AB Journals

Advanced Journal of Science, Technology and Engineering

ISSN 2997-5972 • Volume 6, Issue 2, 2026 • pp. 27–47

RESEARCH ARTICLE Received: 1 Sep 2026, Accepted: 26 Oct 2026, Published: 16 Nov 2026

An AI-Driven, Workflow-Aware Virtual Customer Support Assistant for Small and Medium-Sized Enterprises

Dr. Damilare Akinwunmi*; Sunday Oladipo; Eromoigbe Winner Agbonikpeya; Valerian Olisaemeka Ejezie; Victor Finecountry-Okezie; Dayo Francis Akinyeke; Uchenna Chidi-Aja; Evelyn-Ruth Okiemute Ijomoni; Oluwanifemi Joseph Gbambisomo; Samuel Ugonna Emeka-Ukandu; Oluwafiyinsola Praise Akenroye; Oluwatobiloba Joshua Adekunle; Oluwatobiloba David Fasakin; Esther Fisayomi Aiyenimelo

Department of Computer Science, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

Corresponding Author's Email: akinwunmid@babcock.edu.ng

Abstract

Small and medium-sized enterprises (SMEs) that sell through messaging platforms often handle customer enquiries manually, leading to fragmented conversations, delayed responses and inconsistent service. This article proposes a workflow-aware conversational support assistant that combines a large language model (LLM), retrieval grounding, merchant API tools and context-preserving human hand-off. Using a design-science-oriented literature synthesis, in which sources were identified through keyword searches of academic databases, screened for relevance to messaging-based SME support and LLM-agent design, and analysed thematically, the study derives three design priorities: grounding customer-facing claims in current merchant data, separating model reasoning from validated execution, and transferring accumulated conversational state when escalation occurs. The resulting five-layer architecture comprises messaging integration, agent orchestration, retrieval and grounding, store action, and escalation and hand-off. Volatile data such as price, inventory and order status are retrieved live; state-changing operations require explicit customer confirmation and schema validation before execution. The article also specifies an evaluation protocol based on scripted end-to-end customer journeys using journey completion, factual agreement, action correctness, escalation precision and recall, context preservation, and response latency. The contribution is architectural and methodological; implementation and empirical validation remain future work.

Keywords: Conversational Commerce; Customer Support Automation; Human-in-the-Loop Escalation; Large Language Models (LLMs); Retrieval-Augmented Generation (RAG); Small and Medium-Sized Enterprises (SMEs); Tool-Augmented Agents

How to Cite: Akinwunmi, D., Oladipo, S., Agbonikpeya, E. W., Ejezie, V. O., Finecountry-Okezie, V., Akinyeke, D. F., Chidi-Aja, U., Ijomoni, E. O., Gbambisomo, O. J., Emeka-Ukandu, S. U., Akenroye, O. P., Adekunle, O. J., Fasakin, O. D., & Aiyenimelo, E. F. (2026). An AI-Driven, Workflow-Aware Virtual Customer Support Assistant for Small and Medium-Sized Enterprises. *Advanced Journal of Science, Technology and Engineering*, 6(2), 27–47. https://doi.org/10.52589/AJSTE-CSGFXK

© 2026 by the authors. Published by AB Journals. This is an open access article distributed under the terms of the Creative Commons Attribution (CC BY 4.0) license.

INTRODUCTION

Customer service at a small business spans every interaction with a customer, from the first question through ordering, fulfilment, and follow-up. In many small firms, especially in developing markets, these interactions are spread manually across phone, email, social media, and messaging apps. This produces slow responses, requests that fall through the cracks between channels, and support quality that depends heavily on which staff member happens to be free (Febriani et al., 2025; Gerea & Herskovic, 2022).

The problem is more acute where messaging platforms function as the main sales channel rather than an add-on. In Nigeria, more than half of surveyed MSMEs already using e-commerce relied solely on social media channels such as Facebook, Instagram and WhatsApp for online selling (GSMA, 2023). Nigerian social-commerce research also identifies user trust as an important factor in platform adoption (Odubo & George, 2026). These conditions make the conversation itself an important commercial interface, so a support solution for this setting should operate inside the messaging journey rather than redirect customers to a separate website or helpdesk.

Progress in large language models makes it feasible to build conversational agents that understand natural, unscripted customer language and invoke external tools. A standalone model, however, has no authoritative access to a merchant's current catalogue, prices, inventory or order state, and tool-using agents can still deviate from prescribed procedures over long interactions. A commerce assistant therefore requires both live data grounding and explicit controls over execution.

This paper synthesises the relevant research and proposes a workflow-aware assistant built around four capabilities: natural-language interaction, grounding in a specific merchant's live data, reliable multi-step tool use, and escalation to a human without discarding established context. The contribution is architectural and methodological rather than empirical: the article specifies a reference architecture, its control boundaries, and the evaluation protocol by which a subsequent implementation can be assessed.

Existing AI-driven customer support systems address parts of this problem but not the combination it requires. Retrieval-augmented support assistants answer questions from documentation or historical tickets; for example, a knowledge-graph-based retrieval system used by LinkedIn's customer service team improved retrieval accuracy and reduced median per-issue resolution time (Xu et al., 2024). Such systems ground answers in relatively stable knowledge, however, and do not execute transactions against a merchant's live catalogue, stock and order records. WhatsApp commerce chatbots show that product discovery, ordering and fulfilment can be integrated into a single messaging channel (Mhiribidi et al., 2024), but the literature reviewed does not report how such systems combine open-ended LLM interaction with live-data grounding and execution controls. General-purpose tool-using LLM agents can act on external systems, yet benchmark studies of customer-service tasks report that agents complete the same task inconsistently across repeated trials (Yao et al., 2025) and deviate from prescribed procedures over complete conversations (Arcadinho et al., 2024). Research on machine-human hand-off, finally, models when a conversation should be transferred but gives less attention to the transactional state that should accompany the transfer (Liu et al., 2021).

The research gap addressed here is therefore not the absence of any single capability, but the absence, in the literature reviewed, of a reference design that combines open-ended language

understanding, grounding in live merchant data, gated state-changing actions and context-preserving hand-off within the messaging channels through which many SMEs already sell. Relative to the systems above, the article makes four specific contributions:

C1: A five-layer reference architecture for messaging-channel SME support in which customer-facing commercial facts (price, stock, order status) are resolved against live merchant records at the turn in which they are used, rather than against a static knowledge base or model recall.

C2: An execution-control pattern, consisting of schema validation at the boundary between orchestration and store action and an explicit customer confirmation gate, that treats the model's output as a proposal to act rather than as an action.

C3: A specification of hand-off as the transfer of a defined state package (transcript, relevant order context and pending action) within the customer's existing messaging thread, rather than as routing alone.

C4: A reproducible evaluation protocol comprising traceable requirements, a journey-based test design covering routine and failure scenarios, baseline and ablation conditions, and pre-specified metrics, so that the effect of each control can be tested in a subsequent implementation.

These contributions are design propositions. Their effectiveness has not yet been demonstrated and is stated as hypotheses in Section 4.2.

Accordingly, the study addresses three research questions: RQ1, how can a conversational support assistant ground customer-facing statements in live merchant data while retaining natural-language flexibility? :RQ2, what architectural controls can reduce incorrect state-changing actions across multi-turn journeys? :RQ3, how can escalation transfer sufficient conversational and transactional context for a human agent to continue without unnecessary repetition?

LITERATURE REVIEW AND THEORETICAL FRAMEWORK

Customer Service Challenges in SMEs

Research on multichannel customer support identifies fragmentation as a structural problem rather than simply a result of insufficient staff effort. When channels operate independently, organisations can experience missed messages, disconnected conversations and reduced efficiency (Gerea & Herskovic, 2022). Adding more channels without integration and coordination can therefore increase operational complexity rather than improve service (Febriani et al., 2025). Research focused on SME customer experience also highlights resource constraints, data-protection concerns and difficulty maintaining a consistent experience across channels (Alabi et al., 2024).

Conversational Commerce

Conversational commerce uses messaging applications, chatbots and related interfaces to handle enquiries, purchases and support within the same conversational channel. Industry sources describe messaging applications as an increasingly common channel through which online retailers interact with customers (Shopify Staff, 2023). Academic work demonstrates

that product discovery, ordering and fulfilment can be integrated into a WhatsApp-based commerce chatbot (Mhiribidi et al., 2024). However, the literature reviewed here provides limited evidence on architectures that combine open-ended LLM interaction, live store grounding, controlled state-changing actions and context-preserving human hand-off in one end-to-end design.

From Scripted Chatbots to LLM Agents

Conventional chatbots rely on decision trees and a limited set of recognised intents, which works for expected paths but breaks down on unfamiliar phrasing. LLM-based agents go further by pairing a reasoning engine with memory and access to external tools, letting the system plan across multiple steps and call an interface or database as part of handling a request (Masterman et al., 2024). Wider surveys of agent-based systems describe them as increasingly capable collaborators, while warning that hallucination and domain-specific safety risks remain even with safeguards in place (Xi et al., 2025).

Grounding and Reliability

A general-purpose language model does not have authoritative access to a merchant's current catalogue, price or inventory and may therefore produce unsupported or stale claims. Retrieval-Augmented Generation can mitigate this problem by conditioning generation on retrieved evidence (Lewis et al., 2020). In this study, retrieval grounding extends beyond documents to live merchant records and interface schemas. Task-relevant database and API documentation can materially improve natural-language generation of SQL and API calls (Marketsmüller et al., 2026). Tool-augmented agents can nevertheless struggle to follow procedures across complete conversations, supporting the use of explicit validation and confirmation controls (Arcadinho et al., 2024).

Human Escalation

Hybrid service research indicates that human involvement remains important when automated support reaches its limits. Direct human interaction can improve perceived justice and satisfaction in complaint recovery (Jeanpert et al., 2021). Customers may also prefer early, user-initiated transfer in complaint-handling contexts (Chacon & Martinez-Troncoso, 2026). For an operational assistant, escalation therefore needs clear triggers such as an explicit request for a person, repeated task failure, or a policy-sensitive case. Joint modelling of hand-off and service satisfaction demonstrates the value of integrating escalation decisions with service outcomes, although less guidance is provided on the operational state that should accompany the transfer (Liu et al., 2021). This paper therefore treats the transcript, relevant order context and pending action as the minimum hand-off package.

Theoretical Framework

The proposed system rests on three theoretical positions drawn from the literature reviewed above, each of which constrains a specific part of the design.

The first is the grounding principle underlying retrieval-augmented generation. A model's parametric knowledge does not provide an authoritative view of an individual merchant's current catalogue, pricing or stock position. Conditioning generation on retrieved evidence improves factual performance on knowledge-intensive tasks (Lewis et al., 2020). Applied to commerce, this means that assertions about a product, price, stock level or order status should be supported by a current merchant record rather than model recall. Grounding is therefore

treated in this design as a correctness precondition for transactional facts, not merely as an optimisation of prose quality.

The second is the separation of reasoning from execution. Surveys of agent architectures describe a reasoning-and-acting loop in which a model selects and invokes tools (Masterman et al., 2024), while evaluation work shows that such agents degrade over extended multi-turn interactions and drift from prescribed procedures (Arcadinho et al., 2024). If the model's output is treated as a request to act rather than as the act itself, an interpretation failure produces a rejected call instead of an incorrect order. This position motivates both the validation boundary between the orchestration and store action layers and the confirmation gate placed before any state-changing operation.

The third is continuity of context during service recovery. Human interaction can improve perceived service recovery (Jeanpert et al., 2021), while customers may prefer early, user-initiated transfer in complaint settings (Chacon & Martinez-Troncoso, 2026). Escalation is consequently modelled here as a transfer of accumulated state - transcript, relevant order history and the assistant's pending action - rather than as routing alone. A successful hand-off should allow the human agent to continue from the established state without requiring the customer to repeat information unnecessarily.

Together these three positions define the evaluative criteria used later in this article: an assistant is judged on whether its statements agree with store records, whether its committing actions are correct and confirmed, and whether escalation preserves context.

METHODOLOGY

Research Design and Study Context

This study adopts a design science research (DSR) orientation, in which an artefact is designed and evaluated to address an identified class of problems (Hevner et al., 2004). The design science research methodology describes a sequence of problem identification, definition of solution objectives, design and development, demonstration, evaluation and communication (Peffers et al., 2007). The present article reports the problem definition, design objectives, architecture and evaluation protocol; implementation, demonstration and empirical evaluation are explicitly reserved for the next stage of the work.

Literature identification, selection and analysis: The literature underpinning the design was identified through keyword searches of Google Scholar. Search strings combined terms describing the setting ("SME", "small business", "customer service", "omnichannel", "conversational commerce", "WhatsApp") with terms describing the technology ("large language model", "LLM agent", "tool calling", "retrieval-augmented generation", "chatbot", "human hand-off", "escalation"). Searches covered publications up to October 2026.

Sources were included if they (a) addressed customer support or conversational commerce in SMEs or on messaging channels; (b) reported architectures, failure modes or evaluation methods for LLM-based or tool-augmented agents, retrieval grounding, or human escalation; or (c) supplied methodological, market or regulatory context directly relevant to the design. Peer-reviewed journal and conference papers were preferred. arXiv preprints were retained where they documented recent agent and retrieval techniques not yet covered in the peer-reviewed literature, and industry and regulatory documents were used only for market and

compliance context. Studies of general-purpose chatbots with no service or commerce application, and studies concerned with model training rather than system design, were excluded. The final set comprised 20 sources: nine journal articles, six conference or workshop papers, two preprints, and three industry or regulatory documents.

The selected sources were analysed thematically. Each source was examined for the problem it addressed, the design mechanism it proposed, and the limitations or failure modes it reported. These findings were grouped into the themes presented in Section 2: customer service challenges in SMEs, conversational commerce, the shift from scripted chatbots to LLM agents, grounding and reliability, and human escalation. Limitations that recurred across themes were then translated into the design requirements set out in Section 3.3, from which the three design priorities and the five-layer architecture were derived.

The problem addressed is that SMEs selling through messaging platforms lack a satisfactory customer-support solution. Merchants report difficulty handling customer requests that do not fit a fixed script, while existing automated options either constrain customers to predefined paths or, in the case of general-purpose language models, respond fluently but without access to the merchant's actual data. Requirements are derived from this problem statement together with the limitations identified in the literature: fragmented multi-channel support, the brittleness of rule-based systems, the ungrounded output of standalone LLMs, and the failure modes of tool-using agents over long conversations.

The proposed artefact is an AI-powered, workflow-aware virtual customer support assistant combining tool-based workflows with retrieval grounding and a context-preserving human hand-off path. The design is intended to be portable across merchants with different back ends through adapter boundaries, while recognising that actual portability must be demonstrated empirically.

Existing Customer Support Context

Current SME customer support relies principally on WhatsApp, Telegram, telephone calls and other social media channels. SMEs favour these channels because they preserve direct, open communication with customers. At scale, however, the approach becomes inefficient: conversations disperse across multiple platforms, while information about goods, prices, orders and promotions remains isolated within separate business systems.

Rule-based and scripted automation performs poorly when customer language or dialogue flow departs from a fixed structure. More recent language-model-based systems can interpret open-ended queries, but without controlled access to merchant systems they cannot authoritatively resolve requests that depend on current store data or execute store operations. A workflow-aware design must therefore combine flexible language understanding with grounded reads and constrained tool execution.

This study therefore proposes a conversational, workflow-aware assistant that answers in the customer's own language, retrieves the facts an answer depends on, invokes the store's own services on the customer's behalf, and transfers unresolved conversations to human staff with the accumulated context intact.

Requirement Identification

From the problems identified, the system is required to resolve or appropriately route customer requests concerning product information, pricing, delivery, orders and refunds on

supported messaging channels. Specifically:

- **R1 (Language understanding):** The system must interpret customer needs expressed in natural, unscripted language, including across multiple conversational turns.
- **R2 (Grounding):** Responses must rely on actual merchant data: product records, prices, stock levels and order history.
- **R3 (Controlled execution):** The system must be able to carry out authorised store operations through controlled tool invocation.
- **R4 (Context-preserving hand-off):** Where a request cannot be resolved correctly, or where the customer remains unsatisfied, the system must transfer the conversation to a human agent together with the conversation and order context needed to continue it.

These requirements correspond to four design goals: G1, addressing fragmented SME support; G2, grounding responses in merchant data; G3, improving the reliability of tool augmentation across multi-turn journeys; and G4, preserving conversational context through human hand-off.

Derivation procedure: The requirements were derived in three steps so that each can be traced and re-examined. First, the problem statement in Sections 3.1 and 3.2 was decomposed into four problem elements: P1, customer support that is fragmented across channels and, where automated, dependent on scripts that fail when customers use unscripted language over several turns; P2, language-model responses produced without access to current merchant data; P3, tool-using agents that can execute incorrect or unconfirmed actions over multi-turn journeys; and P4, escalation that discards the context already established with the customer. Second, each problem element was matched to the studies in Section 2 that document it or propose a mechanism for addressing it. Third, each problem-evidence pair was restated as a testable system requirement (R1-R4), assigned to a design goal (G1-G4), allocated to the architectural element responsible for satisfying it, and linked to the metric by which it will be evaluated. Table 1 records this chain, so that a change in the problem definition or the evidence can be followed through to the affected part of the design and evaluation.

Table 1: Traceability from Problem Elements to Requirements, Design Goals, Architecture and Evaluation

Req.	Problem element (source)	Supporting evidence	Design goal	Architectural element	Evaluation metric (Table 3)
R1	P1: support fragmented across channels; scripted automation fails on unscripted, multi-turn language (Sections 3.1-3.2)	Gerea & Herskovic (2022); Febriani et al. (2025); GSMA (2023); Masterman et al. (2024)	G1: address fragmented support within the messaging channel	Messaging integration layer; agent orchestration layer	Journey completion rate
R2	P2: LLM responses lack authoritative	Lewis et al. (2020); Marketsmüller	G2: ground responses in	Retrieval and grounding layer; live	Factual agreement

Req.	Problem element (source)	Supporting evidence	Design goal	Architectural element	Evaluation metric (Table 3)
	access to current catalogue, price, stock and order data	et al. (2026)	merchant data	refresh of volatile fields (Section 3.4)	
R3	P3: tool-using agents act inconsistently and deviate from procedures over multi-turn journeys	Arcadinho et al. (2024); Yao et al. (2025); Xi et al. (2025)	G3: improve reliability of tool augmentation	Confirmation gate (orchestration); schema validation (store action layer)	Action correctness
R4	P4: escalation loses the context already established with the customer	Jeanpert et al. (2021); Chacon & Martinez-Troncoso (2026); Liu et al. (2021)	G4: preserve context through hand-off	Escalation and hand-off layer	Escalation precision and recall; context preservation

Note: *Response latency (Table 3) is not tied to a single requirement; it is measured as the cross-cutting cost of the grounding and validation steps introduced for R2 and R3 (H4, Section 4.2).*

Data and Knowledge Source Preparation

The assistant does not rely on a static, pre-collated store corpus. Its knowledge sources are merchant-controlled systems and are divided into four classes: (1) relatively stable catalogue descriptors such as product name, description, variants and attributes; (2) volatile commercial state such as price and inventory; (3) customer and order records, including order status and purchase history; and (4) action schemas that define the fields, types and permitted values accepted by the merchant's e-commerce API.

These classes are handled according to their volatility. Price, inventory and order status are retrieved from the merchant back end at the turn in which they are needed and are not treated as durable conversational memory. This reduces the risk of quoting a stale price, promising unavailable stock or reporting an outdated order state.

Catalogue descriptors are handled differently because customers may refer to products semantically rather than by stock-keeping unit (SKU), for example, "the blue one from your story." Descriptive fields can therefore be embedded and maintained in a vector index for semantic retrieval. When the catalogue changes, the affected index entries are updated.

Semantic retrieval returns candidate product identifiers rather than authoritative commercial values. Before the assistant states a price, availability or other volatile attribute, it resolves each candidate identifier against the current merchant record. The index supports product discovery while the live merchant API remains the source of truth for transactional facts.

Customer and order records are keyed by an internal customer identifier. Platform-specific identifiers, such as a WhatsApp account identifier or Telegram user identifier, must be linked

to that internal identity through an explicit account-association or verification process; identifiers from different platforms are not assumed to refer to the same person. Once linked, the same customer history can be used across supported channels.

Action schemas are obtained from the merchant integration boundary, such as an API specification or schema registry, when an operation is being prepared. The orchestration layer uses the retrieved schema to construct candidate arguments, while the store action layer performs final validation before execution.

Evaluation Design

Evaluation is designed around scripted end-to-end customer journeys rather than isolated single-turn prompts, because failures in tool-augmented agents often emerge across complete conversations (Arcadinho et al., 2024). A journey may include product discovery, delivery questions, order placement and a later order-status request in a separate session, with a known merchant back-end state used as ground truth.

Journey development: Test journeys will be developed in four steps. (1) Category definition: journeys are grouped into the eight categories in Table 2, which cover routine requests and the failure and escalation scenarios that the controls described in Section 4 are designed to handle. (2) Scenario drafting: within each category, concrete scenarios will be drafted from the requirements in Table 1 and, where a participating merchant consents, from de-identified examples of real customer enquiries, so that the test set reflects the products, phrasing and code-mixed language of the target customers. (3) Specification: each journey is written as a script containing the customer persona and goal, the sequence of customer turns with permitted paraphrases, the initial state of the test store, and an answer key. (4) Review: each script will be reviewed independently by two members of the research team against the merchant's stated policies; disagreements will be resolved by discussion, and the agreed script will be frozen before any system condition is run.

Answer keys: Correct answers and expected tool actions are established from the test store, not from model output. Before each run the store is reset to a fixed snapshot of catalogue, prices, stock, customers and orders. For each turn that requires a factual statement, the answer key records the authoritative value in that snapshot. For each state-changing step, it records the expected tool call and arguments, the point at which customer confirmation must precede the call, and the expected post-condition in the store, such as a new order with a specified line item. For failure journeys, the key records the expected safe behaviour, such as asking a clarifying question, offering an in-stock alternative, reporting a failed transaction without creating an order, or escalating with the specified hand-off package. Where more than one response is acceptable, the key lists the acceptable set rather than a single string.

Selection and size: The test set will include journeys from every category in Table 2, with routine and failure-or-escalation journeys balanced so that both escalation precision and escalation recall can be estimated. The number of journeys per category and the number of repeated runs per journey will be fixed and reported before testing. Repeated runs are needed because tool-using agents can succeed on a task in one trial and fail on the same task in another (Yao et al., 2025).

Table 2: Customer Journey Categories and Answer-Key Construction

Journey category	Example scenario	Answer key: correct answer and expected tool action
1. Product enquiry (routine)	Customer asks whether a named item is available in a given size and what it costs.	Price and stock taken from the store snapshot; no state-changing call.
2. Order placement (routine)	Customer selects an item, quantity and delivery option and asks to buy.	Confirmation message stating item, quantity and price; create-order call with the expected arguments only after the customer confirms; order present in the store afterwards.
3. Order status in a later session (routine)	A verified returning customer asks where an earlier order is.	Status read from the order record linked to the verified identity; no other customer's data disclosed.
4. Ambiguous request (failure)	Customer refers to "the one from your story", which matches several products.	Clarifying question listing the candidates; no price quoted and no order created until the product is identified.
5. Unavailable product (failure)	Requested variant is out of stock or discontinued.	Unavailability stated from live stock; in-stock alternatives may be offered; no order created for the unavailable item.
6. Failed transaction (failure)	Store API returns an error, or payment fails, while an order is being placed.	Failure reported to the customer; no duplicate or partial order; store state unchanged; retry or escalation offered.
7. Unauthorised or out-of-scope request (failure)	Customer asks to change another person's order, or requests an operation the assistant is not configured to perform.	Request declined or escalated; no cross-customer data access; no state-changing call.
8. Escalation to human support (escalation)	Refund dispute, complaint, explicit request for a person, or repeated task failure.	Escalation at the specified turn; ticket contains transcript, relevant orders and pending action; assistant stays silent until the ticket is resolved.

Note: *Categories 4-8 are failure and escalation scenarios. Category 7 also serves as a security check on per-customer authorisation (Section 3.6).*

Each journey is assessed using the six measures defined in Table 3, grouped into three evaluation dimensions:

1. Effectiveness - journey completion rate and factual agreement.
2. Safety and hand-off - action correctness, escalation precision and recall, and context preservation.

3. Efficiency - response latency from inbound webhook to outbound reply.

For reproducibility, the implementation study should fix and report the model and version, system prompt, inference settings, retrieval configuration, tool schemas, test-store state, number and composition of journeys, and the number of repeated runs for stochastic conditions. Acceptance thresholds and the statistical analysis plan should be specified before examining comparative results.

Data Protection and Implementation Requirements

Data protection: The assistant processes personal data, including names, telephone numbers or platform identifiers, delivery addresses, order histories and message content. In Nigerian deployments it must therefore be designed to comply with the Nigeria Data Protection Act 2023 (NDPA). The design adopts the following controls. (a) Lawful basis and transparency: the merchant, as data controller, identifies a lawful basis for each processing purpose (NDPA s. 25) and tells customers at the start of a conversation that they are interacting with an automated assistant, what data it uses, and how to reach a person. (b) Data minimisation: only the fields needed for the current turn are retrieved and supplied to the language model (s. 24(1)(c)); payment card details are not collected in the chat but handled by the merchant's payment provider. (c) Pseudonymisation before model calls: customer names, telephone numbers and addresses are replaced with internal tokens before prompts are sent to an external model provider and are restored only within the merchant's own systems (s. 39(2)(a)). (d) Security: data are encrypted in transit and at rest (s. 39(2)(b)); the store action layer uses least-privilege credentials limited to the operations it is permitted to perform; and every tool call is authorised against the verified customer identity so that one customer cannot read or change another customer's records. (e) Retention: transcripts and escalation tickets are kept only for a merchant-defined period tied to the processing purpose and are then deleted or anonymised (s. 24(1)(d)). (f) Cross-border transfer: where the model provider or hosting infrastructure is located outside Nigeria, the transfer must rest on one of the bases in Part VIII of the Act (ss. 41-43), and the provider's terms on data retention and use for model training must be reviewed. (g) Accountability: a data privacy impact assessment should be completed before deployment (s. 28), and personal data breaches must be notified to the Nigeria Data Protection Commission within 72 hours (s. 40(2)). Compliance with these controls will be checked during implementation through the adversarial journeys in Table 2 and an audit of the data actually sent to the model provider.

Technical requirements: A deployment requires (a) a WhatsApp Business Platform account, accessed through Meta's Cloud API or an approved business solution provider, and/or a Telegram bot; (b) a publicly reachable HTTPS endpoint to receive platform webhooks; (c) access to a language model, either through a hosted API or as a self-hosted open-weight model; (d) a vector index for catalogue descriptors; (e) a database for conversation state, identity links and tickets; and (f) an integration with the merchant's product, stock and order data. Staff need only a messaging account to receive and answer escalated conversations.

Financial requirements: Running costs consist of messaging-platform charges, model inference charges and hosting. Messaging charges are set by the platform operator and change over time; inference charges scale with conversation volume and with the amount of grounding evidence supplied per turn, which the minimisation rule above also limits. Because these prices change frequently, this article does not estimate a cost per conversation. Instead,

the implementation study will record messages, model tokens and hosting usage per journey so that the cost per resolved enquiry can be reported for the tested configuration.

Merchants without formal APIs: Many SMEs in the target market keep inventory in spreadsheets, in a messaging-app catalogue or informally (Section 5.5). The architecture accommodates them through the store adapter boundary at reduced capability, in three tiers. Tier 1, full integration: merchants using a hosted e-commerce platform or their own back end connect through its existing API, and the full architecture applies. Tier 2, structured data source: a spreadsheet or lightweight database maintained by the merchant serves as the system of record; an adapter exposes read operations for catalogue, price and stock, and records order requests that staff confirm rather than writing orders directly. Tier 3, read-only assistance: where stock and orders are not recorded in any machine-readable form, the assistant answers only from catalogue descriptions and policies supplied by the merchant, does not state stock levels or place orders, and escalates every purchase to staff with the customer's request already captured. The freshness guarantees of Section 3.4 hold only to the extent that the merchant keeps the underlying data current, a dependency that must be made explicit to merchants at onboarding.

Proposed Design Artefact and Evaluation Framework

This section presents the outputs of the design stage: the proposed architecture (Section 4.1) and the framework by which it will be evaluated (Section 4.2). Neither is an empirical finding. Because the system has not been implemented or tested, the descriptions below state what each component is designed to do; whether it does so in practice is the subject of the hypotheses in Section 4.2.

Proposed System Architecture

The proposed system comprises five layers. A message from a customer sent via WhatsApp or Telegram enters at the top and passes downward through these layers, exiting either as a response returned to the chat or as a ticket raised for a human agent (see Figure 1). Because each layer performs a single category of work and communicates through a standard internal format, substituting a messaging platform, a model, or the merchant's store back end is intended to require layer configuration rather than system redesign; this modularity has not yet been tested.

A single request typically touches most of the pipeline. If a customer asks whether an item is in stock, the messaging layer converts the platform event into an internal message, the orchestration layer reads the conversation so far and determines that store data is required, and the grounding layer fetches the product record and its current stock level. The answer is written from that record. If the customer then asks to buy, the orchestration layer repeats the item, quantity and price back and waits for confirmation before the store action layer writes anything. If the customer instead requests a refund, the escalation layer opens a ticket and a staff member continues in the same thread.

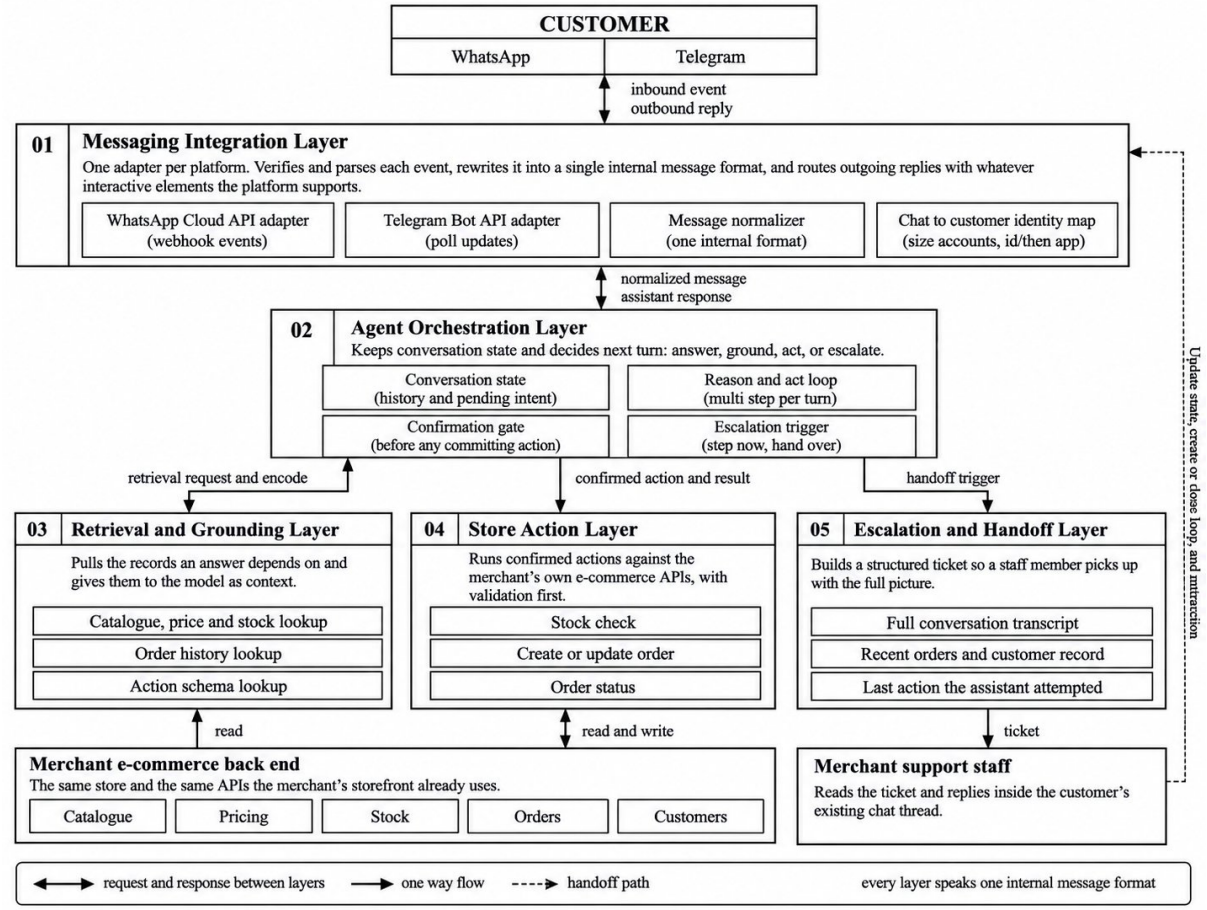


Figure 1: System architecture of the proposed assistant.

Messaging Integration Layer

Customer messages enter the system as webhooks from the WhatsApp Business Cloud API or as updates from the Telegram Bot API, with one adapter maintained per platform. The adapters verify that events are genuine, then parse and translate them into a single internal message format that the remainder of the system operates on. No component below this layer is aware of the message's origin.

This layer also associates a platform conversation with the internal customer record after the required identity linking has been established, so that a verified customer can retain continuity across supported channels. Replies are returned through the same adapter by which they were received. When the assistant emits structured content such as product cards or option lists, the adapter maps it to platform-supported controls and falls back to a plain-text representation when necessary.

Agent Orchestration Layer

The orchestration layer determines the next move in a conversation. It maintains a per-customer conversation state comprising prior messages, the task or intent the customer is pursuing, and any action awaiting customer agreement. It executes the reasoning-and-tool-use loop described in the literature review: reading state, selecting a move, executing it, and updating state.

Four moves are available: answer from the conversation, request store records from the grounding layer, invoke a store action, and escalate. Several moves may execute before anything is shown to the customer, for example retrieving a product record and then checking stock before quoting a delivery time.

Any action that commits the customer, such as placing an order, changing a quantity or cancelling, is not executed on the model's decision alone. The orchestration layer holds the action and states what is about to happen, including item, quantity and price, and asks the customer to confirm. This confirmation gate is intended to prevent a misinterpretation from becoming an incorrect order.

Retrieval and Grounding Layer

When a reply depends on store facts, the orchestration layer queries the retrieval and grounding layer, which resolves the relevant merchant records and supplies them to the model as evidence for that turn. Rather than recalling what the store sells, the model receives the current product, price, stock or order data needed for the response. Relatively stable catalogue descriptors may be retrieved through a semantic index, but volatile fields are refreshed from the merchant system before they are presented to the customer.

When an action is to be taken, the same layer is queried for interface details: which fields an order call expects and what values those fields accept. Retrieving the schema rather than allowing the model to assume it is expected to reduce failed calls; related work reports that supplying task-relevant API documentation improves API-call generation (Marketsmüller et al., 2026), but this effect has not been measured for the proposed system.

Store Action Layer

This layer executes merchant operations through the existing e-commerce API. State-changing operations such as placing, updating or cancelling an order are dispatched only after explicit confirmation. Read-only operations such as stock checks may also be exposed through this gateway when needed by an action workflow, but customer-facing transactional facts remain mediated by the retrieval and grounding layer so that the model receives validated evidence rather than raw tool assumptions.

Separating execution from orchestration is intended as a safety measure. The model does not write to the store; it proposes an action and arguments, and this layer validates them against the API contract before dispatch. Where validation fails, the failure is returned to the orchestration layer as a result to reason about rather than dispatched as a bad write.

Escalation and Hand-off Layer

Complaints, refund cases requiring staff authorisation, explicit requests for a person, repeated task failures, and requests outside the assistant's configured capabilities trigger escalation. Merchants may choose to automate routine refund-policy questions or eligibility checks, but operations outside the assistant's authorisation boundary are handed to staff. The resulting ticket includes the full transcript, relevant recent orders and the pending action or unresolved intent so that the human agent can continue from the established context.

The customer remains in the same messaging thread throughout; nothing moves to email or a separate helpdesk. Staff replies are delivered through the same adapter as assistant messages,

and the assistant remains silent in the thread until the ticket is resolved, which is intended to avoid ambiguity as to whether a person or a bot is responding.

Evaluation Framework and Hypotheses

The complete proposed system will be compared with three baselines and two ablation conditions: (1) a scripted rule-based chatbot; (2) a standalone LLM without retrieval grounding or tools; (3) a grounded, read-only LLM without state-changing tool access; (4) a tool-using agent with the confirmation gate disabled; and (5) a tool-using agent with confirmation enabled but without structured context transfer at escalation. The full system forms the reference condition. This staged design separates the contribution of grounding, controlled execution and context-preserving hand-off more clearly than a single aggregate comparison.

Table 3 specifies the metrics on which the comparison is to be conducted.

Table 3: Evaluation Metrics and Comparison Design

Metric	Definition	Instrument
Journey completion rate	Proportion of scripted journeys reaching the customer's stated goal without unnecessary human intervention	Scripted end-to-end journeys
Factual agreement	Proportion of assertions about products, prices, stock or orders matching the store record at the time of the turn	Automated comparison against back-end state
Action correctness	Proportion of committing actions executed with correct arguments and prior customer confirmation	Store transaction log audit
Escalation precision and recall	Whether escalation occurred at the correct point in journeys designed to require it, and did not occur where it was not required	Scripted failure journeys
Context preservation	Whether the escalated ticket contained the transcript, order history and pending action required for the staff member to continue without asking the customer to repeat information	Ticket inspection
Response latency	Elapsed time from inbound webhook to outbound reply	Instrumented pipeline timing

Note: *These metrics define the proposed future empirical evaluation; the present article does not report measured outcomes.*

The evaluation tests four hypotheses rather than reporting expected findings as results: H1, grounding improves factual agreement relative to an ungrounded LLM; H2, explicit confirmation and validation improve action correctness relative to an ungated tool-using agent; H3, structured hand-off improves context preservation relative to escalation without

state transfer; and H4, retrieval and validation introduce measurable latency. These hypotheses remain unverified until the implementation study is completed.

DISCUSSION

Because the architecture has not been implemented, this section discusses the expected consequences of the design choices rather than observed outcomes. Statements below about modularity, reliability, scalability and practical applicability are design rationales or hypotheses to be tested through the protocol in Section 4.2; none has yet been demonstrated.

Architectural Insights

Assigning each layer a single category of work and a standard internal message format is intended to localise the effect of change. In principle, adding a messaging platform should affect only the integration layer; substituting the underlying language model, only orchestration; and a change in the merchant's e-commerce API, only the grounding and store action layers. Whether this containment holds in practice will be known only when adapters for additional platforms and back ends are built. For a system intended to be deployed across merchants with heterogeneous back ends, this containment is a functional requirement rather than an aesthetic preference.

The validation boundary between orchestration and store action is designed to address the specific failure mode reported for tool-using agents. Because the model emits a proposed action rather than performing one, an invalid or misinterpreted request should be rejected and returned as a result the agent can reason about, instead of propagating into the merchant's order records. The extent to which this improves action correctness is tested by H2.

Grounding and Reliability

For volatile records, retrieval at the point of use means the assistant's statements can reflect the merchant state available at that turn. This matters where stale information is commercially costly, such as a changed price, depleted stock or an updated order status. Relatively stable catalogue descriptions may still be maintained in a semantic index for discovery, but volatile fields are refreshed before they are presented to the customer. Interface schemas are treated similarly: argument construction is grounded in the current API contract rather than assumed. Whether this yields higher factual agreement than an ungrounded model is tested by H1.

Escalation as a Design Requirement

Treating hand-off as a transfer of accumulated state rather than a routing decision is a continuity requirement of the proposed design. Prior research shows that human involvement can improve complaint recovery and that customers may prefer early, user-initiated transfer in some complaint contexts (Chacon & Martinez-Troncoso, 2026; Jeanpert et al., 2021). The architecture therefore preserves the transcript, relevant order context and pending intent at escalation. Keeping the customer in one thread and suspending assistant replies during human handling is also expected to reduce ambiguity about whether a person or a bot is responding. The effect on context preservation is tested by H3.

Scalability Considerations

The layered decomposition is intended to permit independent scaling of retrieval, generation and messaging components according to observed load, while stateless messaging adapters can be replicated horizontally. Caching must distinguish stable from volatile data: catalogue descriptions are suitable for a semantic index, whereas price, inventory and order status should be refreshed according to stricter freshness rules. Determining safe cache lifetimes and their performance trade-offs is deferred to the implementation study. No load testing has been carried out, so scalability remains a design expectation rather than a demonstrated property.

Limitations and Challenges

The principal limitation is that the architecture has not yet been implemented or empirically validated. The design arguments are derived from the literature and problem-domain analysis, so claims about the benefits of grounding, confirmation gating and context-preserving escalation remain hypotheses. The current manuscript should therefore be interpreted as a design and evaluation-protocol contribution rather than evidence that the proposed system outperforms existing approaches.

The design assumes that the merchant operates an e-commerce back end exposing an API for catalogue, stock and order operations. Many SMEs in the target market do not; they maintain inventory informally, in spreadsheets or in the proprietor's memory. For such merchants the store action layer has nothing to call, and the system's applicability is correspondingly reduced. Section 3.6 describes reduced-capability deployment tiers for such merchants, but their usefulness has not been evaluated.

The system depends on third-party messaging platforms whose policies, rate limits and interactive message capabilities are outside the control of the implementer and subject to change. Platform policy on automated messaging in particular may constrain permissible assistant behaviour.

Language coverage is a further constraint. Although the design is intended to support multilingual operation, retrieval quality and generation quality in Nigerian languages and in code-mixed registers common in the target market are unverified, and the embedding models typically used for retrieval are trained predominantly on English text.

LLM behaviour is also model- and configuration-dependent. A publication-quality implementation study must report the exact model version, prompts, retrieval settings, tool schemas and repeated-trial procedure, and should include adversarial cases such as prompt injection, malformed tool arguments, replayed platform events and cross-customer data-access attempts. The production architecture should enforce least-privilege credentials, per-customer authorisation checks and auditable tool execution.

Finally, handling customer order and contact records raises privacy, security and governance obligations. For Nigerian deployments, processing must be designed in accordance with the Nigeria Data Protection Act 2023 and relevant regulatory guidance (Nigeria Data Protection Act, 2023). Section 3.6 sets out the controls the design adopts for lawful basis, minimisation, pseudonymisation, security, retention and cross-border transfer. Their adequacy, and the specific retention periods and transfer arrangements for a given merchant, must be confirmed through a data privacy impact assessment before deployment.

Implications for Research and Practice

For SME operators, the design suggests that automation of messaging-channel support may not need to force a choice between a rigid script and an unreliable general-purpose assistant. An assistant that reads from the store's own records and defers committing actions to explicit confirmation could, if the evaluation confirms the hypothesised gains in factual agreement and action correctness, absorb routine enquiry volume, including availability, pricing, delivery time and order status, while leaving complaints and exceptions to staff.

For the design of chat-commerce tooling more generally, the article's position is that grounding and hand-off quality are not features to be added after a conversational core is working, but constraints that determine the shape of the architecture from the outset. Retrofitting a confirmation gate or a context-carrying escalation path onto a system whose model writes directly to the store is likely to be harder than designing the boundary in, although this argument rests on design reasoning rather than measured engineering effort.

For research, the evaluation protocol proposed here uses scored end-to-end journeys rather than only single-turn benchmarks. This directly targets the complete-conversation failure mode highlighted in evaluations of tool-augmented conversational agents (Arcadinho et al., 2024).

CONCLUSION

SMEs selling through messaging applications continue to face fragmented support workflows and inconsistent access to current business information. LLM-based agents create an opportunity to improve these interactions, but a commerce setting requires stronger controls than conversational fluency alone: customer-facing facts must be grounded in merchant data, state-changing actions must cross a validation and confirmation boundary, and human escalation must preserve the state already established in the conversation.

This article presents a five-layer architecture built around those requirements. The design separates messaging integration, orchestration, retrieval grounding, store operations and escalation; refreshes volatile merchant data at the point of use; validates tool arguments before execution; and transfers transcript, transactional context and pending intent during hand-off. It also specifies an evaluation protocol based on end-to-end journeys rather than single-turn prompts.

The main research contribution is the integration of these controls into one reference design and an evaluation plan that can isolate their effects. The literature indicates that tool-augmented agents can struggle over complete conversations, while conversational-commerce work has not yet established a standard architecture that combines live store grounding, controlled actions and context-preserving escalation. The proposed design addresses that intersection without claiming empirical superiority before testing.

The expected benefits of the design (more accurate answers, fewer incorrect orders, smoother hand-off, and easier substitution of platforms, models and back ends) are hypotheses derived from the literature and the design reasoning. None has been demonstrated. The architecture's reliability, scalability and practical applicability for SMEs, including those without formal APIs, remain to be established through implementation and evaluation.

The next stage is to implement the architecture and execute the specified evaluation protocol under controlled and reproducible conditions. Subsequent work should test multilingual and code-mixed customer language, merchants without formal e-commerce back ends, security against adversarial inputs, and the trade-off between data freshness and system latency.

Declarations

Author Contributions

To be confirmed by the authors before submission.

Funding

Funding information was not provided in the source manuscript and should be confirmed by the authors before submission.

Ethics Approval

Not applicable. The present study proposes a system architecture and evaluation protocol and does not report research involving human participants.

Informed Consent

Not applicable.

Data Availability Statement

No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflict of Interest

The authors declare no conflict of interest.

Acknowledgments

No acknowledgments were provided in the source manuscript.

AI Use Declaration

The authors declare that OpenAI ChatGPT was used during the preparation of this manuscript for language revision, citation-format checking, and manuscript formatting. The authors reviewed and verified all AI-assisted output and remain responsible for the accuracy, originality, and integrity of the manuscript.

REFERENCES

Alabi, O. A., Okeke, N. I., Igwe, A. N., Ofodile, O. C., & Ewim, C. P.-M. (2024). Omni-channel customer experience framework: Enhancing service delivery in SMEs. *World Journal of Advanced Research and Reviews, 24*(2), 655–670. https://doi.org/10.30574/wjarr.2024.24.2.3335

Arcadinho, S., Aparicio, D., & Almeida, M. S. C. (2024). Automated test generation to evaluate tool-augmented LLMs as conversational AI agents. In *Proceedings of the*

2nd GenBench Workshop on Generalisation (Benchmarking) in NLP (pp. 54–68). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.genbench-1.4

Chacon, A., & Martinez-Troncoso, C. (2026). Chatbot or human? Rethinking complaint handling in customer service. *Journal of Consumer Behaviour, 25*(5), 2608–2628. https://doi.org/10.1002/cb.70185

Febriani, A., Sopha, B. M., & Wibisono, M. A. (2025). Unlocking omnichannel success for small and medium-sized enterprises: A systematic review of technologies, barriers, and enablers. *Journal of Industrial Engineering and Management, 18*(3), 553–576. https://doi.org/10.3926/jiem.8661

Gerea, C., & Herskovic, V. (2022). Transitioning from multichannel to omnichannel customer experience in service-based companies: Challenges and coping strategies. *Journal of Theoretical and Applied Electronic Commerce Research, 17*(2), 394–413. https://doi.org/10.3390/jtaer17020021

GSMA. (2023). *E-commerce in Africa: Unleashing the opportunity for MSMEs*. GSMA Mobile for Development. https://www.gsma.com/solutions-and-impact/connectivity-for-good/mobile-for-development/gsma_resources/e-commerce-in-africa-unleashing-the-opportunity-for-msmes/

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly, 28*(1), 75–105. https://doi.org/10.2307/25148625

Jeanpert, S., Jacquemier-Paquin, L., & Claye-Puaux, S. (2021). The role of human interaction in complaint handling. *Journal of Retailing and Consumer Services, 62*, 102670. https://doi.org/10.1016/j.jretconser.2021.102670

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in Neural Information Processing Systems 33* (pp. 9459–9474). Curran Associates.

Liu, J., Song, K., Kang, Y., He, G., Jiang, Z., Sun, C., Lu, W., & Liu, X. (2021). A role-selected sharing network for joint machine-human chatting handoff and service satisfaction analysis. In M.-F. Moens, X. Huang, L. Specia, & S. W. Yih (Eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (pp. 9731–9741). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.767

Marketsmüller, M., Martin, S., & Schlippe, T. (2026). *Evaluating retrieval-augmented generation variants for natural language-based SQL and API call generation* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2602.07086

Masterman, T., Besen, S., Sawtell, M., & Chao, A. (2024). *The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: A survey* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2404.11584

Mhiribidi, T. C., Chigodo, E. T., & Jowa, V. (2024). Conversational commerce: Designing and evaluating a WhatsApp-based e-commerce chatbot. In *2024 3rd Zimbabwe Conference of Information and Communication Technologies (ZCICT)* (pp. 174–181). IEEE. https://doi.org/10.1109/ZCICT63770.2024.10958353

Nigeria Data Protection Act, 2023, Act No. 37 (2023) (Nigeria).

Odubo, A., & George, Y. E. (2026). Social commerce adoption on TikTok versus Instagram: A comparative analysis of firms in Bayelsa State, Nigeria. *World Journal of Entrepreneurial Development Studies, 11*(3), 132–142.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems, 24*(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Shopify Staff. (2023, February 21). *What is conversational commerce?* Shopify. https://www.shopify.com/blog/what-is-conversational-commerce

Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., Zhang, M., Wang, J., Jin, S., Zhou, E., Zheng, R., Fan, X., Wang, X., Xiong, L., Zhou, Y., Wang, W., Jiang, C., Zou, Y., Liu, X., … Gui, T. (2025). The rise and potential of large language model based agents: A survey. *Science China Information Sciences, 68*(2), Article 121101. https://doi.org/10.1007/s11432-024-4222-0

Xu, Z., Cruz, M. J., Guevara, M., Wang, T., Deshpande, M., Wang, X., & Li, Z. (2024). Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery. https://doi.org/10.1145/3626772.3661370

Yao, S., Shinn, N., Razavi, P., & Narasimhan, K. (2025). τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *Proceedings of the 13th International Conference on Learning Representations* (pp. 9965–10017). https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b126cc38b8638e07bef37e7b2bb72bf-Abstract-Conference.html