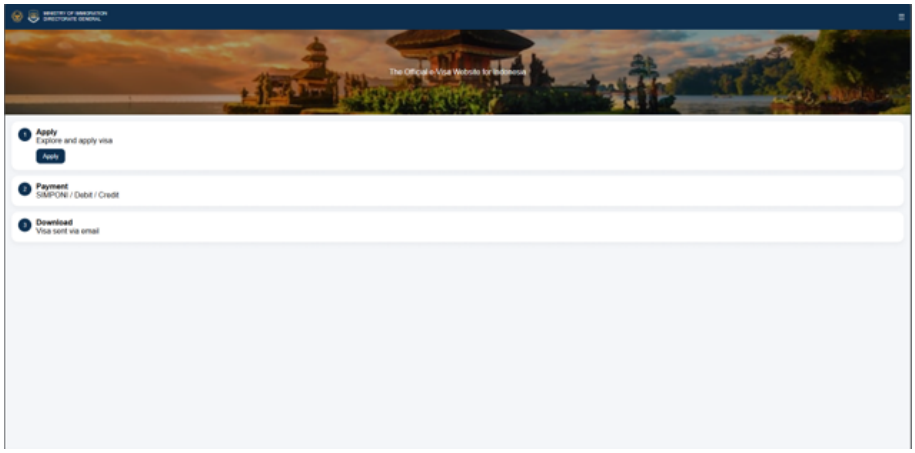

FIGURE 10 E-Visa Interface from the User Perspective.

3.9 | Overall Discussion

Overall, the framework is technically promising, but its contribution should be read as operational visibility rather than absolute positional precision. In immigration supervision, sub-meter accuracy is not always required for early situational awareness. What matters is whether the system can provide timely, auditable and sufficiently reliable location evidence. The GNSS classification shows that most observations are operationally useful, while the RMSE value and the 200 m boundary case show that geofence buffers, repeated sampling, and corroborative intelligence remain necessary before enforcement escalation. The architectural contribution is also significant. A monolithic system may be simpler at first, but it concentrates operational risk in one application layer. Microservices allow reporting, monitoring, and storage components to be modified or scaled independently, while the sidecar separates observability from core location processing. The prototype does not prove that the system will automatically prevent transnational crime or resolve violations within a fixed time frame. It demonstrates improved information readiness: authorized officers can obtain a last known position, view movement history, and prioritize geofence-related anomalies faster than through purely conventional reporting. If a 24-hour escalation target is adopted, it should be framed as an institutional service-level objective supported by the architecture, not as an inherent technical guarantee. The novelty of this study is also practical. Microservices and the sidecar pattern are positioned as a non-intrusive and legally governable integration strategy for mission-critical immigration systems. The architecture is more suitable for routine monitoring than whole-device tracking because it supports the principle of collecting only the information required for a specific administrative purpose. Its limitations include the still-limited prototype dataset, uncontrolled device and network conditions, signal-based rather than fully measured per-subject latency categories, untested GPS spoofing resistance, and the absence of production-scale stress testing.

3.10 | Limitations and Risk Mitigation

The limitations identified in this study should guide the next development cycle. Although the sample was expanded, it remains insufficient for national operational-readiness claims. The 200 m positional error shows that geofence alerts require buffer zones, repeated coordinate sampling, and human review. Unstable GPS signals, poor mobile networks, delayed transmissions, indoor environments, and dense urban areas can reduce reliability. GPS spoofing and device manipulation also require anomaly detection, plausibility checks, device integrity validation, and cross-source corroboration. From a legal and privacy perspective, the system should operate with a clearly defined immigration purpose with role-based access control, encrypted transmission, retention limits, audit trails, and periodic review. A scoped microservices architecture is preferable to whole-device forensic or spyware-like tools for routine supervision because it limits processing to necessary geolocation evidence and auditable service events.

4 | CONCLUSION

This study proposes a geolocation microservices integration framework for real-time geolocation-based immigration surveillance support by combining microservices architecture with the sidecar pattern to enable non-intrusive enhancement of existing systems. The results show operationally interpretable positional data when the error distance of each subject is matched to the GNSS classification standard. Of 50 retained observations, 46 were classified as Very High, three as High, and one as Moderate. MAE was 15.99 m, RMSE 33.85 m, median error 8.87 m, and SD 30.14 m, indicating useful positional awareness with sensitivity to boundary conditions. The implementation also shows practical potential for supporting immigration monitoring through location reporting and integration with digital services such as e-visa platforms. The comparative experiment indicates that microservices provide stronger modularity, fault isolation, and service-boundary control than a monolithic baseline. The sidecar pattern improves observability and auditability compared with a non-sidecar configuration. The architecture also offers a more proportionate approach than device-level forensic or spyware-like tools because it can be limited to location evidence, service logs, and access-controlled audit trails. However, deployment must be accompanied by lawful authorization, proportional data governance, access control, retention limits, audit mechanisms, and procedures for boundary errors, GPS spoofing, and network instability. These safeguards are core requirements for transforming the prototype into an accountable public-sector monitoring system. Future work should expand the number and diversity of test subjects, add repeated coordinate samples, compare monolithic and gateway-only microservices configurations, conduct load and stress testing, measure sidecar overhead and service recovery time, and evaluate legal-operational readiness for immigration governance.

References

1. Imigrasi. Administrative statistics on deportation of foreign nationals, 2022-2025. Directorate General of Immigration, Ministry of Law and Human Rights of the Republic of Indonesia 2025;.
2. Indonesia. Law Number 6 of 2011 concerning Immigration. State Gazette of the Republic of Indonesia Year 2011 Number 52 2011;.
3. Indonesia. Government Regulation Number 40 of 2019 concerning the Implementation of Law Number 6 of 2011 concerning Immigration. Republic of Indonesia 2019;.
4. Indonesia. Law Number 27 of 2022 concerning Personal Data Protection. State Gazette of the Republic of Indonesia Year 2022 Number 196 2022;.
5. Indonesia. Law Number 1 of 2024 concerning the Second Amendment to Law Number 11 of 2008 concerning Electronic Information and Transactions. Republic of Indonesia 2024;.
6. Cota D, Martins J, Mamede H, Branco F. BHiveSense: An integrated information system architecture for sustainable remote monitoring and management of apiaries based on IoT and microservices. Journal of Open Innovation: Technology, Market, and Complexity 2023;9(3):100110.
7. D'Ambrosio N, Perrone G, Romano SP, Urraro A. A cyber-resilient open architecture for drone control. Computers & Security 2025;150:104205.
8. Hari NN, et al. Original article A three-tier microsegmentation framework for enterprise networks under Zero Trust Architecture. Alexandria Engineering Journal 2026;141:150–164.
9. Knoll M, Gygax L, Hillmann E. Pinpointing pigs: performance and challenges of an ultra-wideband real-time location system for tracking growing-finishing pigs under practical conditions. Animal 2024;18(6):101163.
10. Chemweno P, Sullivan BP, Bermperidis G, Thiede S. Exploring the Added-Value of Integrating Real-Time Location Systems for Tracking Critical Maintenance Tools. Procedia CIRP 2022;107:902–907.
11. Mendes B, Araújo M, Goes A, Corujo D, Oliveira ASR. Exploring V2X in 5G networks: A comprehensive survey of location-based services in hybrid scenarios. Vehicle Communications 2025;52:100878.
12. Widhalm P, Ponweiser W, Markvica K, Dragaschnig M. GPS-based analysis of shared e-mobility services. Transportation Research Procedia 2023;72:3489–3496.
13. Schwarz GD, Bauer A, Riehle D, Harutyunyan N. A taxonomy of microservice integration techniques. Information and Software Technology 2025;183:107723.
14. Meijer W, Trubiani C, Aleti A. Experimental evaluation of architectural software performance design patterns in microservices. Journal of Systems and Software 2024;218:112183.
15. Darmoul S, Attajer A, Sallez Y, Chaabane S. Artificial immune systems as a design pattern for disturbance management in cyber-physical production systems. Knowledge-Based Systems 2026;339:115642.
16. Sinnott RW. Virtues of the Haversine. Sky and Telescope 1984;68(2):159–160.
17. Kaplan ED, Hegarty CJ. Understanding GPS/GNSS: Principles and Applications. 3rd ed. Boston, MA, USA: Artech House; 2017.
18. Zandbergen PA, Barbeau SJ. Positional accuracy of assisted GPS data from high-sensitivity GPS-enabled mobile phones. Journal of Navigation 2011;64(3):381–399.
19. European GNSS Agency (GSA). GNSS User Technology Report. Luxembourg: Publications Office of the European Union; 2020.

20. Setiawan P, Purwanto H, Prasetyono B, Jati SP. Digital Transformation in Foreign Surveillance: A Systematic Literature Review (SLR) of the Role of Geospatial Intelligence. Journal of Geoscience, Engineering and Environment Technology 2025;10(02):241–251.

21. Hutagalung GAM. Dampak Teknologi Terbaru dalam Pengawasan Keimigrasian: Antara Efisiensi dan Privasi. Innovative: Journal Of Social Science Research 2023;3(4):8370–8380. http://j-innovative.org/index.php/Innovative/article/view/4351.

22. Xiao X. Architectural Tactics to Improve the Environmental Sustainability of Microservices: A Rapid Review. arXiv preprint arXiv:240716706 2024;https://arxiv.org/pdf/2407.16706.

23. Górski T. UML Profile for Messaging Patterns in Service-Oriented Architecture, Microservices, and Internet of Things. Applied Sciences 2022;12(24).

24. Cavique L, Cavique M, Mendes A, Cavique M. Improving information system design: Using UML and axiomatic design. Computers in Industry 2022;135:103569.

25. Meadows C, Hounsinou S, Wood T, Bloom G. Sidecar-based Path-aware Security for Microservices. In: Proceedings of the ACM Symposium on Access Control Models and Technologies (SACMAT); 2023. p. 157–162.

26. Kotaridis I, Benekos G. Integrating Earth observation IMINT with OSINT data to create added-value multisource intelligence information: A case study of the Ukraine–Russia war. Security and Defence Quarterly 2023;43(3):1–21.

27. Owens K, Eiger Y, Radka B, Kohno T, Roesner F. Understanding experiences with compulsory immigration surveillance in the U.S. In: ACMF AccT 2025 - Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency; 2025. p. 887–899.

28. Saunders N. Security, digital border technologies, and immigration admissions: Challenges of and to non-discrimination, liberty and equality. European Journal of Political Theory 2025;24(2):155–175.

29. Tabassi E. Artificial Intelligence Risk Management Framework (AI RMF 1.0). Gaithersburg, MD, USA: National Institute of Standards and Technology; 2023.

30. Trabelsi Z, Alnajjar F, Parambil MMA, Gochoo M, Ali L. Real-Time Attention Monitoring System for Classroom: A Deep Learning Approach for Student's Behavior Recognition. Big Data and Cognitive Computing 2023;7(1):1–17.

31. Google-Cloud. Introducing Firehose: An open source tool from Gojek for seamless data ingestion to BigQuery and Cloud Storage. Google Cloud Blog 2022;https://cloud.google.com/blog.

32. Suhag R. ATLAS: GO-JEK's real-time geospatial visualization platform. Gojek Engineering Blog 2018;.

How to cite this article: Siera S. J. P. S., Priati A., Cakra T., Edward J. M., (2026), Geolocation Microservices Integration Framework for Real-Time Geolocation-Based Immigration Surveillance Support Using UML and the Sidecar Pattern, *IPTEK The Journal for Technology and Science*, *37(2):141-160*.