

Deploying LLM Agents for Auditable Complaint Investigation in Banking

John Hartley* Conor B. Hamill Azam Din Fayz Siddiqui
Kingsley Nketia Acheampong Greig A. Cowan Raad Khraishi

NatWest AI Research

Abstract

Complaint investigation in retail banking requires handlers to synthesise evidence from heterogeneous enterprise systems into policy-aligned summaries. We present a production-deployed tool-augmented LLM agent at a major UK retail bank that assists this process under human review. Given a complaint record, the agent queries 16 typed, schema-validated tools and returns an investigation summary with a source-cited chronological evidence timeline. We evaluate the system using SME assessment, LLM-as-a-Judge monitoring, rework measurement, and downstream impact testing. On 187 paired complaints, SMEs judged the agent to match or exceed historical human-written production investigations in 90% of cases across faithfulness, relevance, completeness, and coherence. Median investigation time falls from 30 minutes to 23 seconds. Operational controls include daily automated monitoring, pre-defined thresholds, hallucination flagging, and periodic human audit. These results demonstrate that constrained single-agent LLMs with typed tools are viable for scalable, auditable complaint investigation in regulated environments.

1 Introduction

When a customer complains to a retail bank, a complaint handler must investigate the circumstances by gathering evidence from multiple internal enterprise systems, reconciling structured records with unstructured artefacts such as call transcripts, and producing a policy-aligned summary. UK banks must resolve complaints and report outcomes to the Financial Conduct Authority under its Dispute Resolution sourcebook (Financial Conduct Authority, 2024). This process is labour-intensive: handlers navigate up to 50 systems, review lengthy audio transcripts, and synthesise findings into a coherent

narrative. Scale and heterogeneity make investigations time-consuming and cognitively demanding.

Progress in large language models (LLMs) expands the capacity of financial institutions to process information. Increasing context windows allow models to incorporate substantial volumes of multi-modal data (Liu et al., 2024; Hankache et al., 2025; OpenAI, 2025a; Comanici et al., 2025). Agentic approaches extend model capabilities by combining reasoning with tool-use for retrieval and action (Yao et al., 2023). Financial services are exploring agentic capabilities across simulation, analysis, and operational workflows (Batra et al., 2025; Joshi, 2025). However, little published work describes productionised LLM agents that retrieve evidence across heterogeneous enterprise systems (Wang et al., 2024; Chen et al., 2024) in regulated environments while maintaining source-level audibility, human review, and continuous operational monitoring.

We present a single-agent system deployed in production at a major UK retail bank that automates complaint investigation. Our contributions are:

1. We define the complaint investigation task in retail banking and present a single-agent architecture that conducts read-only, cross-system evidence retrieval through 16 typed tools, producing policy-aligned summaries and chronological evidence timelines.
2. We describe a layered evaluation and monitoring framework combining SME assessment, LLM-as-a-Judge, human rework measurement, and downstream impact testing, designed for continuous operation in a regulated environment.
3. We report production deployment results showing the agent matches or exceeds human performance in 90% of cases on an expert-rated sample, while reducing median investigation time from 30 minutes to 23 seconds.

*Corresponding author: john.hartley@natwest.com

2 Task Definition

Problem. Let C denote a structured customer complaint record and let $\mathcal{D} = \{D_1, \dots, D_n\}$ denote the set of internal enterprise systems available to the agent through read-only tools. The task is to generate a structured output

$$Y = (S, F)$$

conditioned on C and tool-retrieved evidence from $\mathcal{D}$. Here, S is a policy-aligned investigation summary and $F = (f_1, \dots, f_m)$ is a chronologically ordered sequence of schema-constrained facts.

Input. Each complaint record C contains a complaint identifier, customer identifier, complaint category, free-text issue description, and receipt date. The initial deployment covers eight high-volume complaint categories, including debt management, chargebacks, fraud, direct debits, and account applications.

Enterprise systems. The agent accesses $n = 16$ enterprise systems through read-only tools. These systems include call transcripts, customer event logs, transaction records, knowledge base articles, KYC records, debit card disputes, and direct debit records, corresponding to the sources most frequently used in complaint handling.

Output. The agent returns a Pydantic-structured object $Y = (S, F)$, where S is a narrative investigation summary and F is an ordered collection of schema-validated facts. Each fact $f_i \in F$ contains typed fields, including a single past-tense factual statement, the originating system, a reference identifier, and an associated date. Facts are ordered chronologically by their associated dates. We use “auditable” to mean that generated facts include identifiers sufficient for a handler or auditor to trace the claim back to enterprise records. An anonymised example is shown in Table 1, and the full schema is provided in Appendix D.

3 Related Work

Tool-augmented LLM agents. Retrieval-augmented generation grounds model outputs in external evidence sources (Lewis et al., 2020). Complementarily, tool-augmented LLM agents interleave reasoning with tool and API use (Yao et al., 2023), including program execution (Gao et al., 2023) and interaction with large API

Table 1: An anonymised example of agent output for a Fixed Rate ISA complaint. The summary and evidence timeline are generated as a single structured object and rendered via a template.

Investigation Summary
The customer raised concerns regarding the maturity date and fixed-term conditions of their Fixed Rate ISA, particularly in relation to when the fixed term begins. During several staff interactions, the maturity date and fixed-term details were explained. Product T&Cs clarify that...
Evidence Timeline
- On 04-06-2025, the customer stated frustration regarding ISA transfers and challenged staff on the accuracy of their technical advice. (Ref: 24000056147227, System: Call Transcripts) ...

surfaces (Schick et al., 2023; Patil et al., 2024; Qin et al., 2024).

Wang et al. (2024) present RCAgent, a tool-augmented LLM agent framework deployed for cloud root cause analysis (RCA) that autonomously collects data across internal systems using self-consistency mechanisms for action trajectories. Similarly, Chen et al. (2024) introduce RCACopilot, which performs RCA of cloud incidents and produces explanatory narratives. Both demonstrate productionised LLM agents querying heterogeneous data sources in the infrastructure reliability domain. We adapt these principles to customer complaint investigation workflows in retail banking, where regulatory requirements impose additional constraints on compliance, evidence provenance, and human oversight.

NLP and LLM agents in financial services. Recent work has applied agentic LLMs to financial-services tasks such as economic simulation, trading, fraud and retail banking (Li et al., 2024; Yu et al., 2024, 2025; Bai et al., 2025; Chua et al., 2025), with broader surveys documenting LLM adoption in finance (Batra et al., 2025; Joshi, 2025). In complaint handling, prior work has focused on classification (Jin and Aletras, 2021; Singh et al., 2023; Roumeliotis et al., 2025), zero-shot categorisation (Hayawi and Shahriar, 2025), summarisation (Feigenblat et al., 2021), and letter writing (Batra et al., 2026). In contrast, we study end-to-end tool-equipped investigation using a constrained single-agent architecture with schema-validated tools, policy grounding, operational risk controls, and auditable evidence trails.

LLM-as-a-Judge evaluation. LLM judges are used to evaluate text (Zheng et al., 2023) but are known to exhibit self-preference, position bias, and correlated errors across models (Panickssery et al., 2024; Shi et al., 2025; Kohli, 2026). We therefore use them only as one signal among SME evaluation, rework measurement, downstream testing.

4 Methods

4.1 Agent design

We use a single-agent architecture to reduce coordination overhead and avoid the compounding failures observed in multi-agent systems (Cemri et al., 2026). This design limits the surface area for failure while retaining sufficient capability for complaint investigation. The agent is implemented using the OpenAI Agents SDK (OpenAI, 2025b) and targets GPT-4.1 (OpenAI, 2025a).

Given a complaint record, the agent iteratively identifies missing information, selects an appropriate tool, and retrieves operational evidence and policy guidance into a private local context. Once sufficient evidence has been gathered, it generates a policy-grounded investigation summary and a chronological evidence timeline. The model determines which tools to call, while the application layer controls how those calls are executed by enforcing typed function signatures, schema validation, identifier injection, and per-call and cumulative token budgets. Figure 1 summarises the system architecture.

Tool-based access. We considered supplying all available data directly in the prompt, but use tool-based access for three reasons. First, many complaints require evidence from only a small subset of systems. Second, indiscriminate data inclusion increases the model’s cognitive load and the risk of exceeding the context window. Third, as coverage expands toward 50 or more data sources, tools provide a more modular and extensible interface. Tool-based access also enables adaptive investigation: later tool choices can depend on evidence retrieved earlier in the trajectory. Conflicting records are surfaced rather than reconciled. The agent does not adjudicate between sources, and each fact carries its own system, reference identifier, and date, so contradictions appear side by side for the handler to resolve.

Model selection. GPT-4.1 was selected for its steerability, instruction-following performance on

Scale’s MultiChallenge benchmark (Scale AI, 2025), and one-million-token context window. Reasoning models from the o-series were considered, but their smaller context windows are less suitable for investigations that may accumulate approximately 120k tokens of evidence. We use GPT-4.1-nano as a lightweight relevancy filter because it offers lower latency and cost for row-selection tasks.

4.2 Policy-grounded summaries

The policy retrieval-augmented generation (RAG) API is exposed to the agent as the Knowledge Base tool. The agent submits natural-language queries, which are embedded using snowflake-arctic-embed-l-v2.0 (Hugging Face, 2025). The API retrieves the top 20 articles by cosine similarity over article titles. We do not apply a separate re-ranking model; instead, the agent can refine its queries iteratively. The knowledge base is the bank’s live policy corpus rather than a snapshot: documents are updated as new policies are issued, so currency is maintained at source by the policy owners rather than detected downstream. Retrieved snippets are cached in local context and cited verbatim to support traceability for audit and human handler understanding.

4.3 Tools

The system provides read-only access to 16 back-office and operational enterprise systems through tools annotated with the `function_tool` decorator from the OpenAI Agents SDK. Each tool includes a docstring that acts as a data dictionary for available fields and their meanings. Tool outputs are serialised as XML. To prevent cross-complaint data leakage, sensitive identifiers are injected by the application layer rather than supplied by the model. Tool exceptions are logged, and tools return a standardised `No data found` message when no matching records are available. Appendix C provides pseudocode for a representative data tool.

Relevancy filter. When a serialised tool output exceeds a configured token threshold, a GPT-4.1-nano filter selects the rows most relevant to the complaint issue description. The filter runs for a bounded number of passes, each targeting a per-iteration token budget. Unfiltered outputs are retained in the execution trace for auditability. Tools also enforce per-call and cumulative token limits, raising explicit errors when limits are exceeded.

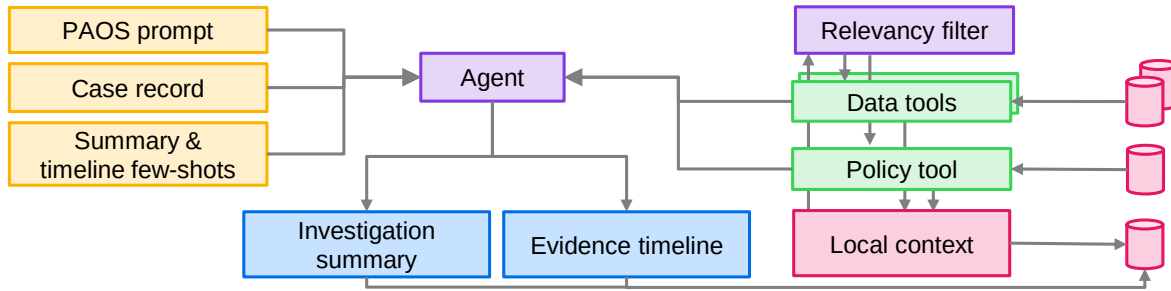


Figure 1: System overview. Complaint records enter an agent loop that coordinates tool calls and plan-act-observe-summarise (PAOS) prompt. The policy tool returns RAG-based guidance, while data tools return typed operational evidence. A relevancy filter extracts complaint-relevant rows when tool outputs exceed the token budget. The agent stores raw evidence in local context and outputs a structured investigation summary with a chronological evidence timeline.

4.4 Prompts and outputs

Prompts follow the GPT-4.1 prompting guide (MacCallum and Lee, 2025). We use a hierarchical Markdown template that separates task instructions, workflow steps, output requirements, and category-specific few-shot exemplars. Jinja2 (Jinja, 2025) renders prompts using complaint-category parameters, with strict compatibility checks to prevent malformed prompts. The user prompt includes concise tool descriptions to support tool selection, and SME-validated exemplars are filtered by complaint category. The final output is enforced through Pydantic structured decoding, as defined in Section 2.

4.5 Portability to other domains

The architecture separates domain-specific components from reusable infrastructure. Only the tool implementations and the policy corpus are specific to banking. The remaining design: typed read-only tools with docstring data dictionaries, application-layer identifier injection, relevancy filtering under a token budget, and schema-constrained output carrying per-fact provenance transfer directly. Domains that share the same structural properties, namely heterogeneous record systems and a mandated audit trail (e.g., insurance claims handling, public-sector casework), can instantiate the pattern by supplying new tools and a domain-appropriate policy corpus.

4.6 Deployment

The system is deployed as a containerised, stateless microservice in the bank’s cloud environment. Figure 2 illustrates the end-to-end data flow. Complaint handlers interact with the system through a web interface, which communicates with a Python microservice. This front-end service publishes in-

vestigation requests to a Kafka topic, decoupling the user-facing layer from the agent back-end and allowing the system to scale horizontally with investigation volume.

The agent microservice consumes messages from the Kafka topic and executes the investigation loop. Within this service, the agent orchestrates tool calls that query source-system APIs through a layered integration architecture of authorisers, typed API clients, and data loaders. Investigation state and outputs are persisted to a PostgreSQL database, which also serves the front-end for status tracking and result display. All LLM calls are routed through the bank’s internal generative-AI gateway under a zero-data-retention policy, ensuring that complaint data is not persisted by the model provider.

Because the system augments rather than replaces human judgment, its integration into existing workflows requires targeted handler training. Training covers the system’s capabilities and limitations, hallucination risks, and best practices for writing detailed issue descriptions that improve investigation quality.

5 Evaluation

Evaluation in regulated production systems requires multiple complementary signals rather than a single aggregate metric. We therefore combine SME quality assessment, LLM-as-a-Judge monitoring, downstream impact testing, rework measurement, operational reliability metrics, and responsible-AI checks.

5.1 SME evaluation

We sampled 187 complaints uniformly at random from a held-out test set. Eighteen internal com-

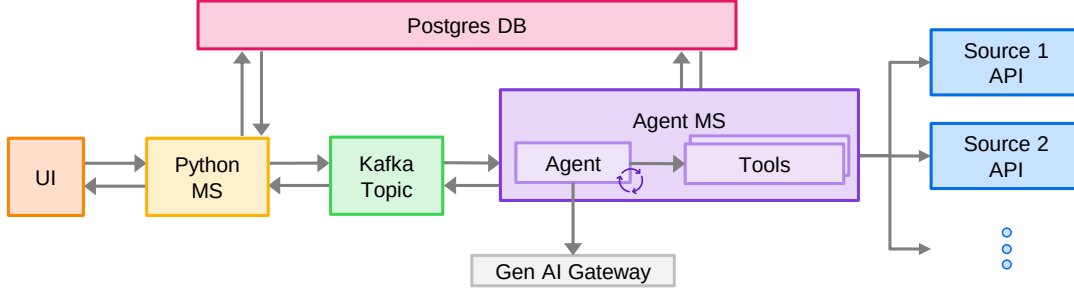


Figure 2: Deployed system architecture. The web UI submits requests via a Python microservice to a Kafka topic. The agent microservice consumes messages, orchestrates tool calls to source-system APIs, and persists results to a PostgreSQL database. LLM inference is routed through the bank’s generative-AI gateway.

plaint handlers with relevant subject-matter expertise participated in the evaluation. Evaluators were active complaint handlers and were assigned only to complaint categories within their area of expertise. Before evaluation, they attended a calibration workshop using annotated examples of the rating criteria.

For each case, one category-specific SME independently scored a historical human-written production summary and an agent-generated summary produced from the same complaint record and enterprise systems. Summaries were shown in random order, and evaluators were blinded to whether each summary was written by a human handler or generated by the agent. Evaluators saw only the investigation summary S , not the agent evidence timeline F , because no equivalent historical baseline timeline was available.

Evaluators rated faithfulness, relevance, and completeness on a binary scale, and coherence on a four-point scale (see Table 6 in Appendix E). We report the proportion of cases in which the agent score was higher than, equal to, or at least equal to the historical human score.

Table 2 shows that agent-generated summaries scored at least as highly as historical human-written production summaries in approximately 90% of paired comparisons across all four dimensions. Cases where the agent underperforms are concentrated among complaints with low-quality intake descriptions and complex multi-system queries that exceed the effective context window.

5.2 LLM-as-a-Judge

We adopt an LLM-as-a-Judge framework (Zheng et al., 2023) for prompt iteration and continuous quality monitoring, focusing on faithfulness and

Table 2: Blinded SME paired comparison evaluation (%). Proportion of cases in which the agent-generated summary scored higher than, equal to, or at least equal to the historical human-written production summary across 187 complaints. Each complaint was reviewed by one category-specific SME. Brackets show 95% confidence intervals computed using bootstrapping.

Metric	> human	= human	≥ human
Faithfulness	24.6 [18, 31]	65.8 [59, 73]	90.4 [86, 95]
Relevance	18.2 [13, 24]	70.6 [64, 77]	88.8 [84, 93]
Completeness	26.7 [20, 33]	63.6 [57, 71]	90.3 [86, 95]
Coherence	26.7 [20, 33]	63.1 [56, 70]	89.8 [86, 94]

completeness. Judge calibration combines SME-validated positive examples with synthetic negative instances generated by perturbing citations, numerical values, and dates, or by removing critical evidence. Prompts are optimised to maximise agreement with SME assessments, measured using Cohen’s κ .

In production, the LLM-as-a-Judge is used for regression detection via seven-day rolling averages. To mitigate self-preference bias, evaluations are performed using multiple models (GPT o1 and GPT o4-mini) (Panickssery et al., 2024). As shown in Table 3, κ scores range from 0.70 to 0.81, indicating substantial to near-perfect agreement (McHugh, 2012).

Table 3: Cohen’s κ between SME ratings and LLM-as-a-Judge evaluations for faithfulness and completeness.

Model	Faithfulness	Completeness
GPT-o1	0.73	0.70
GPT-o4-mini	0.81	0.70

5.3 Downstream impact

After human review, summaries are passed to a downstream system that drafts customer letters. We compared 319 paired letters from agent-produced and human-produced summaries using the downstream system’s existing 12 LLM-as-a-Judge quality metrics (see Table 5 in Appendix A). We treat this analysis as a regression check rather than an independent measure of investigation quality. Paired t-tests showed no statistically significant difference on 11 metrics and a statistically significant 1% difference on one metric, suggesting that agent-generated investigations did not materially degrade downstream letter quality.

5.4 System efficiency

We report efficiency using three samples: 187 complaints for SME quality assessment, 319 for downstream-impact and token analysis, and 1,234 for timing analysis. Sample sizes differ because SME assessment is labour-intensive, whereas timing data is logged automatically for every investigation.

Across 1,234 deployed investigations, the agent completes a complaint in a median of 23 seconds, compared with approximately 30 minutes for human handlers using the same systems. The 95th-percentile duration is 95 seconds (see Figure 3a in Appendix B).

The agent uses a median of approximately 120k total tokens per complaint, and the 95th-percentile is 318k tokens (see Figure 3b in Appendix B). Although GPT-4.1 benchmarks report a 33% performance degradation on latent needle-in-a-haystack tasks beyond this range (Modarressi et al., 2025), SME validation shows that the agent matches or exceeds human-written investigations in approximately 90% of cases. This suggests that generic long-context retrieval benchmarks may not directly transfer to structured complaint investigation, where evidence is organised by source, date, and schema.

5.5 Tool call analysis

Table 4 shows tool invocation counts per investigation, conditional on at least one call. Tool-use patterns vary by source type: large self-contained artefacts such as call transcripts are fetched once, structured operational data is queried sparingly. Policy guidance is queried iteratively as the agent refines its understanding of the applicable guidance.

The absence of long invocation tails suggests that typed interfaces, schema validation, token limits, and prompt guidance help prevent degenerative tool-use loops.

Table 4: Distribution of tool invocation counts per investigation for selected tools, conditional on at least one invocation.

Tool name	1	2	3	4
Call Transcripts	100.0%	0.0%	0.0%	0.0%
Knowledge-Base	30.5%	5.2%	39.4%	24.9%
Credit Card Transactions	83.9%	12.5%	3.6%	0.0%

5.6 Failure modes and operational monitoring

Deployment testing showed a 1% tool-call failure rate, primarily caused by read timeouts. The agent retries failed calls up to five times; among initially failed calls, more than 95% succeed on retry. Each complaint investigation is also subject to a five-minute timeout to prevent runaway execution.

Output controls. Every agent-generated summary is reviewed by a human handler before it enters the complaint record. To support this review, the system flags two explicit failure modes: low faithfulness and tool failure. The interface displays a faithfulness score, with amber alerts prompting additional scrutiny. When a tool fails or times out, the agent returns a partial investigation that explicitly discloses which enterprise systems could not be reached.

Incompleteness is not silent. Human handlers understand what a complete investigation requires for a given complaint type and rework the output with missing information rather than accepting it at face value.

Handlers also provide structured feedback to the business-as-usual (BAU) team, which uses it to refine prompts. Periodic independent quality assurance (QA) audits assess ongoing reliability and alignment with business expectations.

Continuous monitoring. A daily pipeline evaluates all complaints using LLM-as-a-Judge metrics, averaged over seven-day rolling windows. Amber and red thresholds, calibrated from validation confidence intervals, trigger root-cause investigation and re-prompting or rollback, respectively. Generation time is monitored separately with four-minute amber and five-minute red alerts.

Rework measurement. Rework is measured as the normalized Levenshtein distance between the agent output and the human-adjusted version saved after review. This character-level measure captures both substantive corrections and minor stylistic edits without requiring semantic alignment. Seven-day rolling averages are monitored against predefined limits, providing an early warning signal for prompt or model degradation.

Input validation. During tool execution, Pydantic models validate all input arguments, including customer identifiers and account numbers. Heuristic checks verify that specified accounts belong to the relevant customer. When validation fails, the error is returned to the agent, allowing a corrected retry. All validation failures are logged and monitored for upstream data-quality trends.

5.7 Responsible AI

We assessed SME evaluation results by customer gender and nationality and found no statistically detectable differences in sample proportions across these groups. The system was reviewed under internal governance processes designed to support EU AI Act readiness (European Parliament, 2024). A privacy impact assessment confirmed GDPR compliance, and a Consumer Duty assessment was completed within the retail banking governance framework.

6 Conclusion

We presented a production-deployed agentic system for complaint investigation in UK retail banking. The system orchestrates 16 typed, read-only data tools to produce policy-aligned summaries and source-cited evidence timelines for human review.

In a blinded SME evaluation, agent-generated summaries matched or exceeded historical human-written production summaries in approximately 90% of paired comparisons. Median investigation time fell from approximately 30 minutes to 23 seconds, with no material downstream degradation in customer-letter quality.

The deployment combines constrained tool use, schema-validated outputs, source-level metadata, handler feedback, and LLM-as-a-Judge monitoring. These results suggest that tool-augmented single-agent LLMs can support regulated, data-intensive complaint-investigation workflows, subject to current limits in tool and complaint-category coverage.

7 Limitations

The LLM-as-a-Judge evaluation uses two GPT-family models to assess a GPT-4.1 agent, reducing direct self-preference bias (Panickssery et al., 2024). We mitigate the risk of other LLM judge biases by using LLM judges as one monitoring signal among several controls rather than as the sole quality measure. The primary assessment of quality is performed by SMEs.

The demographic fairness analysis covers gender and nationality only. In a UK retail banking context, age and financial vulnerability indicators are arguably more relevant fairness axes and should be evaluated in future work.

The evaluation uses proprietary baselines and datasets from a UK retail banking environment. Due to regulatory, confidentiality, and commercial-sensitivity constraints, we cannot release the data, all enterprise systems, provide a reproducible benchmark, provide a detailed taxonomy of failure modes, or disclose absolute values for some business-impact results. These absolute values were available during internal evaluation. However, the architectural patterns, tool pseudocode (see Listing 1 in Appendix C), and monitoring thresholds described here are intended to support replication in similar regulated environments.

Acknowledgements

We acknowledge the Retail Complaints team for their domain expertise and insightful discussions on business logic. We also thank the SMEs who participated in multiple testing rounds and provided valuable feedback for model improvement. We would like to extend our appreciation to all colleagues who contributed to the development of the project. Finally, we extend our appreciation to Greig Cowan, Head of Data Science, and Graham Smith, Head of Data Innovation, and for their support and for enabling this work.

Ethical considerations

The system was developed and deployed within an existing UK retail banking governance framework. It is used to support complaint investigation, not to make autonomous customer-outcome decisions. All outputs are reviewed by trained complaint handlers before being entered into the complaint record or used downstream.

The agent operates through read-only tools, with data access logged and routed through the organ-

isation’s approved generative-AI gateway under a zero-data-retention policy. Sensitive identifiers required for tool execution are managed by the application layer. Generated outputs include source system names, reference identifiers, and dates to support traceability and audit.

The deployment was reviewed through internal privacy, data-protection, responsible-AI, and consumer-governance processes. Ongoing oversight combines automated quality monitoring, structured handler feedback, and periodic audit to ensure continued alignment with operational and regulatory expectations.

References

- Songran Bai, Bingzhe Wu, Yiwei Zhang, Chengke Wu, Xiaolong Zheng, Yaze Yuan, Ke Wu, and Jianqiang Li. 2025. *Auditagent: Expert-guided multi-agent reasoning for cross-document fraudulent evidence discovery*. In *Proceedings of the 6th ACM International Conference on AI in Finance, ICAIF ’25*, page 1–9, New York, NY, USA. Association for Computing Machinery.
- Devesh Batra, Alexandros Anatolakis, John Hartley, Jude King, Greig A Cowan, and Raad Khraishi. 2026. *Operationalising LLMs for compliance-critical letter writing in financial services*. In *NeurIPS 2025 Workshop: Generative AI in Finance*.
- Devesh Batra, Conor Hamill, John Hartley, Ramin Okhrati, Dale Seddon, Harvey Miller, Raad Khraishi, and Greig Cowan. 2025. A review of llm agent applications in finance and banking. *Available at SSRN 5381584*.
- Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2026. *Why do multi-agent LLM systems fail?* In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Yinfang Chen, Huaibing Xie, Minghua Ma, Yu Kang, Xin Gao, Liu Shi, Yunjie Cao, Xuedong Gao, Hao Fan, Ming Wen, Jun Zeng, Supriyo Ghosh, Xuchao Zhang, Chaoyun Zhang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Tianyin Xu. 2024. *Automatic root cause analysis via large language models for cloud incidents*. In *Proceedings of the Nineteenth European Conference on Computer Systems, EuroSys ’24*, page 674–688, New York, NY, USA. Association for Computing Machinery.
- Xin Jie Chua, Jeraelyn Ming Li Tan, Jia Xuan Tan, Soon Chang Poh, Yi Xian Goh, Debbie Hui Tian Choong, Foong Chee Mun, Sze Jue Yang, and Chee Seng Chan. 2025. *Banking done right: Redefining retail banking with language-centric AI*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 646–658, Suzhou (China). Association for Computational Linguistics.
- Gheorghe Comanici, Eric J. Bieber, Mike Schaeckermann, Ice Pasupat, Naveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3279 others. 2025. *Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities*. *ArXiv*, abs/2507.06261.
- Council of the European Union European Parliament. 2024. Artificial intelligence act (regulation (eu) 2024/1689), official journal version of 13 june 2024. <https://artificialintelligenceact.eu/the-act/>.
- Guy Feigenblat, Chulaka Gunasekara, Benjamin Szneider, Sachindra Joshi, David Konopnicki, and Ranit Aharonov. 2021. *TWEETSUMM - a dialog summarization dataset for customer service*. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 245–260, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Financial Conduct Authority. 2024. Dispute resolution: Complaints (DISP). <https://www.handbook.fca.org.uk/handbook/DISP/>.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning, ICML ’23*. JMLR.org.
- Robert Hankache, Kingsley Nketia Acheampong, Liang Song, Marek Brynda, Raad Khraishi, and Greig A Cowan. 2025. Evaluating the sensitivity of llms to prior context. *arXiv preprint arXiv:2506.00069*.
- Kadhim Hayawi and Sakib Shahriar. 2025. *Dear llm, why are they complaining? on root cause analysis (rca) in customer service automation using large language models (llms)*. In *2025 IEEE 5th International Conference on Human-Machine Systems (ICHMS)*, pages 338–343.
- Hugging Face. 2025. Snowflake arctic embed v2 documentation. <https://huggingface.co/Snowflake/snowflake-arctic-embed-l-v2.0>.
- Mali Jin and Nikolaos Aletras. 2021. *Modeling the severity of complaints in social media*. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages

- 2264–2274, Online. Association for Computational Linguistics.
- Jinja. 2025. Jinja. <https://jinja.palletsprojects.com/en/stable/>.
- Satyadhar Joshi. 2025. A comprehensive review of gen ai agents: Applications and frameworks in finance, investments and risk domains. *International Journal of Innovative Science and Research Technology*, pages 1339–1355.
- Guneet Kohli. 2026. Nine judges, two effective votes: Correlated errors undermine llm evaluation panels. *arXiv preprint arXiv:2605.29800*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. *Retrieval-augmented generation for knowledge-intensive nlp tasks*. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.
- Nian Li, Chen Gao, Mingyu Li, Yong Li, and Qingmin Liao. 2024. *EconAgent: Large language model-empowered agents for simulating macroeconomic activities*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15523–15536, Bangkok, Thailand. Association for Computational Linguistics.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. *Lost in the middle: How language models use long contexts*. *Transactions of the Association for Computational Linguistics*, 12:157–173.
- Noah MacCallum and Julian Lee. 2025. Gpt-4.1 prompting guide. https://developers.openai.com/cookbook/examples/gpt4-1_prompting_guide/.
- Mary L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.
- Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schuetze. 2025. *NoLiMa: Long-context evaluation beyond literal matching*. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 44554–44570. PMLR.
- OpenAI. 2025a. Introducing gpt-4.1 in the api. <https://openai.com/index/gpt-4-1/>.
- OpenAI. 2025b. Openai agents sdk. <https://openai.github.io/openai-agents-python/>.
- Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. *Llm evaluators recognize and favor their own generations*. In *Advances in Neural Information Processing Systems*, volume 37, pages 68772–68802. Curran Associates, Inc.
- Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2024. *Gorilla: Large language model connected with massive apis*. In *Advances in Neural Information Processing Systems*, volume 37, pages 126544–126565. Curran Associates, Inc.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. 2024. *ToolLLM: Facilitating large language models to master 16000+ real-world APIs*. In *The Twelfth International Conference on Learning Representations*.
- Konstantinos I. Rounteliotis, Nikolaos D. Tselikas, and Dimitrios K. Nasiopoulos. 2025. *Think before you classify: The rise of reasoning large language models for consumer complaint detection and classification*. *Electronics*, 14(6).
- Scale AI. 2025. Multichallenge leaderboard. <https://labs.scale.com/leaderboard/multichallenge>.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. *Toolformer: Language models can teach themselves to use tools*. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551. Curran Associates, Inc.
- Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. 2025. *Judging the judges: A systematic study of position bias in LLM-as-a-judge*. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 292–314, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.
- Apoorva Singh, Raghav Jain, and Sriparna Saha. 2023. *Reimagining complaint analysis: Adopting Seq2Path for a generative text-to-text framework*. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251, Nusa Dua, Bali. Association for Computational Linguistics.
- Zefan Wang, Zichuan Liu, Yingying Zhang, Aoxiao Zhong, Jihong Wang, Fengbin Yin, Lunting Fan, Lingfei Wu, and Qingsong Wen. 2024. *Rcagent: Cloud root cause analysis by autonomous agents with tool-augmented large language models*. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM '24*, page 4966–4974, New York, NY, USA. Association for Computing Machinery.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023.

React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Yangyang Yu, Haohang Li, Zhi Chen, Yuechen Jiang, Yang Li, Jordan W. Suchow, Denghui Zhang, and Khaldoun Khashanah. 2025. **Finmem: A performance-enhanced llm trading agent with layered memory and character design.** *IEEE Transactions on Big Data*, 11(6):3443–3459.

Yangyang Yu, Zhiyuan Yao, Haohang Li, Zhiyang Deng, Yuechen Jiang, Yupeng Cao, Zhi Chen, Jordan W. Suchow, Zhenyu Cui, Rong Liu, Zhaozhao Xu, Denghui Zhang, Koduvayur Subbalakshmi, Guojun Xiong, Yueru He, Jimin Huang, Dong Li, and Qianqian Xie. 2024. **Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making.** In *Advances in Neural Information Processing Systems*, volume 37, pages 137010–137045. Curran Associates, Inc.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhonghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. 2023. **Judging llm-as-a-judge with mt-bench and chatbot arena.** In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

A Downstream System Impact

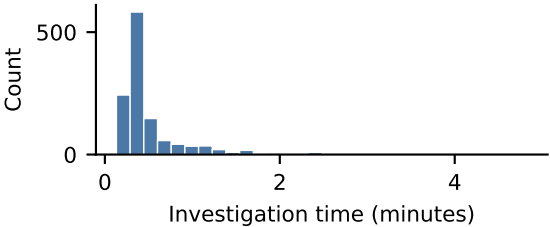
The investigation records produced by the agent are consumed by a downstream letter-generation process. To assess downstream impact, we compare LLM-as-a-Judge evaluations of letters generated from investigations written by human handlers and those produced by the investigation agent. We find minimal differences in letter quality between the two groups (see Table 5).

Table 5: Differences in downstream letter-writing quality for 319 summaries written by the agent and by human handlers.

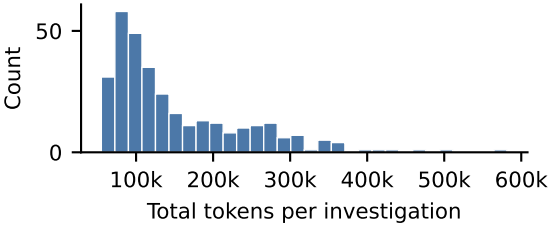
LLM-as-a-Judge metric	Δ Score (%)
Coherence	0
Compliance	0
Empathy	0
Fluency	0
Formatting	0
Grammar	0
Guidelines	0
Hallucination	0
Passive sentences	0
Principles	1
Reading age	0
Reading ease	0

B System Efficiency

In Figure 3a we show the distribution of investigation time per complaint, and in Figure 3b we show the distribution of token count usage per complaint.



(a) Distribution of investigation times per complaint (1,234 complaints).



(b) Distribution of total token usage per complaint (319 complaints). Median $\approx$ 120k.

Figure 3: Operational distributions from production deployment.

C Tool Pseudocode

Listing 1 shows pseudocode for a data tool. Each tool is a typed, decorated function that retrieves data, records the invocation in the trace, and serialises the result to XML.

D Output Schema

Listing 2 shows the Pydantic output schema that enforces structured decoding of the agent’s response.

E Evaluation Metric Definitions

Table 6 lists the four evaluation metrics and their definitions as presented to SME evaluators.

```

1 @function_tool
2 @error_handle
3 def get_call_transcripts(
4     ctx: RunContextWrapper[Trace],
5 ) -> str:
6     """Retrieve call transcripts.
7
8     XML tags follow the data dictionary:
9     - call_id: Unique call identifier
10    - datetime: Call timestamp
11    - transcript: Call transcript text
12    """
13    cid = ctx.context.customer_id
14    data = fetch_transcripts(cid)
15
16    ctx.context.record_tool_call(
17        system="call_transcripts",
18        rows_returned=len(data),
19    )
20
21    xml = serialise_to_xml(data, DICTIONARY)
22
23    if count_tokens(xml) > TOKEN_LIMIT:
24        xml = relevancy_filter(
25            xml, ctx.context.issue_detail
26        )
27
28    return xml

```

Listing 1: Pseudocode for a data tool.

Table 6: Evaluation metric definitions presented to SME evaluators.

Metric	Scale	Definition
Faithfulness	Binary	Is the information in the summary backed up by data from the systems?
Relevance	Binary	Is the information in the summary relevant to the customer complaint?
Completeness	Binary	Does the summary contain the information needed to conclude on the complaint?
Coherence	4-point	Is the text well-organised with a clear and coherent flow of ideas? (Unacceptable / Low / Good / Excellent)

```

1 class Fact(BaseModel):
2     """Atomic fact sourced from a system."""
3
4     description: str = Field(
5         description="One factual statement relevant to the "
6         "complaint, phrased in the past tense."
7     )
8     database: System = Field(
9         description="Name of the database where the "
10        "fact was found."
11    )
12    reference: str = Field(
13        description="Reference ID for the fact."
14    )
15    date: str = Field(
16        default="No date found",
17        description="Date recorded (dd-mm-YYYY)."
18    )
19
20
21 class ComplaintInvestigationSummary(BaseModel):
22     """Output of the complaint investigation."""
23
24     complaint_id: str = Field(
25         description="Unique complaint identifier."
26     )
27     summary: str = Field(
28         description="Summary of the facts."
29     )
30     facts: list[Fact] = Field(
31         description="Chronological list (oldest to "
32         "newest) of atomic events."
33     )

```

Listing 2: Pydantic output schema (simplified).