International Journal of Emerging Research in Engineering and Technology
Pearl Blue Research Group| ICRCEDA2025-Conference Proceeding
ISSN: 3050-922X | https://doi.org/10.63282/3050-922X.ICRCEDA25-143

Original Article

Explainable and Context-Aware Financial Nudges via Event-Driven Microservices

Anuraga Prasanna Mandaleeka
Member - IEEE, California, USA.

Abstract - *As financial decision-making becomes increasingly automated and personalized, users demand not only timely and relevant financial recommendations, but also transparency in how those suggestions are generated. This paper presents a novel microservices-based framework that delivers context-aware financial nudges enhanced by explainable AI (XAI), designed for real-time deployment in modern fintech applications. The proposed system leverages event-driven microservices to continuously ingest and process multi-modal data streams including transaction history, geo location, time-of-day patterns, and behavioral signals to deliver actionable insights such as spending warnings, savings opportunities, and goal reminders. Each microservice is responsible for a modular task such as context classification, recommendation generation, or explanation rendering. What distinguishes this work is the integration of explainability modules using interpretable AI techniques (e.g., SHAP, counterfactuals, rule-based trees) embedded within each service. This enables the system to answer, "Why am I getting this nudge?" in natural language, thereby fostering user trust and behavioral compliance. The framework is evaluated on synthetic and anonymized financial datasets to simulate diverse user behaviors. Results demonstrate the effectiveness of contextual triggers (e.g., time, location, prior habits) in increasing user engagement, while explainability boosts users' perceived relevance and trust in the system. The architecture adheres to principles of modularity, fault isolation, and data minimization, making it suitable for deployment in privacy-sensitive financial environments. This research bridges the gap between intelligent personalization and transparent automation in fintech, paving the way for ethical, user-centered financial advisory systems.*

Keywords - *Context-Aware Nudges, Event-Driven Microservices Explainable AI, Financial Nudges, Kafka, SHAP, XAI.*

1. Introduction and Motivation

In today's fast-paced digital environment, users are constantly bombarded with notifications, messages, and alerts from numerous apps and services. Compounding this challenge is the well-documented decline in human attention spans, which have dropped from an average of 12 seconds in 2000 to just 8 seconds by 2015shorter than that of a goldfish. Within this context, delivering timely and engaging personalized financial alerts becomes not only challenging but essential. Such alerts play a critical role in helping users avoid overspending, identify risky financial behavior, and stay aligned with their budgeting goals. However, generic notifications like "You've spent more than usual" or "Try saving more" often fail to resonate, lacking the specificity and clarity needed to drive meaningful action. As a result, while financial apps increasingly employ AI for personalization, users may ignore, dismiss, or even distrust these messages if they do not clearly understand their relevance or origin. There is a significant gap in user trust and regulatory transparency, and explainability remains a challenge in high-stakes environments.

Recent advancements in Explainable AI (XAI) and microservices-based personalization offer an opportunity to reimagine the kind of notifications financial apps provide to consumers. By leveraging real-time behavioral insights and providing tailored alerts or "nudges" that explain their rationale, financial technology platforms can improve engagement, trust, and long-term financial wellness. This paper outlines a modular architecture that uses event driven microservices, messaging systems, machine learning models, and SHapley Additive exPlanations (SHAP)-driven explanation generation to deliver contextual, interpretable, and actionable nudges. While popular fintech tools like Cleo and Revolut offer real-time spending notifications and basic categorization, their systems often lack transparency into why a particular message was generated. This paper proposes architecture that addresses this gap by combining real-time data ingestion, decision intelligence, and human-readable explanation mechanisms in a scalable microservices framework.

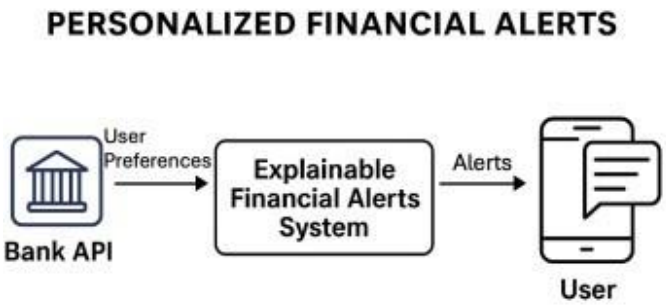

Fig 1: Personalized Financial Alerts Nudge System

2. Related Work

Recent years have seen increasing interest in using artificial intelligence to provide personalized financial guidance. However, explainability and user trust remain major challenges, especially in high-stakes domains like personal finance. Explainable AI (XAI) has gained traction to help users understand how and why AI systems make certain decisions. SHAP (SHapley Additive exPlanations), introduced by Lundberg and Lee [1], offers a model-agnostic approach that explains predictions by attributing contributions to individual input features. Its mathematical grounding makes it suitable for sensitive applications like financial nudging, where transparency is essential. Explainability frameworks have also been adopted in regulated contexts such as credit scoring and financial ratings, demonstrating real-world applicability of XAI in finance [2]. In fact, a user study by Ben-David et al. [3] found that when users were provided with explanations (like SHAP outputs), they were more likely to trust and follow automated financial advice.

Personalized nudgingtimely, relevant prompts to influence user behaviorhas been explored in fintech applications but is often limited to simple rules or static triggers. While apps like Cleo and Revolut offer basic spending alerts and categorization, they lack transparency around why specific notifications are triggered. This limits user trust and may reduce long-term engagement. Some financial platforms like Cleo and Revolut offer personalized alerts, but their reasoning models are opaque and not publicly documented. Cleo uses a chatbot interface to deliver budget reminders and spending tips, but its alerts often lack detailed explanations tied to user behavior. Revolut offers real-time spend tracking and category-based budgeting; however, its notifications typically focus on descriptive analytics rather than providing explanatory context about what triggered a specific alert. Academic work on XAI in financial decision-making exists but rarely focuses on real-time, user-facing explanations within modular architectures. The need for interpretability in financial decision-making has been widely acknowledged, particularly as machine learning systems face scrutiny for opacity and lack of auditability in high-stakes environments. This is critical because financial advice directly affects user decisions and well-being.

On the infrastructure side, microservices and event-driven architectures are commonly used in modern financial systems to ensure scalability and modularity. Kafka is widely adopted for real-time event processing due to its strong delivery guarantees, partitioning, and ability to replay [3]. However, few published systems combine these architectural benefits with explainable AI to deliver contextual, human-readable financial nudges. In summary, while existing literature explores explainability, personalization, and microservices individually, there is a gap in integrating all three into a real-time, transparent financial advisory system. This paper addresses that gap by proposing a Kafka-based microservices pipeline that delivers explainable, context-aware nudges to users based on behavioral and transactional signals.

3. System Architecture and Context-Aware Data Pipeline

Despite widespread adoption of AI-driven personalization in financial applications, many existing systems fall short in two key areas: contextual relevance and explainability. Users often receive generic alerts such as “You’re spending more than usual” or “Try saving more,” which fail to account for the user’s financial habits, time-sensitive context (e.g., rent week or holidays), or location-based spending. These alerts are typically static, one-size-fits-all, and lack transparency about why they were triggered. As financial decisions directly impact users' well-being, trust in automated suggestions becomes crucial. However, current systems rarely communicate the logic behind their recommendations. Without clear, understandable reasoning, users may distrust or ignore these messagesundermining both user engagement and financial outcomes. Moreover, most existing architectures are monolithic or tightly coupled, making it difficult to scale, personalize, or add interpretability features.

The problem becomes more acute in real-time systems, where delivering timely and tailored nudges at scale requires a modular, fault-tolerant architecture. At the same time, the financial domain demands rigorous attention to privacy, auditability, and regulatory transparencyrequirements that are poorly supported in traditional black-box AI systems. This study addresses these gaps by asking: How can we build a real-time, modular system that delivers explainable, context-aware financial nudges, while preserving user trust, system scalability, and data privacy? The proposed system is composed of modular microservices that interact asynchronously a messaging system. In this proposal, Apache Kafka is a suitable implementation, and the system

utilizes Kafka "topics" to interact via events. Each service fulfills a discrete responsibility and collectively enables real-time, explainable nudging at scale.

3.1. Data Ingestor:

This stateless microservice reads transaction data from the Bank API Interface and produces structured events to the "transactions" Kafka topic. It performs basic validation, schema enforcement, and user ID tagging for downstream correlation.

3.2. Context Processor:

This component subscribes to the "transactions" topic and enriches each record with historical user behavior, spending category tags, budget goals, geolocation, and time-based context. It relies on internal APIs to retrieve user profile data and budget metadata from a secure database. The enriched event is then published to the "context-events" topic.

3.3. Nudge Decision Engine:

This service consumes from the "context-events" topic. It evaluates if a nudge should be sent using either:

- Rule-based logic (e.g., "if grocery spending exceeds 80% of budget")
- A trained ML model that classifies events as 'nudge-worthy' or not. The model is trained on labeled behavioral data and validated against financial outcomes. Events deemed relevant are passed to the XAI module along with prediction metadata and model inputs. A corresponding message is published to the "nudge-decisions" topic.

3.4. XAI (Explainable AI) Module (SHAP Engine):

This module consumes decision outputs and applies SHAP to the ML model inputs to generate explanations. SHAP (SHapley Additive exPlanations) is a unified framework for model interpretability that assigns a contribution value to each input feature by analyzing the marginal impact of that feature across all possible combinations [1]. SHAP has also been successfully applied in financial settings such as accounting audits, where transparency into model outputs is critical for trust and compliance [4]. In financial nudging, SHAP can highlight which factors it evaluates such as a spike in dining-out expenses or nearing a budget threshold had the most influence on the alert generation.

For example, if a user's grocery expenses increased 25% compared to their previous three-month average, and they are approaching 90% of their budget limit, SHAP assigns proportional values to these features. This not only aids developers and data scientists in understanding model behavior but also enables transparent and tailored communication with users. The system then uses a templating engine to convert SHAP outputs into readable explanations such as: "This alert is based on your recent grocery spending pattern and your budget goals." Optional visualizations like SHAP force plots or summary bar charts can be embedded into the alert for additional clarity. The system also logs the SHAP explanation for developers to analyze and optionally includes simple visual cues like bar charts for the user. Final messages are published to the "nudge-explanations" topic.

3.5. Notification Service:

This component listens to "nudge-explanations" and pushes formatted alerts to the end user via in-app notifications, emails, or integrated chatbots, depending on the preferred mode of alerting. It ensures delivery acknowledgments and retries failed sends. In this proposal, each service is ideally independently deployable and designed for horizontal scalability. Stateless services persist minimal metadata in internal databases, relying heavily on Kafka's sequencing and service coordination. A service mesh manages traffic, load balancing, retries, and encryption of service-to-service communication. Kafka Connectors may also be employed for integration with databases or analytics pipelines. The system logs enable developers, product managers and auditors to view the explanations that the system makes to improve or make modifications to the system. This is core to Explainable AI and will help us determine why certain explanations are being triggered. To deliver effective financial nudges, the system must go beyond simple transactional analysis and instead construct a rich, contextual understanding of each user's financial behavior. This context is derived by aggregating multiple input streams that, when combined, provide a nuanced and real-time snapshot of user activity, preferences, and risk factors.

The foundation of this context model is built on three primary data sources. First, real-time and historical bank transaction data is ingested through open banking APIs (e.g., Plaid), offering structured insight into merchant details, amounts, timestamps, and categorized spending. These inputs allow the system to track trends, flag budget violations, and detect anomalies across common categories like groceries, dining, or utilities. Second, user behavior signals, such as logins, previously ignored nudges, and updates to savings goals, such as adding spending thresholds for buying a home, are collected to assess user engagement and responsiveness. These behavioral events inform the system's ability to fine-tune the tone, frequency, and relevance of nudges over time. Third, regular events such as salary deposits, rent or mortgage due dates, and subscription renewals are used to model expected financial flows. These patterns establish a temporal rhythm for each user, enabling the system to anticipate cash flow constraints or peaks in spending behavior.

To complement these core inputs, the system incorporates a range of enriched contextual signals. For example, if a user opts to share his location data with the app, geolocation data can enable and trigger location-based alerts. An example is flagging

increased discretionary spending when the user is near a high-end shopping area. Likewise, spending history across prior months helps establish personalized baselines and identify deviations, such as seasonal spending surges or recurring overspending trends. Temporal patterns, including pay cycles and bill due dates, allow for timely and anticipatory nudging. The system can also integrate with personal or financial calendars enables proactive budgeting recommendations for holidays, travel, or known large expenses. Finally, the system accounts for user-defined financial goalssuch as saving for a home or paying off debtby assessing how real-time spending aligns with or deviates from these objectives. This enables more impactful messages like, “This purchase may delay your goal of saving for a house.”

These diverse data streams are unified and processed by the Context Processor microservice, which generates a consolidated, timestamped snapshot of each user’s financial state. This enriched context is then published to the context- events Kafka topic, providing critical input to downstream services such as the Nudge Decision Engine and the XAI Module. Together, these components ensure that real-time financial nudges are not only timely but also personally relevant and interpretable, grounded in each user’s evolving financial landscape, and also providing the necessary context to guarantee user’s satisfaction and engagement. The proposed system relies on Apache Kafka as the event-streaming backbone, facilitating scalable, asynchronous communication between microservices. Each Kafka topic represents a distinct semantic layer in the data processing pipeline, enabling modularity, resilience, and reusability. The topic design and processing stages align with best practices for Kafka- driven microservices as outlined in [4]. The 'user-events' topic captures behavioral telemetry from user-facing interfaces such as mobile apps or web portals.

Events include user interactions like app logins, ignored nudges, and goal modifications. These signals provide real-time insight into engagement patterns and behavioral trends. The Context Processor consumes this stream to dynamically adapt nudging logic and to personalize recommendations based on user responsiveness. The 'transactions' topic is the main source of financial activity data. It ingests normalized transaction records via integrations with third-party banking APIs (e.g., Plaid). Each event typically includes the transaction amount, category, merchant metadata, and timestampall necessary to understand individual users’ spending habits. This stream is consumed by the Context Processor for enrichment and is retained for historical trend analysis and downstream auditability via logs. If users opt in to location sharing, the system publishes geolocation metadata to the 'location-events' topic. This enables the detection of contextual triggers such as store visits, travel, or geographically relevant expenses.

The Context Processor fuses this stream with transactions to support location-aware nudgingfor example, detecting discretionary spending spikes while on vacation. Another example cited earlier is discretionary spending alerting when a user is in close proximity to a retail store and if they are past their monthly spending thresholds. The 'context-events' topic contains event payloads enriched by the Context Processor. It represents a holistic snapshot of user financial behavior at a point in time, integrating data from 'transactions', 'user-events', 'location-events' (if enabled), and internal user profile databases. Enrichment features may include budget consumption percentage, deviation from average category spend, upcoming bill reminders, and inferred pay cycles. This stream feeds the Nudge Decision Engine and enables decision-making that is context aware.

The Nudge Decision Engine produces events to the 'nudge-decisions' topic, which encapsulates the outcome of rule-based logic or machine learning classification. Each event includes a binary decision (e.g., ‘send’ or ‘suppress’) by way of a flag, along with prediction confidence scores, triggering conditions, and relevant feature metadata. These records are consumed by the XAI module for explanation generation and are also useful for model performance monitoring and A/B testing. The 'nudge-explanations' stream contains fully formatted, human-readable messages generated by the XAI module. These explanations are derived from SHAP value analysis and templated into user-facing messages. For instance, a typical payload might state: “Your grocery spending this week is 25% above your monthly average.

Consider adjusting your weekly grocery budget.” The Notification Service consumes this topic to distribute alerts via mobile push, email, or chatbot interfaces, per user preferences. This event-topic structure decouples service responsibilities, supports horizontal scaling, and enables data replay for compliance, model retraining, and debugging. Each topic serves as a well-defined contract between upstream and downstream consumers, reinforcing the reliability and modularity of the financial nudging system.Kafka’s features of ordering and ability to replay are crucial. Partitioning by user ID ensures ordered processing. At- least-once delivery is employed for idempotent consumers and processors. Exactly-once delivery is required for sensitive decisioning processors. The system ensures that services can recover from faults and reprocess events without inconsistencies. As demonstrated in [4], Kafka offers strong guarantees for event ordering and service decoupling, which are essential in real-time financial pipelines.

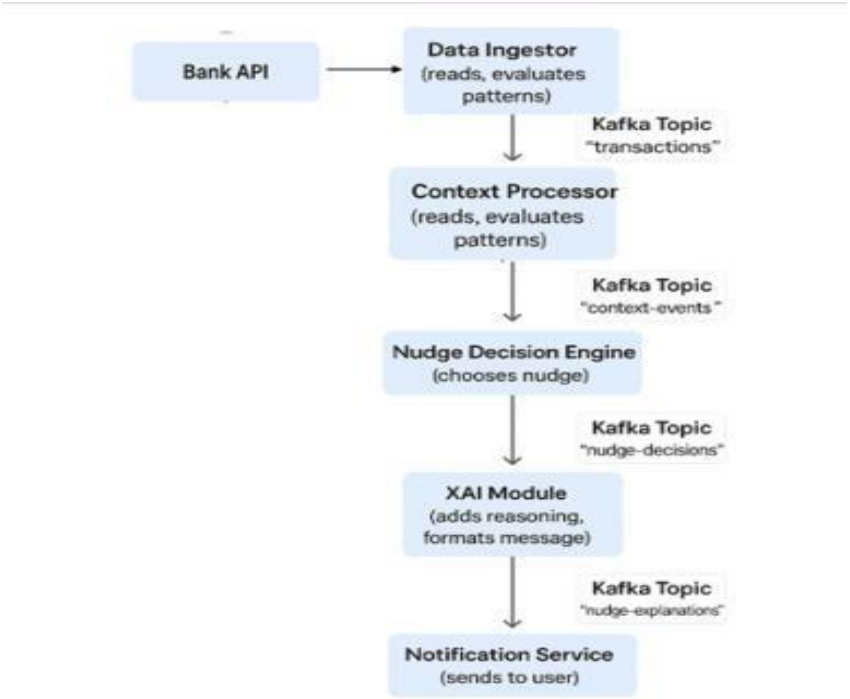

Fig 2: System Overview of Modular Microservices and Interactions with Kafka Topics

4. Explainable AI for Financial Nudging

The XAI module uses SHAP (SHapley Additive exPlanations) to provide transparency in machine learning decisions, particularly those related to personalized financial nudging. SHAP values help explain how each featuresuch as grocery spending, budget limits, or time-based patternscontributed to a model's output. For example, a user might receive an alert that states: "This alert is based on your increased grocery expenses this week and the fact that you're nearing your monthly grocery budget." This message is generated dynamically from SHAP values in conjunction with data from the user's transaction history and personal profile. The module may optionally include visual aids such as bar charts or force plots to enhance user comprehension. SHAP is a model-agnostic method grounded in cooperative game theory, to explain machine learning predictions. SHAP assigns a value to each input feature, indicating how much it contributedpositively or negativelyto a particular output decision. This makes it well-suited to generating transparent, individualized explanations for financial nudges triggered by the Nudge Decision Engine. Recent research emphasizes the need for temporal explainability in financial applications [5], supporting our use of sequential transaction history as part of the SHAP-based justification process.

In this context, each feature is treated as a "player" in a prediction "game," and the outcomei.e., the system's decision to trigger a nudgeis broken down to reveal each feature's role. Within the system's architecture, model inputs and prediction outcomes for nudge-worthy events are passed to the SHAP engine, which uses appropriate explainers such as TreeExplainer or KernelExplainer, depending on the underlying model type. These SHAP values not only support developer-facing interpretability and auditability but also allow downstream components to generate user-facing explanations that build trust and behavioral compliance. Recent research underscores the need for temporal explainability in financial domains [5], which motivates our use of time-sequenced transaction histories as part of the SHAP-based reasoning process. SHAP performs feature attribution by quantifying how much each input contributed to a model's prediction for a given user event. In the context of financial nudging, this includes features such as current spending within a specific category like groceries, the percentage of the user's monthly budget already utilized, the velocity of spending based on recent daily trends, the presence of recurring transactions (e.g., subscriptions or bills), and the calendar date, such as proximity to payday. These contributions help the system generate nuanced, user-specific explanations that clarify the reasoning behind each nudge.

4.1. Example Attribution Output:

Below is an example of the how the SHAP framework might assign values to a certain type of event:

Table 1: Example Attribution Output

Feature	SHAP Value
Grocery Spend (past week)	+0.35
Budget Threshold (90% hit)	+0.28
Salary Deposit (recent)	-0.12
Previous Alert (ignored)	-0.05

The system converts these values into a human-readable message:

"This alert was generated based on your increased grocery spending this week and the fact that you are nearing your set budget." Visualizations like SHAP bar plots or force plots can be included in emails or app dashboards to provide deeper insight into the model's rationale. SHAP offers several advantages within this system. It provides local interpretability by explaining individual predictions for specific user events, rather than offering only generalized model behavior. This is crucial for personalized financial nudges where users expect explanations tailored to their specific actions. Additionally, SHAP stands out as the only method with formal guarantees of fairness, consistency, and local accuracy, making it a reliable choice for financial systems where trust and transparency are critical. Beyond user-facing benefits, SHAP also supports internal model debugging by revealing potential sources of bias, feature leakage, or unintended correlations during the model development phase. However, the computational cost of SHAP can be significant, especially in real-time applications.

While SHAP is powerful, it can be computationally intensive. The system implements several performance optimizations: caching commonly encountered SHAP explanations, using SHAP variants and asynchronous explanation generation. Employing fast SHAP variants like TreeExplainer for decision tree models and generating explanations asynchronously via Kafka to prevent any delay in user interactions. These optimizations ensure the architecture remains responsive and scalable, capable of serving thousands of users simultaneously without introducing latency bottlenecks. It is worth mentioning the the drawback of using caching may lead to organizations exceeding their budgets to maintain extensive caching mechanisms. By incorporating SHAP in the decision pipeline, the system not only delivers more actionable and context-aware nudges, but also helps organizations build transparency and trust with users. Users are more likely to respond positively and take meaningful action to financial advice when they understand its rationale, and regulators are more likely to accept systems that maintain explainability logs and reasoning traceability.

Let's look at a short example of how a user connects their bank account and the information that the system ingests as transactions:

- A spike in grocery spending is detected by the Context Processor.
- The Nudge Engine evaluates this against past behavior and budget goals.
- A decision to nudge is made.
- SHAP explains this decision based on features: high spending and nearing budget.
- A notification is generated: "You're close to your grocery budget. Consider limiting expenses this week."

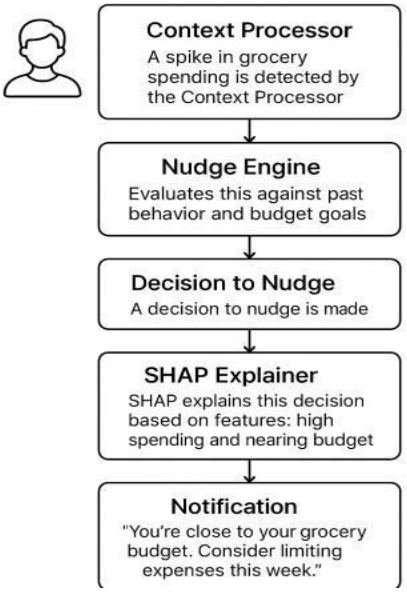

Fig 3: Example of How the System Ingests Data

SHAP has been employed in various financial AI systems, including explainable trading models [6] and responsible transaction classifiers [7]. However, few existing architectures integrate SHAP into a real-time nudging pipeline with event- driven microservices, as proposed in this work.

5. Security and Privacy Considerations for the System

The proposed system handles sensitive user financial and behavioral data, requiring robust safeguards to ensure security, data privacy, and regulatory compliance. Security is enforced at multiple layers of the architecture, from API authentication to inter-service communication and data governance. All third-party integrations with banking APIs use OAuth 2.0 for secure, delegated access, with access tokens scoped to the minimum permissions necessary to reduce risk. Internally, role- based access

control (RBAC) governs permissions between stateless microservices. The system adheres to privacy-by-design principles by processing and storing only the features necessary for contextual modeling and decision-making, while personally identifiable information (PII) is redacted or pseudonym zed wherever feasible. User consent is explicitly obtained for all data collection financial, geo location, and behavioral with preferences managed through a centralized consent management service that allows real-time updates or revocation. Kafka's event history and employing secure logging provides a transparent mechanism for auditing and rollback. External API calls are rate-limited and monitored for anomalies, and API gateways enforce throttling, IP whitelisting, and token expiration to prevent abuse. All data is encrypted both in transit via TLS and at rest, with scoped tokens and rate limits further securing API access. These layered protections ensure the system remains compliant, resilient, and trustworthy in handling financial data.

6. Conclusion and Future Work

This paper presented a modular, real-time nudging system that combines event-driven microservices, machine learning, and explainable AI (XAI) to deliver actionable, context-aware financial advice. By leveraging Kafka for scalable event streaming and SHAP for interpretable decision-making, the system addresses core shortcomings in existing financial notification systems, which often fail to engage users due to their generic nature and lack of transparency. The value of the system lies in several dimensions. From a technical standpoint, the microservices-based architecture ensures modularity, scalability, statelessness, and fault tolerance, making it deployable in modern cloud-native fintech environments. Kafka's event replay and partitioning capabilities allow for robust, stateless processing, auditing, and traceability features critical for financial applications subject to regulatory oversight.

They reinforce the system's asynchronous, real-time processing model, enabling services to scale independently while maintaining consistency and fault tolerance. The architecture also enables rapid iteration, A/B testing, and continuous integration of new data sources without disrupting existing workflows and secure logging for providing understanding why certain decisions were made by XAI. Security is enforced at multiple levels of the system through OAuth 2.0 authentication, encrypted communication (TLS), role-based access control (RBAC), and user-consent-driven data policies to ensure safe and compliant handling of sensitive financial information. From an end-user experience perspective, the system builds trust by providing clear, personalized, and timely nudges that are backed by explicit explanations. Rather than presenting opaque alerts, the system leverages SHAP to generate user- facing rationale, for example, "Your grocery expenses are 25% higher than your monthly average", which enhances perceived fairness, personalization, transparency, and relevance.

This level of interpretability is especially crucial in financial decision- making, where users are sensitive to advice that affects their well-being. From an industry and regulatory perspective, the system aligns with emerging requirements for responsible AI, including explainability, fairness, data privacy, and user consent. As financial institutions face increasing pressure to provide auditable, ethical AI systems, this architecture provides a practical blueprint for integrating explainability at every stage from decision to delivery while honoring user privacy and data minimization principles. Future work will explore multiple enhancements. First, behavioral A/B testing will be conducted to measure the real- world effectiveness of nudges and explanations in driving positive financial behavior.

Metrics such as user behavior, budget adherence, and user trust scores determined via adoption and engagement will help validate the system's impact. Second, the architecture will be extended to cover additional domains such as debt management, emergency savings, and financial goal planning, requiring new models and rule sets. Third, incorporation of reinforcement learning or adaptive feedback loops will allow the system to dynamically tailor nudges over time based on user outcomes and preferences. Overall, this work contributes an applied, urgent and interpretable framework for embedding ethical decision intelligence into real-time fintech applications. By combining personalization with explainability, it moves one step closer to closing the trust gap between consumers and automated financial systems.

References

[1] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Conference on Neural Information Processing Systems*, Long Beach, California, 2017.

[2] S. Kim and J. Woo, "Explainable AI framework for the financial rating models: Explaining framework that focuses on the feature influences on the changing classes or rating in various customer models used by the financial institutions.," in *International Conference on Computing and Pattern Recognition*, Shanghai, China, 2021.

[3] J. Kreps, N. Narkhede and J. Rao, "Kafka: a Distributed Messaging System for Log Processing," 2011.

[4] V. K. Carl, P. Weber and O. Hinz, "Applications of Explainable Artificial Intelligence in Financea systematic review of Finance, Information Systems, and Computer Science literature," *Management Review Quarterly*, vol. Volume 74, no. June 2024, p. 867– 907, 28 February 2023.

[5] P.-D. Arsenault, S. Wang and J.-M. PATENAUDE, "A Survey of Explainable Artificial Intelligence (XAI) in Financial Time Series Forecasting," *ACM*, p. 35, 2018.

[6] C. Maree, J. E. Modal and C. W. Omlin, "Towards Responsible AI for Financial Transactions.," in *IEEE Symposium Series on Computational Intelligence*, Canberra, Australia, 2020.
[7] D. B. David, Y. S. Resheff and T. Tron, "Explainable AI and Adoption of Financial Algorithmic Advisors: an Experimental Study," in *Artificial Intelligence, Ethics, and Society*, 2021.
[8] K. Venkatachalapathi, "Event-Driven Architecture: Harnessing Kafka and Spring Boot for Scalable, Real-Time Applications," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 10, no. 6, pp. 474-86, 2024.
[9] V. L. Bandara, "Medium," [Online]. Available: https://vitiya99.medium.com/event-driven-microservices-with-spring-boot-and-kafka-73506c96fd43.
[10] S. Kumar, V. Mendhikar and V. Ravi, "Explainable Reinforcement Learning on Financial Stock Trading using SHAP," 2022.