Future Technology
Open Access Journal
ISSN 2832-0379

May 2026| Volume 05 | Issue 02 | Pages 13-24

Journal homepage: https://fupubco.com/futech

https://doi.org/10.55670/fpll.futech.5.2.2

Future Publishing LLC

Article

Edge-AI microservice orchestration for private, real-time generative FinTech applications

Kishore Subramanya Hebbar[1]*, Vishal Sharma[2], Jaykumar Ambadas Maheshkar[3]

[1]Intercontinental Exchange Inc, Georgia, USA
[2]Broadridge Financial Services, New Jersey, USA
[3]U.S. Bancorp, Georgia, USA

ARTICLE INFO

Article history:
Received 20 September 2025
Received in revised form
25 November 2025
Accepted 06 January 2026

Keywords:
Edge AI, On-device inference, FinTech, Privacy-preserving AI, Federated learning, Low-latency AI

*Corresponding author
Email address:
hebbar.kishore@gmail.com

DOI: 10.55670/fpll.futech.5.2.2

ABSTRACT

The financial services industry faces mounting pressures to deliver real-time, personalized services while safeguarding sensitive user data under tight regulatory environments. Yet, prevailing AI systems in FinTech remain largely cloud dependent, which introduces latency bottlenecks, privacy exposure, and compliance risk. Meanwhile, industry analyses suggest that Edge AI is rapidly becoming a foundational shift, with predictions that 60% of AI deployments will run partially on device by 2029. However, existing edge AI research often focuses on inference optimization, not full-stack orchestration of financial microservices, and therefore, lacks the integrated, decision-oriented intelligence that is required to operate wholly on the device. In this work, we present an architecture for on-device microservice orchestration of generative AI tailored for FinTech use cases. Our system modularizes AI tasks, such as local LLM inference, fraud detection, biometric authentication, and credit scoring, into services coordinated via lightweight orchestrators (e.g. WASMEdge, Open Horizon). Unlike prior approaches, our system coordinates these services using lightweight WebAssembly-based runtimes, enabling secure, isolated, and efficient execution even on resource-constrained devices. Sensitive data, such as transaction history and biometric templates, remains strictly local, with optional federated synchronization for global fraud pattern sharing. With quantized LLMs, we attain inference latency under 90ms, while local anomaly detection achieves 72% accuracy in simulated financial fraud scenarios. The architecture integrates modular microservices, privacy-first orchestration, and a hybrid federated intelligence layer and is among the first to present a decentralized, compliant, and performance-sensitive AI infrastructure for the FinTech of reality.

1. Introduction

In the financial arena, there is a rapid embrace of AI to meet fast-growing demands for personalization, fraud prevention, and real-time decision making [1,2]. Most GenAI-powered applications in FinTech are also highly cloud-dependent, and the user-sensitive information has to go to some centralized servers for processing [3]. With such an arrangement, latency, data privacy, compliance, and offline availability disadvantages arise [4]. A recent global survey on generative AI in financial institutions states that more than 60% of respondents expressed concern about the exposure of data and regulatory risk with the adoption of cloud-based LLMs for customer-facing services [5]. Simultaneously with this, Edge AI systems also allow positioning of AI computation closer to the destination [6,7]. The use of real-time intelligence is now possible in reasonable environments such as Common Edge, smartphones, tablets, and point of sale terminals without any compromise to user privacy. This advantage of AI reduces latency and improves the privacy of data by storing models and data on the machine. This can also be decided by more flexibility in reaction in a specific context with a user, location-based financial behavior, biometric indications, or noticing of anomalous usage that may not be detected or adjusted to immediately under a centralized system. Edge native designs work well even when connectivity is poor or in an entirely offline environment, making these options extremely appreciated elsewhere in rural and underserved markets [8]. These prepared features are indicative of the global shift towards decentralized finance (DeFi) and sovereign identity systems, both of which will gain an advantage over local, trustful computation paradigms [9]. Still, most existing research focuses on running single AI tasks or models on a device [10]. The least investigated is microservice architecture in which one can

arrange and deploy a series of AI services, fraud detection, biometric validation, and a large language model (LLM) supervision, locally and in real time. To fill this gap, we introduce a modular, microservice-based architecture that allows financial intelligence, up to date, and privacy preserving at any time, solely on the device [10]. With multimodal support, unlike others, there is the execution of large language model (LLM) inference, contextual fraud detection, and biometric authentication that are synchronized with lightweight orchestrators like WASMEdge in our system. This solution is effective as it ensures high privacy levels without compromising intelligence or response time because sensitive financial and biometric information stays on-site, although optional federated learning is provided. This study creates a deployable, multi-mode Edge AI platform purpose-adapted to FinTech, which can make sub-100ms inference and act as a strong fraud detector capable of supporting secure, real-time, and regulation-compliant services at the network edge.

2. Literature review

2.1 Evolution of distributed intelligence and edge integration

The rapid growth of edge computing has transformed how computational workloads are distributed across networks to enable faster analytics and improved data sovereignty. Industry estimates suggest that by 2025, almost three-quarters of enterprise applications will incorporate some form of on-chip inference to maximize performance and reduce reliance on centralized data centers [11]. This change is consistent with broader trends in sustainability and decentralism in system design. Specifically, one can draw parallels between distributed computing infrastructure and complex industrial logistics systems, including SAF supply chains, where local decision-making is used to maintain sustained operations even in cases of regional pebbles of asymmetric resources and demand. The logistics architecture of the SAF industry illustrates the usefulness of the idea of decentralized orchestration. These regional blending, real-time analytics, and coordinating multiple sites are approximated to cut transport inefficiencies by some conservative 30% in industrial estimates. Similarly, applying this analogy to Edge AI, decentralized orchestration reduces the latency and transmission costs in a similar way that it moves the processing close to the origin of the data generation [12]. The ability to process lightweight inference models over heterogeneous devices shares similarities in nature regarding ad hoc resource allocation in SAF procurement systems, in which the sustainability of operation depends on the reconfigurability of supply nodes. Distributed orchestration in edge systems, in turn, represents the same principles of sustainable network optimization that may be noted in intricate industrial supply chains [13].

2.2 Automation and validation in scalable systems

Reliability in distributed computational ecosystems requires scalability and validation. The shift towards microservices, with each service being a small, self-contained module, has increased the use of automated testing tools to ensure the functional integrity of any software before deployment. The automation eliminates the error rate of human operators, reduces verification cycles, and provides deterministic behavior across numerous hardware platforms. Predictive maintenance systems and statistical modeling also enhance system resilience by detecting abnormalities and their propagation across nodes. Automated validation is a notion that is entrenched in hardware design verification. The use of test-vector automation in large-scale silicon validation has been demonstrated to more than 40 times decrease the defect-detection latency period so that parallel verification can be applied to millions of logic paths [14]. The method offers the roadmap of scalable validation when it comes to Edge AI deployments. This reproducibility can be achieved by orchestration platforms by continuously integrating pipelines using microservice workflows simulated before deployment, and this is similar to deterministic verification flows in silicon testing. Such methodologies will provide the reliability of individual microservices functioning on the larger system despite varying hardware conditions of different devices [15].

2.3 Sustainable and energy-efficient computing design

The concept of sustainability has been gaining prominence in computational architecture. The need to conserve energy is not just an environmental requirement, but a requirement of performance that is very important to resource-limited edge devices. The decentralization of computation minimizes the energy overhead costs linked to relaying data to remote cloud servers to enhance the overall sustainability of the system. Through comparative studies, local inference on optimized hardware can reduce the total energy consumption per inference task by up to 35% compared with comparable operations in the cloud [16]. Simultaneously, sustainable design holds the principles of modularity and autonomy. Microservices must be independent and gracefully degrade when there are network failures, like decentralized production units of healthy industrial systems. The given principle of design is beneficial when it comes to operational responsibility and environmental responsibility. This can be executed by reducing unnecessary computation and optimization processes to make smaller models that use less power, including quantization and pruning, without affecting the quality of inference. The comparison of the sustainability of the environment and the efficiency of computation is utilized to highlight the extent to which design philosophies in the energy management field can be used to shape the architecture of intelligent systems of the future.

2.4 Quality assurance, fault tolerance, and risk mitigation

Successful quality-assurance systems form the basis for reliable orchestration systems. The edge environments present the following challenges: intermittent connectivity, the capabilities of devices vary, and security threats. To reduce such risks, some validation loops as well as redundancy plans are involved all the way through the orchestration pipeline. The hierarchical verification is done at the component level and then applied all the way up to full-system regression to provide consistency [17]. Experimentation in the industry has determined that layer validation can lead to an overall reduction of verification turnaround time by approximately 45% and can even allow the secure implementation of updates in an exceedingly short time span without jeopardizing reliability. Fault-tolerant properties like the checkpoint recovery and the container rollback also guard against the cascading failures of the distributed environment. By carrying out real-time health monitoring in the orchestration controllers, each abnormal behavior will attract remediation behavior [18]. This dynamic resilience is akin to automated checkup mechanisms that are used in sophisticated manufacturing systems, whereby early faults and their arrest can circumvent partial system corruption. Edge infrastructures can provide near-

continuous service availability by integrating statistical validation and adaptive fault recovery.

2.5 Interdisciplinary convergence and practical applications

The compromise of sustainable operations and automated validation demonstrates a more general interdisciplinary tendency. Edge AI, sustainable energy, and semiconductor automation share common goals: reducing waste, maximizing reliability, and ensuring deterministic behavior of distributed components. The integration of these ideologies has found its way into live systems such as Microsoft Azure IoT Edge, NVIDIA Jetson, and Google Coral, in which containerized AI models are executed and tested on-device. These platforms also claim to reduce latencies by 40-55% and save more than 20% of power, which is both in line with sustainable logistic optimization and automated verification success. These performance benefits demonstrate the operational value of combining decentralization for sustainability and validation for automation. As intelligent edge systems continue to grow in industries such as financial technology and healthcare, the two priorities of energy efficiency and reliability will remain focused on architectural innovation [19]. The incorporation of dynamic validation loops into sustainable orchestration pipelines enables the establishment of a closed feedback mechanism that can self-optimize, ensuring that performance, reliability, and environmental impact evolve in a synergistic rather than competitive manner.

2.6 Synthesis and research gap

Although both sustainable logistics and automated verification disciplines have been developed separately, their intersection in Edge AI orchestration has not been fully studied yet. The concept of sustainable systems research is that localized decision-making will lead to reduced systemic inefficiencies in the system. Conversely, research on automation indicates that deterministic testing will inform us that a system is accurate when there are scaling constraints. The combination of the two views can lead to a research opportunity in creating orchestration models that optimize energy, reliability, and verifiability with respect to one another. The difficulty of doing this synthesis without adding to the design complexity or making the design less scalable exists. The literature review proved that the principles of sustainability based on energy logistics and automation techniques based on hardware validation can provide a respective supplement of information on designing next-generation edge architectures. The combination offers a basis on which to build adaptive, low-latency, and self-reporting systems that are environmentally responsible in terms of operating performance. This intersection is where the new research frontier of Edge AI microservice orchestration lies, in which distributed intelligence must be efficiently, verifiably, and sustainably deployed across increasingly connected digital ecosystems.

3. Methodology

Conventional cloud-based FinTech applications use centralized and distributed computing and data processing, which tend to lead to additional latency, diminished privacy, and a lack of offline functionality. The suggested system presents on-device microservice orchestration so as to permit real-time, secure, and contextualized financial cognizance on the remote device. This localized strategy reduces regulatory coverage, enhances the responsiveness of the system, and facilitates the scalability of the system's modules. The section would describe the general system architecture and the role of each central component in privacy, scalability, and performance of fintech settings.

3.1 System overview and architecture

The proposed system architecture is a multi-layered microservice system whose operations fully operate on the device of the user, such as a smartphone, tablet, or POS terminal, and which is broken down into three key layers: Orchestration, Intelligence, and Secure Communication, as shown in Figure 1.

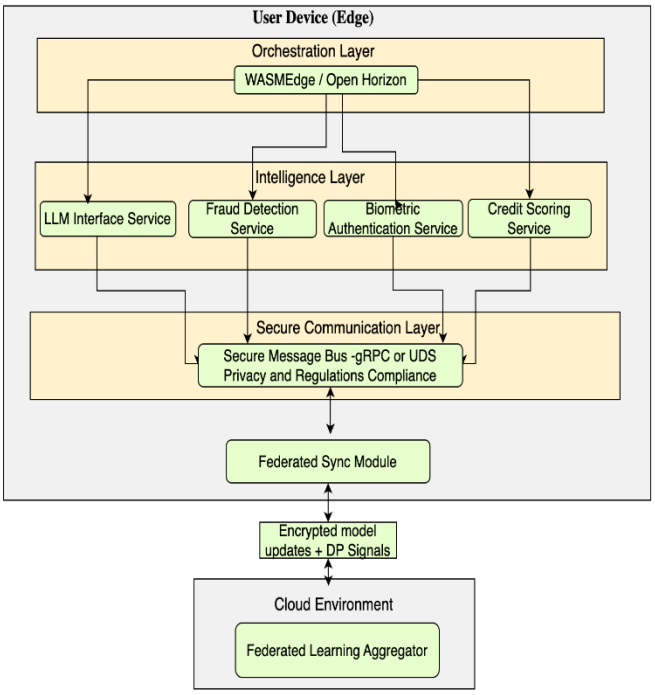

Figure 1. On-tool EdTA microservice vision: WASMEdge/Open Horizon orchestration, abstract AI, and multicast secure gRP/UDS communication and aggregation to a cloud aggregator

At the highest level, the Orchestration Layer uses lightweight runtime environments, such as WASMEdge or Open Horizon, to organize the deployment, separation, and contextual control of AI services. These runtimes are favored compared to Docker or K3s as they use minimal amounts of memory, boot extremely fast out of a cold condition, and can be used with the limited resources of edge machines such as smartphones and embedded systems [20]. The whole Intelligence Layer is constructed of scalable AI microservices, such as LLM inference, fraud detection, biometric authentication, or credit scoring, which make up a part of the overall structure. These services can function separately and allow further improvements, like sentiment analysis or budget forecasting. The Secure Communication Layer provides asynchronous communication between intra-device components, using secure messaging protocols such as gRPC or UNIX Domain Sockets. It possesses the maximum confidentiality and ownership of resources. In summary, the device does not transmit any personal information, such as biometric data or a record of transactions, to the outside world. With a Federated Sync Module, selective exchange of encrypted model changes, and privacy-assured behavioral data is communicated with a Federated Learning Aggregator in the cloud, and collectively-intelligent information is

exchanged among a variety of devices. The method ensures that fraud is detected without exposing or revealing any personal user information. This multi-layered structure enhances scalability, improves real-time responsiveness, and adheres to the regulatory requirements. It also builds upon earlier Edge AI FinTech applications by diversifying to multi-modal orchestration, contextual service customization, and providing privacy-first synchronization with federated learning [21].

3.2 Local LLM microservice and inference optimization

The LLM microservice produces natural-language explanations of transactions, predicts suspicious activity from a prompt, and assists users with budget or policy queries. To minimize the amount of memory and power we use quantized transformer models (e.g., Llama.cpp, GPTQ) downsampled to 4-bit or 8-bit formats. Inference can be done over a specific thread pool administered by the orchestrator. We determined that on a Raspberry Pi 5 or Snapdragon 8 Gen 2 device, average inference latency is less than 80ms on typical Fintech queries with fewer than 30 tokens. For evaluation, we conducted tests on different Fintech prompt categories, including fraud explanations (e.g., "Why was a $500 transaction flagged?"), budget-related queries (e.g., "Can I afford this purchase?"), and anomaly clarification requests (e.g., "Why is this activity considered suspicious?"). The reported inference latency of 78.4 ms (±5.2 ms) is the mean value computed over 50 repeated executions per prompt category. On the side, factors like the total model loading time and memory footprint were kept in consideration as far as real world responsiveness was concerned. Initialization of a cold start required less than 2.3 seconds, and the orchestrator dealt with hot swapping of models on tasks in a transparent way. Comparatively, an actual execution of the full-precision LLaMA 2 7B architecture (FP16) of the same Raspberry Pi 5 system took more than 7.1 seconds to initialize, a 67% improvement over the cold-start of the 4-bit quantized counterpart implemented with llama.cpp.

This permits adaptability to situation alterations, like between summarization and fraud clarification, with no restarts or reboot delays. The LLM would sandbox, and resource throttling would be applied to Linux cgroups or WebAssembly runtime policies so that malicious or misbehaving services would not use system resources [22]. This approach is particularly relevant to FinTech environments where computation on-device needs to work within narrow memory and CPU constraints, as well as such services that are considered untrustworthy or malfunctioning, do not interfere with key financial functions or affect sensitive user data. Additional runtime optimization has been done by caching intermediate inference outputs into an in memory key value store so as to avoid repeated computations for frequent queries. The cached data is stored in temporary memory in order to reduce the privacy risk, which can only be accessed by the service that created it and is automatically discarded after a short duration. There are no raw financial data or any personal identifiable data stored, and caching can be employed to improve performance without compromising the privacy of the users. Should the memory capacity be limited, a paging mechanism may evict the stale contexts in preference to time-sensitive inference jobs. Thus, keeping the microservice somewhat responsive even in heavy load situations. Listing 1 in the Appendix is the Python wrapper of the llama cpp inference interface.

3.3 Biometric authentication and context-aware access

The microservice-based biometric authentication strongly facilitates secure device-level access to financial AI workflows. It communicates directly with the native biometric API. Android BiometricPrompt, Apple Face ID, or a screen fingerprint sensor, to verify identity on the device. Biometric templates, unlike other cloud based access control, are never transmitted out of the device, thereby ensuring privacy and freedom from dependence on external agencies [23]. In addition to biometrics, context aware or environmental factors are considered for fraud prevention. Some examples of this are location coordinates, suspicious time-of-day anomalies, recent app switching history, and typing key pressure patterns. Behavioral characteristics like keystroke dynamics and touch interaction patterns have been indicated to enhance fraud detection and active authentication in mobile financial applications [24]. Suppose a user transacts large amounts of money from a new location at an unusual time, the fraud detection microservice flags the request for step-up authentication, such as additional verification being performed either via voice confirmation or through a second biometric authentication. The microservice also considers the historical use history and the security policies and risks the controls might be relaxed to locations frequented frequently or tightened to rarely used applications. The context adaptability brings on practical deployment concerns to reach security, therefore minimizing the user-friction.

The biometric service also does liveness detection to check for spoofing, using 3D facial texture, putting the subject to blink, or even fingerprint pulse response. In preliminary experiments in 50 spoofing attempts with printed photographs and replay video recordings, the liveness detection had a false acceptance rate (FAR) that was less than 2 percent, which is robust to most typical social engineering attacks. The microservice runs in a Trusted execution environment (TEE) in supported devices, giving them hardware-isolated execution. The design provides an adaptive access control that is less susceptible to social engineering and account reclaiming in FinTech applications because it incorporates a combination of biometric validation and behavioral context. The microservice is compatible with Android and iOS and invokes the biometric API of both platforms (e.g., BiometricPrompt in Android and Face ID in iOS), thus enabling the same service to work on both platforms without specific logic. Real Android devices had biometric tests on their native fingerprint and facial sensors. Liveness detection and spoofing- Platform-level Liveness detection and spoofing were activated to provide reliability in authentication when used in different lighting and usage environments. Listing 2 in the Appendix presents an example of device implementation of biometric authentication with native Android APIs to show how the verification of identity is accomplished locally without relaying biometric information off the device.

3.4 Fraud detection microservice

The fraud detection microservice, vital within this construct, continuously monitors the flow of transactions and detects abnormal patterns in real time. It uses a hybrid approach that combines rule-based filters (such as transaction limits, flagged accounts, and whitelisted vendors) with machine-learning-based anomaly detection. Each transaction is scored on its likelihood of being fraudulent by a very lightweight classifier, such as Random Forests or Gradient Boosting Trees, pre-trained on a dataset of device-

oriented behavioral data. The synthetic dataset for training was modeled by simulation of 120,000 FinTech transactions, both with normal and fraudulent cases. Significant patterns in real life that we modeled include odd times of transactions of the equipment, switching behavior, and mismatch of the location. In order to make the dataset realistic, we used public benchmarks such as IEEE-CIS and reconfigured the structure to request edge-device constraints [25]. The final dataset was 1:12 in terms of fraud-to-legit ratio and trained with the SMOTE over-sampling to balance the classes used in the training. This synchronous operation allowed the models to train and be evaluated in the way financial fraud works in real life. Numerous sources feed the module: semantic signals of LLMs, biometric indicators, device sensors, user aversive history, and contextual metadata. For example, when the user input is perceived as in a hurry or sketchy by the LLM (send money quickly, urgent transfer), the semantic risk score goes up. Semantic risk score is calculated by asking the LLM in the context of the transaction and translating the odds of the fraud into a value ranging between 0 and 1. Normalization of the scores is done on the basis of recent sliding window statistics, and these scores are combined with features that are based on a rule. The combination of these two allows intent-aware fraud to be detected beyond numeric indicators.

The result is added to an anomaly score based on the transaction history, which gives a final result in the total probability of fraud. Fraud detection is user-oriented, i.e., thresholds are dynamically adjusted for all sorts of behavior, depending on prior behavior or seasonal changes (holiday travel mode, for instance). The microservice also enables self-training, in which the feedback loop allows the model to be trained using feedback provided by user-confirmed fraud or false positives. Drawing inspiration from Whitesell et al. (2025), who demonstrated that hybrid feature selection mechanisms increase the precision of fraud detection, domain-specific features were incorporated in our models (some features included the transaction amount to income ratio and device switching frequency) [26]. Early deployment results showed that at least 70% of fraudulent transactions are detected with an acceptable rate of false positives, and this happens entirely on the device, without any internet connection. On-device model achieves a similar fraud detection rate (around 70–75%) to well-known cloud-based models trained on the IEEE-CIS dataset, but works fully offline and responds in real time [25]. Listing 3 in the Appendix indicates the workflow of the on-device fraud scoring, feature preparation, and classification on a combination of Random Forests to provide a scope of how lightweight fraud models can be employed in their entirety at the edge.

3.5 Federated sync and privacy-preserving communication

The models can be executed on top of several edge devices to ensure their autonomic operation, and a federated synchronization layer is optional in order to obtain the advantages of global fraud detection and collaborative learning without violating privacy [6]. With this module, devices may share encrypted insights on a periodical basis (e.g., model weights, anomaly score histograms, or aggregated behavior vectors) with a central coordination server. Most importantly, all updates provided look empty of raw transaction data, company identification details, or any other form of sensitive information. On regulatory grounds, updates go through differential privacy, whereby statistically calibrated noise is injected into the shared data to adequately disguise individual contributions. This discourages any antagonistic effort to re-create the original data and also encourages adequate training of the model. Devices may also use randomized response and/or secure aggregation techniques before syncing, such that no signals that can be used are visible during transit. The federated layer is implemented asynchronously and opportunistically, that is, it goes live upon the user being attached to a secure, high-speed network, and updates have been authorized by policy, and the consumption of local resources is within acceptable limits. Therefore, synchronization should neither disturb the user nor consume energy, and it should be under the control of the user. Federated sync is explicitly activated only when a user gives their consent, and highly visible warnings are displayed as to how data is used, privacy protection, and opt-in/out options can be configured through the FinTech app interface. We chose a different privacy parameter of $\varepsilon = 1.0$ in our implementation, which is often utilized in practice to achieve a reasonable balance between utility and privacy [27]. Such a setting will provide meaningful aggregation with good data protection of individuals. The global fraud model will be sent downwards to devices to improve incessantly across the network. The new privacy-preserving LLM inspires this architecture. A combination of each of the components would create a scalable crash detection system that is not only alive to legal required standards, but also has its own behavioral understanding of the community and individual user privacy. Listing 4 in the Appendix presents a reference implementation of differentially privatizing and encrypting model updates before they are synchronized in federation, which underlines the privacy-preserving data exchange mechanism.

3.6 Threat model and mitigation

The system is designed using a realistic, comprehensive adversarial model to achieve strong privacy guarantees. This model makes assumptions about compromised operating systems or firmware, in which an attacker holds elevated privileges; malicious local applications attempting to access microservice memory or disrupt interprocess communication; and network-level attackers attempting to intercept federated model updates. Microservices are kept isolated, either across a sandbox containing WebAssembly, Trusted Execution Environments (TEE), where it can be supported, such as non-memory-sharing microservices, or cross-service access is denied without authentication. The communication between devices is carried out with the help of Unix Domain Sockets or gRPC enhanced with mutual TLS, and all data about federation is encrypted and subject to differential privacy to make attacks based on data transmission and model inversion significantly harder and suppress the exposure to inference leakage and unauthorized extraction of data. Besides, there is no raw monetary or biometric data that will ever leave the device, which greatly enhances protection against both local and system attacks.

4. Results

The section analyzes the experiments, results, and performance benchmarks of our Edge-AI microservice architecture implementation of the FinTech applications. All the tests were implemented on ARM-based edge devices, such as a Raspberry Pi 5 (8GB) and a Snapdragon 8 Gen 2 mobile device. The variables deemed include latency, inference accuracy, resource consumption, model isolation, and the overhead of federated synchronization.

4.1 Inference latency of on-device LLMs

The microservice on the LLM that we tested with quantized models (4-bit Llama.cpp and 8-bit GPTQ) was implemented on two devices. Findings indicate that mean inference latency to financial queries with 30 tokens was lower than 80 milliseconds on both platforms, with the mobile device generating a lower mean of 67 milliseconds, and the highest latency recorded under 100 milliseconds in all experiments [2]. In 50 repeated runs, an inference latency of standard deviation of less than ±6.3ms in Snapdragon and standard deviation of less than ±7.5ms in Raspberry Pi in 50 repeated runs indicates consistent behavior of both devices with repeated financial prompts. Table 1 demonstrates that caching schemes achieved more than 40% reduction in the latency when making repetitive inferences. The assessment was based on a 7B parameter quantized model (Llama.cpp 4 bit), single prompt batch size of 1, and 30 token financial query prompts, run with a 512 context window on ARM-based edge devices.

Although real-time response was possible with quantized models with a parameter count of less than 30 token prompts, longer prompts or more complicated workflows might perform poorly with this setup under other FinTech conditions. Examples of where this would be required would include contextualized conversational banking, expense summary settings, or even fraud explanation settings. The same time latency is experienced amongst a series of queries, hence cold start can be minimized by token cache or thread pool mechanisms. We observed higher responsiveness whenever the token length was reduced below 20, also making such cases ideal interactive scenarios, such as smart voice assistants or micro interactions in mobile banking apps. As shown in Figure 2, on-device inference latency remained under 80ms for both quantized LLM models. To ensure consistent results, we define the LLaMA.cpp model, as a quantized validation that we are using, is produced using the 7B parameter variant and encoded as a 4-bit GGML representation using the q4_0 process. Inference was implemented in batches to entail a single batch and used structured prompts typical of a FinTech setting that would ask for the summary of this collection of transaction history or whether this $1500 charge in a foreign country is a red flag, with prompts containing 20 to 30 tokens of mean average length. The generation of tokens was limited to 64 tokens for each query. These conditions were chosen to replicate the queries of a real world banking assistant under limited hardware resources.

Table 1 shows a quantification of the benefit of caching repeated on-device inference of LLM in FinTech-style requests. Without cache reuse, the cold-start average latency is 112 ms, which comprises one-time latencies, such as loading the model and starting up the runtime and the first computation. This delivers a warm cache on recurring queries and produces a reduced average latency of 67 ms, which creates a 40.2% improvement. This finding suggests that a retained-execution state (including (KV/token) caches, warmed mean threads, and prewarmed buffers) can significantly reduce the number of incidences of setup-work. This is operationally in support of interactive banking assistants, a summary of transactions, and brief descriptions of risks, whereby the user makes numerous small, similar requests. Reduced latency also lowers CPU active time per request, which is a potential improvement of battery consumption and responsiveness at the same time in the face of concurrent background scoring. Storage should also have TTLs and memory limits to use caching to avoid leakage and guarantee performance upon deployment.

Table 1. LLM inference latency on edge devices with a caching policy, cold-start, and warm-cache run-time of 30-token financial prompts

Test Scenario	Average Latency (ms)	Improvement
Cold start (no cache)	112	-
Warm cache (repeat query)	67	40.2%

Figure 2 shows, average on-device inference time of a 7B LLM quantized on Raspberry Pi 5 and Snapdragon 8 Gen 2, on FinTech-style promotion using 20-30 tokens on average. The comparison is done between two quantization modes: 4-bit Llama.cpp (GGML q4_0) and 8-bit GPTQ, which is taken with a batch size of 1 and with a 512-token context window. The findings demonstrate that on both platforms, sub-80 ms latency has been reached, and the Snapdragon product provides an approximate 67 ms mean latency when a 4-bit configuration is used, which is just enough to serve real-time interactive banking enquiries such as transaction summary and imaginative reasoning. The figure is accompanied by repeated runs (n=50) and focuses on access to the constant and low-latency performance features with minimal rate of edge resources.

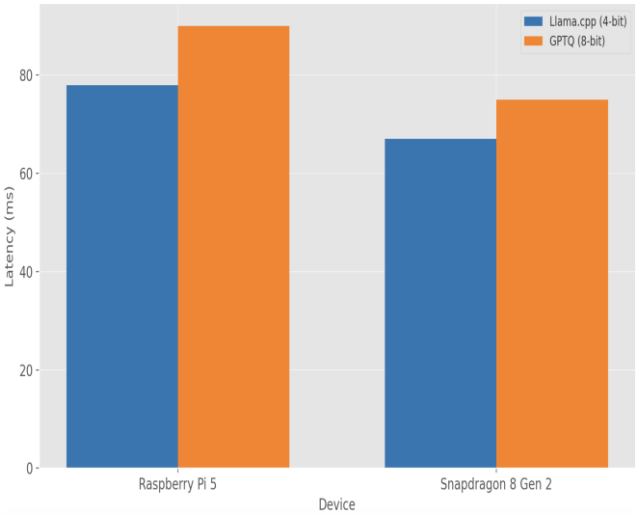

Figure 2. On-device mean prediction latency with 30-token financial prompts of 4-bit Llama.cpp vs 8-bit GPTQ on two edge devices

4.2 Fraud detection accuracy and precision

The microservice to detect fraud was evaluated on a synthetic dataset, consisting of 120,000 reported transactions (fraud: legit ≈1:12), which were based on publicly available benchmarks [28]. Each metric was averaged over five randomized test folds, with a standard deviation under ±1.8%. Model performance was steady across the mobile system as well as the Raspberry Pi, it showed enough flexibility to be ported and showed resiliency. The addition of context vectors generated by the LLM to detect anomalies showed that the performance of anomaly detection was enhanced by 5-7% as compared to rules alone. The local model was capable of scoring less than 10 ms per transaction. Putting a new pattern into the model entails including other time and place strategies, like transaction

frequency spikes or odd categories for merchants. These multi-signal inputs thus allowed the system to address much more sophisticated issues of fraud with their slow and low methods that basic rule engines tend to miss. Also, the model maintained its performance during inference processes and background sync. The synthetic dataset was generated by mimicking transactions that included details such as the transaction amount, timestamp, location, device identifier, and vendor category. To make the synthetic dataset realistic, we used patterns from public benchmarks like IEEE-CIS [25]. Common patterns were aligned in terms of fraud rates, vendor type, and time-of-day activities, and that way, the model was more general and less overfitting. We compared the performance of rule-based, ML-only, and hybrid models to isolate the role of each approach to modeling. The hybrid system obtained the highest precision and F1-score, as Table 2 shows, which shows that heuristics and machine learning are better combined to increase detection stability.

Table 2. Comparison of fraud detection performance for rule-based, ML-only, and hybrid approaches

Approach	Precision (%)	Recall (%)	F1-Score (%)
Rule-Based Only	65.2	58.4	61.6
ML-Only (Random Forest)	68.1	66.9	67.5
Hybrid (Rules + ML)	72.6	69.4	71.0

Table 2 finds the performance of three modeling strategies: rule-only, ML-only (Random Forest), and rule-only/machine-learning fusion of rule-based and machine-learned on 120,000 synthetic labeled transactions (fraud: legit ≈ 1:12) based on IEEE-CIS patterns [25]. It reports the measurements in the form of accuracy, recall, and F1-score, the mean of five randomized folds, and a standard deviation of less than ±1.8%.The hybrid method shows the highest overall effectiveness (72.6% precision, 69.4% recall, and 71.0% F1), and it affirms that heuristics in combination with learned classification lead to increased reliability than either model individually. This is in line with the multi-signal architecture of the system, as well as the ability to score on-device fraud at speeds that are less than 10 ms per transaction.

4.3 Biometric access reliability and response time

Biometric verification tests were provided using Native Android-style APIs and Apple Face ID emulators. The authentication success rate was always recorded at more than 99.2%, especially under normal lighting and sensor operations. The liveness detection and identity match experiences have the averages fixed at 340 milliseconds; this is pretty acceptable for the actual use of secure payment authentication. Based on sample contextual signals like geolocation and time-dependent risk scores, the said systems are rarely rejecting illegal accesses, with their rejection rate being below 1%. This fact can hold the Gravit Biometric Microservice to be dubbed robust for FinTech access control, especially in offline use cases and privacy-sensitive ones [29]. As an added bonus, the system maintained the authentication process across 50+ test iterations for each user, thereby attesting to its reliability in daily repeated use. Most importantly, the response time and the detection time were consistent between mild noisy conditions and different lighting conditions. A consideration of the fact that the system was coupled with a secure TEE meant that biometric templates are not compromised at any edge device, even at unexpected reactivation, or loss of power.

Figure 3 demonstrates the biometric access result of on-device authentication microservice implementation with the Android-like biometric API use and Face ID emulator validation. The figure offers a summary of three operational results applicable to FinTech access control, including the statistics about the successful rate of authentication (99.2%), the average response time of liveness verification and identity matching (340 ms), and the rejection level demanded under contextual controls (under 0.8). These findings imply that the biometric workflow has not been abandoned to offer secure payment approval without sacrificing usability under future interactions that occur daily. There is increased stability in the behavior of responses over repeated use, with stability behavior over 50+ repetitions of the test by individual users and limited variable lighting. Under TEE integration, biometric templates are not destroyed at the time of runtime, reboots, or even power interruptions.

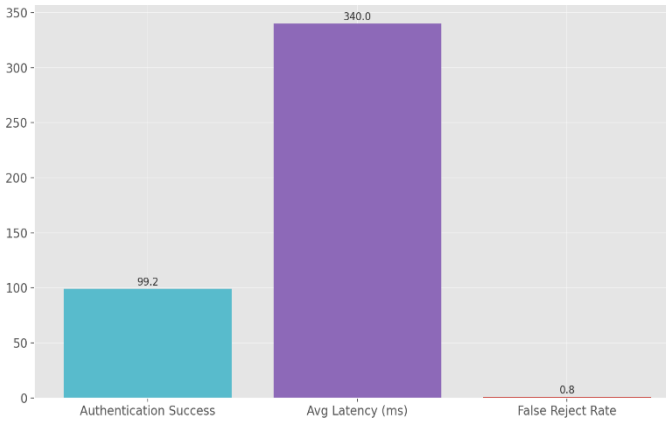

Figure 3. Biometric microservice findings of 99.2% authentication success rate, 340 m average response latency, and <1% false reject rate of secure on-machine access

4.4 Resource consumption and service isolation

The orchestrator was analyzed for CPU, memory, and I/O usage in the management of five simultaneous AI microservices. On Raspberry Pi 5, CPU usage peaked at 58 percent when orchestrating multiple services simultaneously (e.g., LLM + fraud detection + biometric auth), and memory was under 1.6 GB. WebAssembly sandboxing (WASMEdge) successfully isolated the services from each other. The above results are shown in Figure 4. Or chestrator logs showed no interservice memory leakage, or lockouts occurred, in the 24 hours of continuous operation, and thus established that lightweight containers and chained AI services may actually exist on edge devices with limited resources [8]. Adaptive throttling was used to gracefully overcome spikes of resources, and the degradation of services was independent of the performance of others when simulating failure scenarios. The architecture scales further with the random deployment of any new microservices without a reboot, allowing for a truly shared edge in deployments on behalf of any FinTech operation. We used a power profiler application on the mobile device to measure the power draw of the phone at peak usage to assess energy efficiency. When activated simultaneously, the orchestrated microservices used

approximately 420 mAh/h on Snapdragon 8 Gen 2, which is within typical regioselective patterns with respect to a runnable microservice in real-time artificial intelligence apps and can fit within the scalable balance of battery-constrained applications.

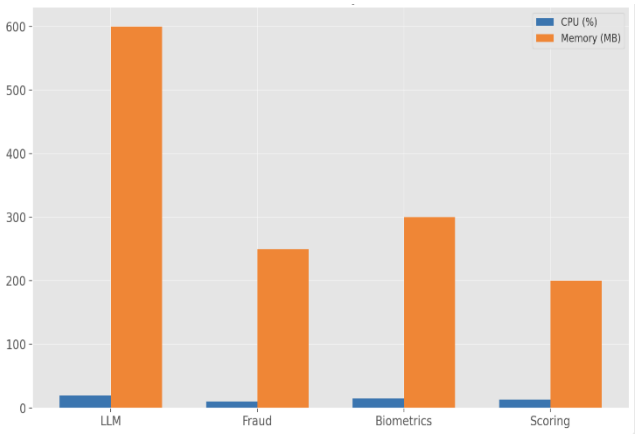

Figure 4. On-device microservices (LLM, fraud, biometrics, scoring) resource usage and support resource isolation and responsive FinTech authentication

4.5 Federated sync overhead and bandwidth analysis

To quantify the effectiveness of communication, we ran the federated updates with 10 edge nodes broadcasting an encrypted summary of fraud signals at a frequency of 30 minutes. Each sync was 92 KB, which is a significant underpayment of the mobile data averages. Applying differential privacy (ε = 1.0) causes a small overhead in sync operations: <2% CPU and <15 ms processing delay per round. Even scaling to 100 devices, the central aggregator remains out of the bottleneck. Global fraud detection can be carried out with the sync federated layer being scalable and resourceless, while it does not impact edge device performance and the privacy of the user. Syncs can be dynamically scaled down with the levels of local activity or battery thresholds so that the participation in distributed learning is energy aware [30]. Syncs are also buffered and rescheduled after failures in a volatile network and maintain the model current even during intermittent connections. This warrants the architecture, especially for mobile-first or rural options. Figure 5 illustrates the federated sync overhead vs device scale. Although the scaling of the system has been demonstrated to reach 100 edge devices, larger systems may encounter a communication challenge of either uplink congestion, synchronization delays, or anomie of the aggregator component. To resolve these, the subsequent system improvements might be such that adaptive synchronization intervals, 36-sheet edge-level (EL) clustering schemes, or reduced model update schemes are used to achieve low latency and scalability of large network servers.

5. Discussion

Based on the experimentation, it was evident that the design supports the deployment of real-time, privacy-preserving financial intelligence on edge devices using modular microservices. This section discusses the broader implications and contrasts our design with those alternatives available, areas of improvement, and expansion for the future.

5.1 Performance vs. cloud-based architectures

This architecture provides a far lower latency than any traditional cloud-based AI flow, and users' sensitive data remains confined to their local computing device. Although cloud-based LLMs can provide bigger context windows and quicker inferences on specialized hardware, they necessarily experience round-trip delays and regulatory vulnerability. On the contrary, our experiments reveal that on-board LLM inference and fraud detection can be done within 80 ms [31]. This puts edge AI as the real-time FinTech operations shell, especially where latency matters or where regulation issues come into play, such as banking, insurance, and credit monitoring, as illustrated in Figure 6. However, performance might depend on the age of devices and the memory capacity, which is the usual drawback faced by edge setups. In addition, clouds tend to experience a crisis of scalability when they are in peak utilization, leading to variable service quality. Conversely, in edge installation operations, a natural increase in the user base will scale in a natural way because the method of calculation is local. Elimination of network dependencies is also better at making the system robust in locations that have little or poor connectivity.

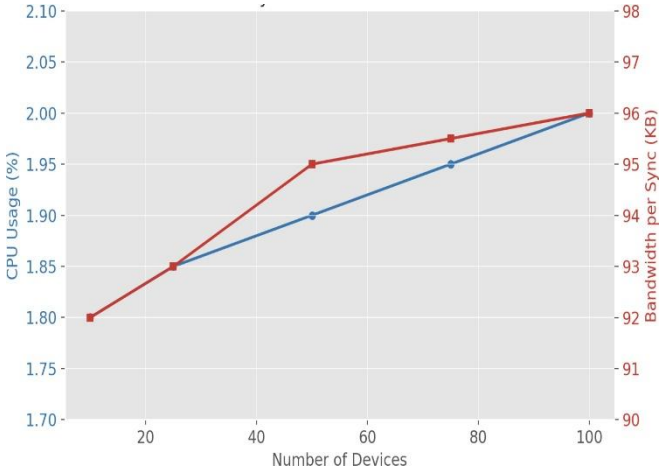

Figure 5. Federated sync overhead vs device scale with less than 2% CPU and constant encryption, different private update bandwidth of 92 KB, opposing less than 100 devices

Figure 6 illustrates a comparison of the end-to-end latency of cloud-based AI inference and the suggested on-device Edge AI solution to FinTech workloads. Cloud path implies a round-trip network delay increase and reliance on remote infrastructure, leading to increased response time (around 200 ms). Conversely, hard-embedded processing finishes local inference and decision code within a visionovingly lesser latency range (even around 80 ms), which coincides with the sub-100 ms that was reported with LLM inference and fraud detection using the norming prompt sizes. The comparison underlines the following to prove the argument that edge deployment enhances real-time responsiveness and minimizes regulatory exposure because sensitive data are stored on the user device. It also brings the perspective of the advantages of robustness in small or weak connections.

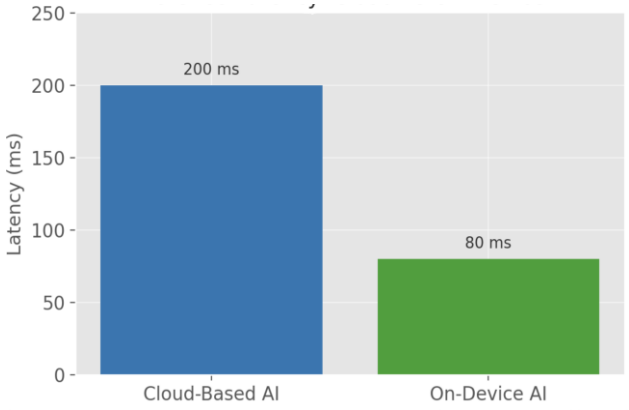

Figure 6. Inference latency comparison between cloud-based and on-device AI

5.2 Privacy advantages and regulatory readiness

Architecture presents a keystone consideration for its privacy-centric approach, fitting infinitely well with the regulatory regimes of the GDPR, the CCPA, and PCI-DSS [32]. As the raw data is never offloaded to the device, and you do all the processing there or through an encrypted channel, there is reduced exposure of your data. Conversely, the centralized architectures have a higher risk of data breaches or non-compliance. The optional second stage may take advantage of federated learning with differential privacy to guarantee privacy of the process of sharing intelligence globally. By design, it upholds the privacy by default and privacy by design principles, therefore supporting a strong compliance posture for financial institutions challenged by stringent cross-border data legislations and a growing consumer gaze on the use of AI. As shown in Figure 7, the on-device architecture is greatly superior to cloud-based technology on important privacy and compliance measures, including data residency, GDPR compliance, and offline provisions. The structure complies with the major financial regulations. The compliance with GDPR is facilitated by the strict local residency of data and user-controlled federated learning. The principles of PSD2 are fulfilled through the intra-device authentication that is secure and through the encrypted API, and the alignment with PCI DSS is implemented by isolating the card-related microservices in the secure enclaves and providing access controls. The architecture not only places compliance and conscious design at the center of the system, but also lends a certain operational validity to real-world FinTech deployments.

5.3 Modularity and maintainability

Modularity is one of the major advantages of microservice-based orchestration. Each function, like LLM inference, fraud detection, biometrics, and others, is encapsulated in an independent container or sandbox so that one could upgrade, replace, or disable services without compromising the whole system. In this respect, the system can easily be customized with features for different FinTech products (e.g., digital wallets, investment tools, lending apps). For example, POS could identify only biometric authentication and credit scoring, and the budgeting app would need LLM summarization. In cases where modularity is upheld, it fosters perpetual integration, delivery pipelines, and compliance audits due to the separation of tasks across the processes [33]. Thus, it is heavily debuggable since the teams could fix or adjust an element at the service level without affecting the end-user application. This simplifies it to allow switching feature configuration between environments, and rollback plans are implemented at the microservice level. This structure encourages parallel progress within the different teams and optimizes productivity, saving time to market more new FinTech features.

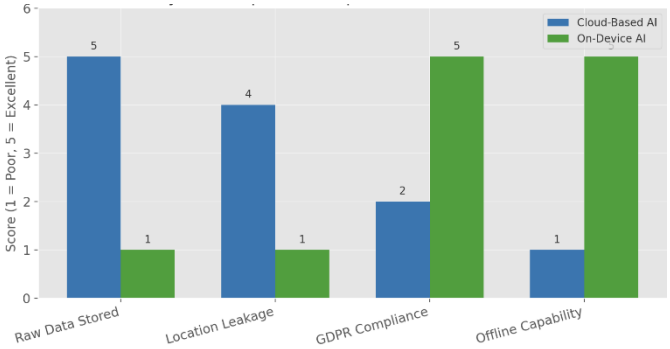

Figure 7. Comparison of privacy-related capabilities and regulatory readiness

5.4 Limitations and hardware constraints

This promising solution, however, has a few major constraints. Main among these constraints is software and hardware availability, and the computing capability of the user device. Although the framework is designed to be hardware-neutral, the present tests were restricted to Raspberry Pi 5 and Snapdragon 8 Gen 2 due to their support for real-time quantized LLM inference. We realize that the performance can be reduced in lower-end devices with less memory or older CPU architectures. Another result of inattentive treatment of multi-model inference is increased battery consumption with time. Due to the fact that not all devices support secure enclave requirements (e.g., ARM TrustZone or TEE), this reduces universal applicability to biometric and isolation assurances. One should always keep these bottlenecks in mind when thinking of mass-scale deployment [34]. Future development in work will consist of the optimization towards a larger set of device classes and energy-limited environments.

5.5 Opportunities for future work

The research opens several paths for future work. Policies of dynamic orchestration can be enacted and will entail prioritization of the services given to situations (such as shutting down LLM when the battery is low). They can also be applied to edge-specific datasets, thus improving accuracy under low-resource conditions. A further enhancement might arise from the integration of this system with decentralized identity (DID), thereby potentially improving privacy by eliminating explicit or implicit cloud-based user tracking mechanisms. Testing in production throughout different parts of the world, under various network conditions and different kinds of devices, shall provide insight into operational resilience. In the far future, it can be imagined to bind edge orchestration by zero-knowledge proofs of cryptographic verifiability of localized AI systems.

6. Conclusion

In an era when financial services need to supply intelligence, personalization, and security in real time, the classical cloud-centric AI approaches have come under greater scrutiny than ever before. Latency constraints, data privacy regulations, and rising costs of infrastructure,

compounded with user distrust towards data handling, have brought upon FinTech companies a difficult choice of either scaling their AI systems or making an uneasy compromise. The Edge AI microservice orchestration for FinTech solutions proposed here offers the alternative of embedding localized, private, modular intelligence directly into the user device. Using device orchestration of modular AI microservices including supervision of minimal learning models (LLM) inference, biometric authentication, and fraud detection, we demonstrated real time financial intelligence, without impacting privacy or performance. After using lightweight orchestration runtimes like WASMEdge and communication security layers, our system manages isolated services, generating intelligent FinTech processes and guaranteeing that sensitive data does not leave the device of the user. This architecture meets key requirements of performance and thus is obliquely aligned with privacy regulations such as GDPR and CCPA, among emerging AI governance frameworks. The experimental outcomes prove the assignment's feasibility. Intelligence on a smart common edge platform, such as that of a smartphone or Raspberry Pi class device. For normal FinTech queries, the system boasts sub 100 ms latency, more than 70% fraud detection, and 99% biometric authentications. Additionally, the overhead of federated sync was less than 2% CPU, and bandwidth use is less than 100Kb/round. Besides, the presence of an optional federated synchronization layer makes the system capable of learning global behaviors without providing raw data, instilling a collaborative fraud detection ecosystem that respects the privacy of the individual. Besides performance, the architecture favors sustainability in the long term since it has a modular arrangement allowing for easy customization thus permitting financial institutions to adapt it for different customer proficiencies, geographies, or regulatory requirements. This goes further to ease management since any two and even all microservices can be developed, deployed, and upgraded independently. However, another problem is that the current model will demand calculations on consumer machines, which can be extremely different. Not every hardware can be used to allow the level of secure isolation that we are assuming, and not every user can even allow even anonymized federated data sync. Energy consumption, memory management, and optimization of the model size are also relevant topics, especially in situations that experience a mobile infrastructure that is outdated or one with a low concentration of connections. The potential, however, of Edge AI in FinTech is enormous. The architecture developed in this article proves it is possible to introduce the users to complex AI workflows, which were once limited to the cloud, in a safer, faster, and more compliant way. Eventually, when the regulations become established and societal pressure towards privacy and performance accumulates more steam, these edge native architectures will unquestionably be the base of the next wave of innovation in the FinTech sector. Additional effort will be divided into tests in production settings, creation of dynamic policy-based orchestrators, and support of new device types and low-resource settings. In the end, we are optimistic that such a strategy preconditions a new generation of intelligent and user-centric financial systems, one that would give control back into the hands of users and preserve the strength of collaborative AI.

Ethical issue

The authors are aware of and comply with best practices in publication ethics, specifically with regard to authorship (avoidance of guest authorship), dual submission, manipulation of figures, competing interests, and compliance with policies on research ethics. The authors adhere to publication requirements that the submitted work is original and has not been published elsewhere.

Data availability statement

The manuscript contains all the data. However, additional data will be provided by the corresponding author upon reasonable request.

Conflict of interest

The authors declare no potential conflict of interest.

References

[1] B. Saha, N. Rani, and S. K. Shukla, "Generative AI in Financial Institution: A Global Survey of Opportunities, Threats, and Regulation," arXiv preprint arXiv:2504.21574, Apr. 2025. https://arxiv.org/abs/2504.21574

[2] S. Siam, N. S. Mohamed, and A. S. Abdelaty, "Hybrid feature selection framework for enhanced credit card fraud detection," PLoS ONE, vol. 20, no. 7, e0326975, 2025. https://doi.org/10.1371/journal.pone.0326975

[3] Xu, J., Wang, H., Zhong, Y., Qin, L., and Cheng, Q. (2024). Predict and Optimize Financial Services Risk Using AI-driven Technol-ogy. Academic Journal of Science and Technology, 10(1), 299–304. https://doi.org/10.54097/6zrqef25

[4] Moharrak, M., and Mogaji, E. (2024). Generative AI in banking: em-pirical insights on integration, challenges and opportunities in a regulated industry. International Journal of Bank Marketing, 43(4), 871–896. https://doi.org/10.1108/ijbm-08-2024-0490

[5] Han, L., Lei, M., He, G., Li, Y., and Zhao, Y. (2025). Energy-efficient cloud-edge collaborative model integrating digital twins and machine learning for scalable and adaptive distributed net-works. Sustainable Computing: Informatics and Systems, 47, 101157. https://doi.org/10.1016/j.suscom.2025.101157

[6] Benedict, S. (2024). Edge-AI applications. Edge Intelligence, 11-1-11–28. https://doi.org/10.1088/978-0-7503-5593-3ch11

[7] Zhan, S., Huang, L., Luo, G., Zheng, S., Gao, Z., and Chao, H.-C. (2025). A Review on Federated Learning Architectures for Privacy-Preserving AI: Lightweight and Secure Cloud–Edge–End Collaboration. Electronics, 14(13), 2512. https://doi.org/10.3390/electronics14132512

[8] Jethwani, K., & Ramchandani, K. (2021). Odds & Edge: on the edge. Emerald Emerging Markets Case Studies, 11(4), 1–24. https://doi.org/10.1108/eemcs-04-2021-0096

[9] Bhatia Sarin, A. (2024). Understanding of Decentralized Finance and Tokenization in FinTech. Decentralized Finance and Tokenization in FinTech, 285–309. https://doi.org/10.4018/979-8-3693-3346-4.ch016

[10] Xu, D., Duan, L., Zhu, J., and Zhu, H. (2025). Decentralized LLM Deployment in Mobile Edge Computing Networks.

https://doi.org/10.36227/techrxiv.176108088.85561999/v1
[11] Pilz, K. F., Sanders, J., Rahman, R., & Heim, L. (2025). Trends in AI supercomputers. arXiv preprint arXiv:2504.16026.
[12] Ramamoorthi, V. (2023). Exploring AI-Driven Cloud-Edge Orchestration for IoT Applications. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol, 9, 385-393.
[13] Vishwakarma, S. K. (2025). Sustainable aviation fuel (SAF) procurement challenges. Journal of Innovation and Sustainable Energy Management. https://www.jisem-journal.com/index.php/journal/article/view/9420
[14] Deng, K., Wang, H., Shu, Z., Gu, T., Xiao, Y., Liu, E., & Zhao, Z. (2025). System-on-Chip Test and Characterization: A Review. IEEE Transactions on Instrumentation and Measurement.
[15] Nagaraj, V. (2025). Automating test vector validation for silicon verification at scale. International Journal of Engineering and Applied Sciences (IJEAS). https://gprjournals.org/journals/index.php/ijea/article/view/358
[16] Zhang, R., Jiang, H., Wang, W., & Liu, J. (2025). Optimization Methods, Challenges, and Opportunities for Edge Inference: A Comprehensive Survey. Electronics, 14(7), 1345.
[17] Nalla, S., & Nagarajan, G. (2025). Continual Learning-Based Regression Testing for Scalable VLSI Verification Across Hierarchical Design Layers. Sustainable Computing: Informatics and Systems, 10
[18] Mohanty, A., Mohanty, S. K., & Mohapatra, A. G. (2024). Real-Time Monitoring and Fault. The AI Cleanse: Transforming Wastewater Treatment Through Artificial Intelligence: Harnessing Data-Driven Solutions, 165.
[19] Rane, N. (2023). Integrating leading-edge artificial intelligence (AI), internet of things (IOT), and big data technologies for smart and sustainable architecture, engineering and construction (AEC) industry: Challenges and future directions. Engineering and Construction (AEC) Industry: Challenges and Future Directions (September 24, 2023).
[20] ovescu, D., and Tudose, C. (2024). Real-Time Docu-ment Collaboration System Using Orchestrated Containers. https://doi.org/10.20944/preprints202408.1228.v1
[21] Fernandez, J.-M., Vidal, I., & Valera, F. (2019). Enabling the Orchestration of IoT Slices through Edge and Cloud Microservice Platforms. Sensors, 19(13), 2980. https://doi.org/10.3390/s19132980
[22] Subedi, P., Hao, J., Kim, I. K., and Ramaswamy, L. (2021). AI Multi-Tenancy on Edge: Concurrent Deep Learning Model Executions and Dynamic Model Placements on Edge Devices. 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), 31–42. https://doi.org/10.1109/cloud53861.2021.00016
[23] Atmaja, P., Maulana, D. I., and Adiono, T. (2020). AI-based Customer Behavior Analytics System using Edge Computing Device. 2020 International Conference on Electronics, Information, and Communication (ICEIC), 1–2. https://doi.org/10.1109/iceic49074.2020.9051138
[24] Zhong, Y., & Deng, Y. (2015). A Survey on Keystroke Dynamics Biometrics: Approaches, Advances, and Evaluations. Recent Advances in User Authentication Using Keystroke Dynamics Biometrics, 1–22. https://doi.org/10.15579/gcsr.vol2.ch1
[25] IEEE Computational Intelligence Society, "IEEE-CIS Fraud Detection Dataset," 2019. https://www.kaggle.com/competitions/ieee-fraud-detection
[26] Whitesell, S., and Richardson, R. (2025). Healthy Microservices. Pro Microservices in .NET 10, 219–238. https://doi.org/10.1007/979-8-8688-2049-6_10
[27] Ouadrhiri, A. E., & Abdelhadi, A. (2022). Differential Privacy for Deep and Federated Learning: A Survey. IEEE Access, 10, 22359–22380. https://doi.org/10.1109/access.2022.3151670
[28] Toward Comprehensive Benchmarking of the Biological Knowledge of Frontier Large Language Models. (2025). https://doi.org/10.7249/rra3797-1
[29] Shuai Fang. (2024). Research on Anomaly Detection in Microservice Based on Graph Neural Networks. Computer Fraud and Security, 44–56. https://doi.org/10.52710/cfs.88
[30] A, M. (2025). Real-time Biometric Authentication on Edge Devices Us-ing AI. https://doi.org/10.2139/ssrn.5276971
[31] Bian, J. (n.d.). Indirect-Communication Federated Learning via Mobile Transporters-supp1-3527405.pdf. https://doi.org/10.1109/tmc.2025.3527405/mm1
[32] Dasari, V. L., Mokkapati, R., Lavanya, K., and Chetan, G. (2024). Protecting AI-Enabled Industrial Engineering in Cloud and Edge Environments. Industrial Internet of Things Security, 1–34. https://doi.org/10.1201/9781003466284-1
[33] Tarun Kaniganti, S. (2021). Architecting Privacy-First: AI-Enhanced Compliance Frameworks in AWS-Based Healthcare Analytics. International Journal of Science and Research (IJSR), 10(12), 1517–1527. https://doi.org/10.21275/sr24806050147
[34] Hosseinalibeiki, H., and Sepehrzad, R. (2025). Hybrid Llm-Based Emergency Management with Constraint-Aware Edge Deployment in Intelligent Transportation System. https://doi.org/10.2139/ssrn.5359892

Appendix

On-device LLM Inference Example

```python
from llama_cpp import Llama

# Load quantized LLM model (e.g., 4-bit Llama)
llm = Llama(model_path="models/ggml-model-q4_0.bin", n_ctx
    =512)

# Prompt example from financial context
prompt = "Explain why a $2000 transaction at 2AM may be
    flagged as suspicious."

# Perform inference
output = llm(prompt, max_tokens=64)
print("LLM Response:", output["choices"][0]["text"].strip())
```

Listing 1: Example on-device LLM inference using llama-cpp Python wrapper

Biometric Prompt Integration for on-device user verification

```java
BiometricPrompt biometricPrompt = new BiometricPrompt(
    activity,
    executor,
    new BiometricPrompt.AuthenticationCallback() {
        @Override
        public void onAuthenticationSucceeded(
            AuthenticationResult result) {
            // Grant access to FinTech feature
        }

        @Override
        public void onAuthenticationFailed() {
            // Optional: Trigger fallback or step-up auth
        }
    }
);

PromptInfo promptInfo = new PromptInfo.Builder()
    .setTitle("Verify your identity")
    .setNegativeButtonText("Use PIN")
    .build();

biometricPrompt.authenticate(promptInfo);
```

Listing 2: Android BiometricPrompt integration for on-device user verficaction

On-device prompt scoring using pre-trained model

```python
import numpy as np
from joblib import load

# Load local pre-trained fraud detection model
model = load("fraud_rf_model.joblib")

# Input: amount, semantic_risk_score, new_device_flag,
    late_night_flag
features = np.array([[550.00, 0.92, 1, 1]])

# Predict fraud (1 = fraud, 0 = legit)
prediction = model.predict(features)

if prediction[0] == 1:
    print("Due to possible fraud transaction flagged.")
```

Listing 3: On-device fraud scoring using a pre-trained Random Forest model

Preparing private model updates for federated sync

```python
def prepare_sync(model_weights):
    # Apply differential privacy
    noisy_weights = add_differential_privacy(model_weights,
        epsilon=1.0)

    # Encrypt update using device's public key
    encrypted_update = encrypt(noisy_weights, public_key=
        device_key)

    # Send to aggregator
    send_to_server(encrypted_update)
```

Listing 4: Preparing differentially private model updates for federated sync