

GLS Chatbot: A Governed Multi-Agent Conversational System for Domain-Specific Customer Service

Bernat Coma-Puig, Guillermo Andrés Galván Hernández, Victor Mas Montserrat, Edgar Sarria, David Niñoles, Sonia Alonso

CX AI Hub, GLS Spain

{Bernat.Coma,GuillermoAndres.Galvan,Victor.Mas,
Edgar.Sarria,David.Ninoles,Sonia.Alonso}@gls-spain.com

Abstract

Hi, I'm Gisela, GLS's assistant. How can I help you today? This is how the Spanish deployment of the GLS Chatbot greets its customers. Behind the greeting is a governed multi-agent system for parcel-logistics customer service, built on self-hosted open-source small language models with RAG. An orchestrator separates routing, execution, and generation; probabilistic components are bracketed by deterministic safeguards, customer-impacting actions require explicit user confirmation, and out-of-scope, abusive, or manipulative inputs are refused. In production since November 2025, the system has handled over 1.7 million conversations across Spain and Germany. In Spain, the average realised cost during May 2026 was approximately 0.07€ per conversation, with sub-3-second latency. We describe the architecture, evaluate the Spanish deployment on task completion, routing conformance, and RAG answer quality, and report lessons from production.

1 Introduction

Logistics customer service increasingly relies on conversational automation to handle high parcel volumes and expectations of immediate support. Many deployed solutions still follow rigid dialog trees that limit flexibility and trigger conversational breakdowns that degrade user trust (Følstad et al., 2024). At the other extreme, reliance on proprietary LLM APIs reduces organizational control and auditability, particularly under strict European data-governance frameworks (European Union, 2024; European Parliament and Council of the European Union, 2016), and introduces usage-priced costs that scale with volume (Chen et al., 2023). Moreover, unconstrained generative agents in customer service have produced concrete failures that affect the reputation of the company (e.g., British Columbia Civil Resolution Tribunal (2024)).

This paper presents the **GLS Chatbot**, a cost-efficient, domain-adapted multi-agent system for

real customer-service operations (Fourney et al., 2024; Tran et al., 2025; Guo et al., 2024). It combines self-hosted open-source small language models (SLMs) with retrieval-augmented generation and a deterministic orchestration layer that separates responsibilities across specialized components (Belcak et al., 2025; Lewis et al., 2020). This modular design supports country-level localization (data sources, policies, capabilities) over a shared architectural core. In production since November 2025, the system has handled over 1.7 million conversations across Spain and Germany at approximately 0.07€ per conversation,¹ presented to Spanish users through the persona Gisela.

We analyse this deployment as a *governed* system for a regulated EU setting: we describe its architecture and governance mechanisms (pre-requisite enforcement, explicit confirmation for customer-impacting actions, refusal of out-of-scope, abusive, and manipulative inputs), present an evaluation of the Spanish deployment, and share lessons from production, including cost-optimisation choices and observed user behaviour.

2 System Overview

The system provides a 24/7 self-service channel for GLS customers, accessible through the company's website and WhatsApp, handling parcel-related requests via a natural-language chatbot interface. The assistant guides the interaction to collect missing information and presents structured outputs when this improves clarity and user decision-making.

2.1 Supported tasks

In the Spanish deployment, the chatbot supports (i) parcel status retrieval, (ii) allowed actions (post-tracking actions, e.g., Out of Home (OOH) redirection, neighbour/concierge delivery, delivery notes,

¹Average realized cost in Spain during May 2026, computed as total infrastructure spend divided by conversations handled; see Section 4.

and delivery date changes when eligible), (iii) escalation to human support under business-defined conditions, and (iv) company-related questions (e.g., “How can I contact GLS?”), answered via RAG.

2.2 Architecture Overview

The system is decomposed into five specialised agents² communicating through a typed JSON contract, which enables reproducible orchestration and lets each agent scale elastically on its observed demand. Decoupling routing and execution from generation is the organising principle: it keeps customer-impacting decisions out of a probabilistic model, lets each agent be adapted or swapped independently, and supports reuse across country deployments.³

Orchestrator (OR). This agent manages conversation state and routes messages between components by detected intent and task progress. It is deterministic control-flow logic backed by MongoDB, not a model, and is the only component with write access to persistent conversation state, which keeps behaviour auditable.

Triggerer (TR). Performs constrained tool selection: a typed JSON demand from a closed catalog with its parameters and any missing prerequisites.⁴ It is a Mistral 3 14B Instruct SLM⁵ (Liu et al., 2026) with a classification-style prompt, run on every turn under a sub-second budget. Because the output space is closed, errors are bounded: the router may pick the wrong demand but cannot produce open-ended output, and customer-impacting selections are validated downstream.

For deterministic requests, the Triggerer can emit a `static_response`, which the Orchestrator serves directly without invoking the Conversational Agent (see Section 2.5).

API Integrator (API). It executes validated, non-generative calls against GLS’s logistics systems (e.g., status retrieval, delivery-date change).

²Three are *generative*, backed by an SLM (the Triggerer, RAG, and Conversational agents); two are deterministic (the Orchestrator and API Integrator). We retain the term *agent* for uniformity.

³A PhoneBot is under development, extending the pipeline with STT/TTS agents for spoken interaction.

⁴Designed in October 2024, prior to the current consolidation of agentic tooling, it fulfils the role of a tool call in contemporary frameworks.

⁵The underlying model checkpoints have evolved since the initial deployment. The system was initially deployed with Mistral Small 3.2 24B Instruct; the model assignments reported here reflect the configuration at the time of writing.

RAG Component (RAG). This agent answers company-knowledge questions from indexed documentation, providing grounded context for user-facing responses. Its retrieval and generation pipeline is described in Section 2.3.

Conversational (CA). It produces the final user-facing reply from the Orchestrator’s structured input (API results, RAG content, or clarification requests). Like the Triggerer it is a self-hosted Mistral 3 14B Instruct SLM, prompted with a brand-aligned persona (Gisela in Spain) and the user’s language. It is the only generative agent whose responses reach the user directly.

2.3 RAG Design

The RAG component handles company-knowledge queries using an offline question-based indexing stage followed by online retrieval and grounded answer generation.

Question-based indexing. Rather than directly indexing source-document passages as the retrieval representation, we use an LLM as an offline preprocessing step to generate representative questions that each document can answer. Each generated question retains a reference to its source document and is indexed for retrieval. The number of questions scales with the length and information density of the document. This representation is motivated by the mismatch between the language used in internal documentation and the question-like formulations used by customers: retrieval operates over representations closer to likely user information needs, while the original source documents remain the authoritative evidence for answer generation.

This design is related to generated-query document expansion. Doc2query generates queries that a document is likely to answer and appends them to the document before indexing (Nogueira et al., 2019b), while docT5query follows the same general document-expansion principle with a stronger query-generation model (Nogueira et al., 2019a). Our implementation follows the same document-side intuition, but uses the generated questions themselves as retrieval proxies linked to the original source documents rather than appending them to the indexed document text.

Retrieval and generation. At inference time, the user query is matched against the generated-question index using hybrid retrieval implemented with LlamaIndex (Liu, 2022), combining BGE-M3 dense retrieval and BM25 lexical retrieval (Chen

et al., 2024; Robertson and Zaragoza, 2009). The highest-ranked questions identify candidate source documents. For each candidate document, Gemma 4 12B-it (Gemma Team et al., 2026) extracts the aspects relevant to the user query. The resulting evidence is then aggregated across candidates and passed to the same model to synthesise the final grounded answer. The generated questions therefore determine which source documents are considered relevant, while the final response is grounded in information extracted from the original documentation. End-to-end response quality is evaluated in Section 3.3.

Knowledge-base maintenance. Question generation is performed outside the online serving path and therefore does not contribute to customer-facing inference latency. When a source document is added or modified, the preprocessing step is re-run only for that document and its corresponding index entries are replaced. This allows the knowledge base to be updated incrementally without rebuilding the complete corpus.

2.4 Execution Flow and Safeguards

Governance is enforced by active guidance of the user toward supported tasks, and lightweight checks distributed along the pipeline (Table 1). Every probabilistic judge is bracketed by deterministic ones, and the degree of control scales with risk, with whether the content will be persisted or executed, and with whether it reaches someone other than the user.

Guidance Within Scope. The chatbot supports a predefined task set (Section 2.1) and guides users toward valid outcomes by requesting missing information when needed.

Safeguard Classes. *Deterministic checks* (e.g., regex prefilters, schema validation, and eligibility gates) are model-free and fail closed. *SLM judges* are used only where semantic interpretation is needed, typically emitting a short verdict. Customer-impacting actions never depend on an SLM verdict alone: semantic judges remain bounded by deterministic gates.

Manipulation vs. Abuse. Manipulation (injection, jailbreaks, instruction-extraction) is refused at input by a deterministic prefilter and an SLM security judge, terminating the interaction; frustration-driven profanity is deliberately *not* treated as an attack. Offensive content in *executed* parameters is handled differently: a delivery note is persisted and

reaches a courier or identifies a third party, so a semantic judge rejects insults or threats and requests a rephrasing rather than blocking the user. The judge fails open and the loop is unbounded, so the claim is not that no offensive note ever passes, but that such content is intercepted and re-requested before being persisted.

Action Validation and Confirmation. Customer-impacting actions pass mostly deterministic eligibility gates; the SLM is invoked only for what rules cannot resolve, such as whether a neighbour identification names both a person and a location. No such action executes in the turn it is requested: the system proposes it, requests explicit confirmation, and re-validates eligibility at execution, so a change in parcel state between the two phases cannot let an already-invalid action through. Every action with real-world effects is thus mediated by an informed user confirmation.

Escalation and Cross-Agent Redundancy. Escalation illustrates the double-bracketing: eligibility is first resolved deterministically against a whitelist of permitted parcel statuses, then an SLM judge assesses whether intent justifies escalation; both must agree. The guarantee is re-applied in the Conversational Agent, which receives an injected constraint when escalation is blocked and enforces it in its own output judge. This cross-agent redundancy is intentional.

Repair and Idempotency. Several checks correct rather than only block: the parcel-status judge normalises identifiers and postal codes from variable formatting and recovers values referenced in earlier turns (likewise for neighbour notes), guiding the interaction toward an actionable request. Completed actions are marked in dialogue state (e.g., created, derived) so a restated request is not re-executed. The Conversational Agent further uses constrained decoding to suppress control tokens during generation, complementing post-hoc format validation.

Example Flow. A status inquiry exercises the static fast path and triggers only a subset of safeguards: (i) the user asks “Where is my parcel?”; (ii) the Triggerer clears the prefilter and guard judge and, with identifiers missing, emits a `static_response`, so the Orchestrator serves a predefined reply without invoking the CA; (iii) the user sends “1301963361 / 28022”; (iv) the Triggerer emits `parcel_status`, the parcel-status judge validates the id and postal code, and the Or-

Stage	Mechanism	Agent	Role	Failure behaviour
Input	Injection filter ^D	TR	Matches known prompt-injection and jailbreak patterns	Refuse request
	Manipulation judge ^S	TR	Detects attempts to override instructions or extract hidden context	Refuse request
	Parcel-identity judge ^{S‡}	TR	Validates parcel identifiers and postal codes; may recover values from dialogue history	Repair parameters or ask again
	Free-text argument judge ^S	TR	Checks whether free-text arguments carry enough information to execute	Request missing information
	Note-content judge ^S	TR	Screens persisted delivery notes for abusive or threatening content	Request rephrasing
	Action eligibility check ^D	OR	Verifies action availability, identity consistency, and business constraints	Do not execute action
	Escalation eligibility check ^D	OR	Applies status-based escalation rules	Do not escalate
	Escalation-intent judge ^S	TR	Assesses whether escalation is justified under policy	Do not escalate
Output	Language judge ^S	CA	Checks consistency with the user’s language	Regenerate response
	Contact-information check ^D	CA	Prevents disclosure of unauthorised contact details	Suppress response
	Response-policy judge ^S	CA	Enforces scope, privacy, and escalation policies	Fallback response
	Control-token check ^D	CA	Prevents exposure of reserved control tokens	Fallback response

Table 1: Safeguards applied throughout the execution flow. Deterministic mechanisms (^D) rely on rules and business constraints; SLM-based judges (^S) are used only when semantic interpretation is required. The parcel-identity judge ([‡]) additionally supports parameter recovery and normalisation. Customer-impacting actions execute only after passing the corresponding deterministic controls.

Day	Avg Conv.	Avg Users	Avg Esc.	Esc. Rate	Turns/Conv	Avg Dur.
Mon	7,811	6,133	479	6.44%	5.61	3.4m
Tue	7,824	6,245	529	7.00%	5.76	3.5m
Wed	7,747	6,180	526	7.03%	5.81	3.5m
Thu	6,969	5,519	430	6.38%	5.68	3.5m
Fri	7,117	5,537	436	6.41%	5.77	3.5m
Sat	2,498	2,138	151	5.69%	5.41	3.3m
Sun	1,200	1,067	50	3.87%	4.74	2.9m

Table 2: Daily-average conversation metrics for the Spanish deployment, Nov 20 2025 – May 31 2026 (averages per day of week across the period).

chestrator dispatches to the API; (v) the API returns the parcel state, history, and allowed actions; (vi) the CA renders the reply, which passes the language, contact-info, and core-rules checks before display.

Customer-facing operations such as Parcel Shop redirection add validation: they execute only with a valid parcel context and the authorising allowedActions flag, free-text arguments are checked separately, and the system requests explicit confirmation before dispatch.

2.5 Operational Deployment

Deployed since November 20th 2025, the system handled 1.1M conversations in Spain (Table 2) and 669.4k in Germany up to 31 May 2026. To meet GDPR requirements, all SLMs are self-hosted in Europe⁶, keeping data and inference within EU infrastructure.

Deployment Setup. Each country has its own Azure resource group. The Orchestrator runs as a containerised service on Azure Container Apps; the SLM-backed agents are served through Azure ML managed endpoints, each backed by vLLM (Kwon et al., 2023) on A100 instances. Conversation state is persisted in MongoDB Atlas. The deployment scales elastically and independently at both the orchestration and model-endpoint levels based on per-agent load.

Latency and Static Fast Path. Per-agent latencies differ substantially: the Triggerer responds in < 1 s, the Conversational Agent in < 2.5 s, while the RAG path is slower because it combines hybrid retrieval, per-document aspect extraction, and generation with Gemma 4 12B-it. For deterministic requests, the Triggerer can instead emit

⁶Azure’s Germany West Central region.

a `static_response`, allowing the Orchestrator to return a predefined response without invoking the Conversational Agent. In a production sample,⁷ this fast path served 261,464 interactions, accounting for 34.8% of all chatbot turns. Most static interactions (61.0%) were clarification requests for missing information: a postal code (29.8%), both a parcel identifier and postal code (27.5%), or a parcel identifier alone (3.7%). A further 36.1% were deterministically formatted parcel-status responses and 2.9% served a predefined Parcel Shop listing. Typical end-to-end response time is under 3 s, with the static path reducing both latency and model usage.

Cost Analysis. The realized cost per conversation depends on the measurement period, since infrastructure spend is dominated by fixed A100 provisioning while traffic varies over time. In May 2026, total Azure spend divided by conversations handled by the Spanish deployment yielded an average cost of 0.07€ per conversation. This figure should therefore be interpreted as a period-specific average rather than a fixed per-query cost. At shorter time scales, the unit cost varies inversely with traffic, being lower during high-volume periods and higher during off-peak periods. Section 4 discusses subsequent cost and inference optimisations.

3 System Evaluation

All quantitative evaluations in this section concern the Spanish deployment. We evaluate the system along three complementary axes: an objective audit of routing conformance over all Triggerer calls in the audit window (Section 3.1), a quality assessment of conversation outcomes on a sampled set, using an LLM judge validated against a human annotator (Section 3.2), and an independent analysis of RAG answer quality (Section 3.3). The first measures conformance to predefined routing and eligibility rules, the second assesses task-level outcomes, and the third evaluates the knowledge component independently. For complementary qualitative inspection, Appendix A provides access to the live chatbot together with mock shipment records that allow its main interaction flows to be tested without real customer data.

3.1 Routing Conformance Audit

Between May 9th and June 8th 2026⁸ the Triggerer processed 752818 calls. The router placed 99.99% of decisions within the demand catalog, with none null or unparseable; the remaining 101 cases (0.013%) contained an invalid demand value, typically a control reason emitted in the demand field; all were rejected by downstream deterministic validation before execution.

Checking every customer-impacting decision against the parcel’s `allowedActions`, the router’s action demands were consistent with the allowed actions except for 9 parcel-shop lookups (0.04%) emitted without parcel context, which the deterministic gate blocks; all 4352 identifier-mode redirections referenced the correct parcel. Of 13590 escalation decisions, none targeted an ineligible status; 4778 (35%) concerned delivered parcels, which the rules route to conditional semantic review rather than certify automatically.

These results measure conformance to the predefined routing and eligibility rules; they do not by themselves measure conversation-level outcome quality.

3.2 Conversation Outcome Evaluation

We classify each conversation’s outcome into one of six categories (Table 3) and validate the LLM judge against a human annotator on 100 sampled conversations (Table 4): they agree on 89 cases, with Cohen’s $\kappa = 0.786$ (substantial; the gap from raw agreement reflects the strong TA-dominated class imbalance). The categories form three tiers: successful outcomes (TA, TNA); undesired interactions (OB, WH); and content failures (IC, MO). At the tier level the judge’s distribution closely matches the human’s, 84 vs 87 successful, 14 vs 12 undesired interactions, and 2 vs 1 content failures, providing a similar coarse-grained outcome distribution despite disagreements on individual labels.

Per-class recall is 98.5% for TA, 78.9% for TNA, 71.4% for OB, 57.1% for WH, and 50.0% for IC. MO recall cannot be estimated because no human-labelled MO case occurs in the validation sample.

Two systematic judge errors affect the most safety-relevant categories. First, hallucination detection (IC) is weakest: the judge caught one of two cases. Both shared the same cause: the Con-

⁷Collected from July 25th to July 29th.

⁸To guarantee Data Protection requirements, we do not keep conversations from our customers for more than 30 days.

Label	Outcome
TA	Requested action completed (e.g., status delivered, date changed).
TNA	User attempted an action the rules or parcel status correctly disallowed.
OB	System offered an action, then correctly retracted it as not possible.
WH	System’s own error blocked an achievable task (misroute, wrong ask, missed prerequisite, lack of understanding).
IC	Wrong or unsupported information (invented status, fabricated policy).
MO	Technical failure (error, empty response, malformed output).

Table 3: Outcome taxonomy for conversation-level evaluation, applied by both the human annotator and the LLM judge.

versational Agent confirmed a successful escalation (“your case has been escalated”) although it had not executed, misreporting a failed upstream action rather than fabricating freely. Second, the judge labels as malformed output (MO) predefined fallback messages that fired as designed: every predicted MO case was a deliberate safe fallback to an out-of-scope or unparseable input, not a technical failure. Thus, IC/MO verdicts are the least reliable, and we treat content-failure detection as a known limitation (Section 5). Remaining disagreements lie mainly on the TNA/WH boundary, where distinguishing a correct block from a system error requires judging whether the action was actually possible. Some trace to informal WhatsApp input (typos, fragments, abbreviations), which can strain understanding and appear as WH errors. This degrades comprehension rather than control: deterministic gates typically turn misunderstandings into clarification requests rather than incorrect executions.

Applied to 500 sampled conversations from the Spanish deployment (Table 5), the judge classified 87.8% as successful outcomes, 10.8% as undesired interactions, and 1.4% as content failures. The low-count IC and MO estimates should be interpreted cautiously.⁹

3.3 RAG Evaluation

We evaluate the knowledge component on 66 held-out questions over the 54-document corpus, disjoint from the 1100 generated questions in the retrieval

⁹Wilson 95% confidence intervals: IC, 0.31–2.04%; MO, 0.20–1.75%.

		LLM Judge (predicted)						
		TA	TNA	OB	WH	IC	MO	Total
Human (truth)	TA	64	0	0	1	0	0	65
	TNA	2	15	0	2	0	0	19
	OB	0	2	5	0	0	0	7
	WH	1	2	0	4	0	0	7
	IC	1	0	0	0	1	0	2
	MO	0	0	0	0	0	0	0
Total		68	19	5	7	1	0	100

Table 4: Agreement between the human annotator (rows, treated as ground truth) and the LLM judge (columns) over 100 sampled conversations. Shaded diagonal cells are agreements. Overall agreement 89% (Cohen’s $\kappa = 0.786$).

Outcome	TA	TNA	OB	WH	IC	MO	Total
Count	342	97	18	36	4	3	500
%	68.4	19.4	3.6	7.2	0.8	0.6	100

Table 5: LLM-judge outcome distribution over 500 evaluated conversations in Spain.

index. The questions were manually constructed with reference to the corpus and selected for coverage: their gold-document sets span 51 of 54 documents, with the remaining three considered out of scope because they describe shipment-state semantics handled elsewhere. The bank is also diverse: mean pairwise TF-IDF cosine similarity is 0.019, and only 2 of the 2,145 pairs exceed 0.5, both intentional minimal pairs differing in whether the corpus contains the answer.

Each question uses a *rubric* defining required facts, acceptable alternatives, and forbidden claims rather than a reference answer. Ten questions are missing, for which the expected behaviour is abstention. Final verdicts were assigned by a human expert. An LLM independently reviewed the 617 rubric items as a consistency check, to help identify information the human evaluation might have overlooked, but did not determine the final label. Verdicts are pass, explanation (correct but incomplete), documents (gold documents retrieved but final answer incorrect), and fail (expected documents not retrieved).

Across the 56 answerable questions, the expected documents were always retrieved and no answer contradicted the retrieved evidence; errors were limited to omissions. For the 10 missing

Question type	<i>n</i>	pass	expl.	docs	fail
factual	44	41	3	0	0
pointer	10	9	1	0	0
either	2	2	0	0	0
missing	10	9	0	0	1
Total	66	61	4	0	1

Table 6: RAG evaluation results by question type. For all 56 answerable questions, the source documents were successfully retrieved; the reported verdicts therefore reflect the quality of the final generated answer. For missing questions, pass indicates appropriate abstention, while fail indicates that the system provided an unsupported answer.

questions, the system abstained on 9 but answered one question about GLS’s worldwide workforce using the figure for GLS Spain, the only unsupported answer observed.

4 Discussion and Lessons Learned

Static Fast Path. Handling deterministic turns without open-ended generation reduces latency and model usage while limiting exposure to generative errors, making such traffic a high-impact production optimisation.

Complexity for Flexibility. Decomposition adds complexity but isolates responsibilities behind typed contracts, allowing components to scale or change independently while making localisation and governance more tractable.

The Cost of Staying in Scope. Keeping generative agents within scope remained challenging: instruction-tuned SLMs sometimes propose unsupported remedies beyond the system’s task definition. Layered controls, including explicit prohibitions, output checks, and judges, reduce this behaviour at the cost of complexity and occasional over-cautious refusals. Across the deployment, we are aware of no reported confidentiality or content-safety incidents, though this should not be interpreted as an estimated incident rate. Scope adherence therefore remains an ongoing engineering requirement rather than a one-time prompting problem.

Cost and Inference Optimisation. The reported 0.07€ per-conversation cost reflects capacity provisioned for peak demand rather than an optimised steady-state configuration. Preliminary tests during the first weeks of August, using Gemma 4 12B-

it across all SLM-backed agents with multi-token prediction, reduced inference time and average cost to 0.06€ per conversation. This was observed during a low-demand period, when fixed provisioning would normally increase per-conversation cost. The improvement comes from higher per-GPU throughput, allowing each instance to serve more requests and reducing scaling requirements. Prompt compression, smaller-model routing, and caching provide further optimisation opportunities.

The WhatsApp Channel. Informal WhatsApp input contributes to WH errors (Section 3.2); parcel-identity repair and deterministic gates typically turn misunderstandings into clarification requests rather than wrong actions.

5 Limitations

Evaluation scope. Our evaluation is based on a sample of production conversations judged by both a human annotator and an LLM judge, together with component-level metrics. It does not constitute a controlled comparison against alternative architectures (e.g., a single-model baseline or a proprietary-API system), and we therefore cannot make causal claims that the multi-agent design outperforms such alternatives on quality; our contribution is a deployment account rather than a head-to-head benchmark.

Reliance on LLM-as-a-judge. Part of our evaluation uses an LLM as an automated judge. LLM judges are known to exhibit biases and to correlate imperfectly with human judgement, particularly on borderline cases such as the distinction between a correct refusal and a misunderstanding. We mitigate this with a human-judged subset, but the larger-scale results inherit the judge’s limitations.

Proprietary data and single domain. The system operates on proprietary logistics data in a single domain (parcel customer service) for a single organisation. While the architectural principles are intended to generalise, the specific results (intent distribution, escalation rates, cost figures) reflect this setting and may not transfer to other domains or organisations.

Cost figures are deployment-specific. The reported per-conversation cost reflects our particular provisioning choices (self-hosted models on A100 instances, capacity sized for peak season) and the cloud region and pricing available to us. These figures are expected to represent an upper bound

for our configuration and are not directly comparable to other deployments with different hardware, traffic patterns, or pricing agreements.

Language and locale coverage. The deployment covers two country markets and a limited set of languages. Behaviour in lower-resource languages, or for users mixing languages within a turn, is not systematically evaluated here.

Safety evaluation breadth. Our safety assessment focuses on the categories most frequent in production (out-of-scope, abusive, and manipulative inputs, including prompt-injection attempts). It is not an adversarial red-teaming study, and we do not claim robustness against a determined or novel attacker.

6 Ethical Considerations

Data protection and user privacy. The system is designed around data minimisation. Most parcel-related interactions require only a parcel identifier and postal code. Some actions require additional personal data, such as an email address for escalation or recipient information for alternative delivery. To meet EU data-protection requirements (GDPR), all data and model inference are kept within EU infrastructure using self-hosted models. The deployment was developed in coordination with the organisation’s Data Protection team, and conversation data is processed accordingly: conversations are deleted from our databases after 30 days.

Human oversight and escalation. The system is designed to keep customer-impacting actions under explicit user confirmation and to escalate to human agents under business-defined conditions rather than attempting to resolve every case autonomously. This preserves a human fallback for situations the automated system is not equipped to handle.

Transparency to users. Users are informed that they are interacting with an automated assistant at the start of the conversation.

Safety and misuse. The system incorporates input screening for abusive and manipulative content and refuses out-of-scope requests, reducing the risk of generating harmful, misleading, or off-brand responses. Because customer-facing actions are executed by a deliberately non-generative integration layer only after validation and confirmation, the

risk of a language model triggering an unintended real-world action is structurally constrained.

Risk of automated errors. As with any deployed NLP system, incorrect routing or generation can affect users (e.g., a failed or delayed resolution). The governance mechanisms and human-escalation path are intended to bound the impact of such errors, and the deterministic orchestration layer provides an auditable record supporting accountability and review.

Acknowledgments

We thank the Spanish PMO team and our colleagues across GLS, including Customer Experience, IT, Data Protection, Customer Service, and Security, for their support in developing, deploying, and operating the chatbot. We are also grateful to those colleagues across GLS who trusted us and supported the project throughout its development. We also thank Marta Arias from the Universitat Politècnica de Catalunya for her insightful feedback on the manuscript.

References

- Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. 2025. Small language models are the future of agentic ai. *arXiv preprint arXiv:2506.02153*.
- British Columbia Civil Resolution Tribunal. 2024. Mofatt v. Air Canada, 2024 bcrt 149. <https://canlii.ca/t/k2spq>. CanLII. Decision of 14 February 2024 (Member C. C. Rivers).
- Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 4(5).
- Lingjiao Chen, Matei Zaharia, and James Zou. 2023. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*.
- European Parliament and Council of the European Union. 2016. [Regulation \(EU\) 2016/679 of the European Parliament and of the Council on the protection of natural persons with regard to the processing of personal data](#). Official Journal of the European Union, L 119, 1–88.
- European Union. 2024. [Regulation \(EU\) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence \(artificial intelligence act\)](#). OJ L 2024/1689, 12.7.2024.

Asbjørn Følstad, Effie L-C Law, and Nena van As. 2024. Conversational breakdown in a customer service chatbot: Impact of task order and criticality on user trust and emotion. *ACM Transactions on Computer-Human Interaction*, 31(5):1–52.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, and 1 others. 2024. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, and 1 others. 2026. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xi-angliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, and 1 others. 2026. Ministral 3. *arXiv preprint arXiv:2601.08584*.

Jerry Liu. 2022. [LlamaIndex](#).

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to docttttquery. *Online preprint*, 6(2):1–3.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O’Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.

A Setup and Availability

A set of preconfigured mock shipment records is provided (see Table 7) to enable testing of the chatbot’s full functionality without relying on real customer data. The chatbot is accessible on the company’s official website¹⁰. Although this paper exemplifies a subset of allowed actions, the system supports all of them. For actions not covered in this document, the chatbot itself guides the user through the required steps and validations.

B UX Scenarios and Interaction Flows

We demonstrate the system through a set of representative customer service scenarios. Due to the probabilistic nature of language generation, the system produces different but semantically consistent responses across interactions.

B.1 Parcel Status Inquiry

Goal: To provide the current shipment status. This is required for most of the interactions between the chatbot and the user.

Required identifiers: (i) parcel ID and (ii) postal code.

System Outcome: Formatted shipment status information (see Fig. 1).

Safeguards and validation: Parcel identification is mandatory for any parcel-related inquiry: the assistant requires a valid parcel ID and matching postal code before retrieving status. If information is incomplete or unclear, it explains what is needed (and where to find it) and blocks the request until the required identifiers are provided, without guessing.

B.2 Allowed Actions (Parcel Shop Delivery)

Goal: Execute an allowed post-tracking action. We exemplify this by redirecting the parcel to a Parcel Shop.

Pre-requisite: Successful parcel identification (parcel ID + postal code).

Information required: Selection of the Parcel Shop.

System Outcome: The assistant proposes eligible Parcel Shops and completes the redirection only after explicit user confirmation (see Fig. 2).

Additional actions supported: The system applies the same prerequisite-and-confirmation pattern to other post-tracking actions, including neighbour/concierge derivation, delivery notes, and de-

¹⁰<https://gls-group.com/ES/es/home/>

Parcel ID	Status	Out of Home	Neighbour	Date change	Add note	Escalation
1139815480	On delivery	✓	✗	✗	✗	✗
1139818330	Incident	✓	✓	✓	✓	✗
1139817212	Delivered	✗	✗	✗	✗	✓

Table 7: Mock parcel records for testing the chatbot; allowed actions are those of Section 2.1. The postal code is 28022 for all records.

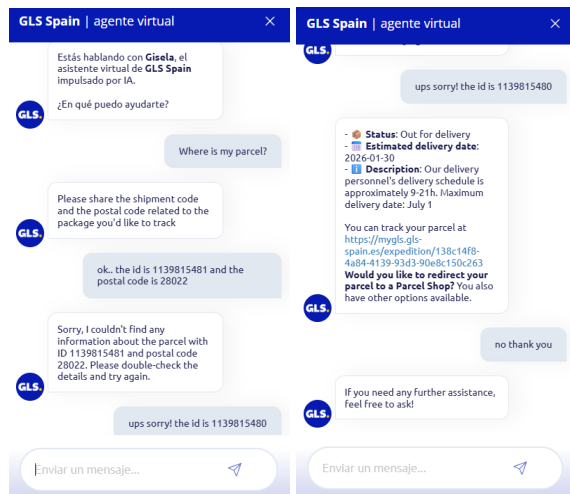


Figure 1: Parcel Status Inquiry on GLS Chatbot.

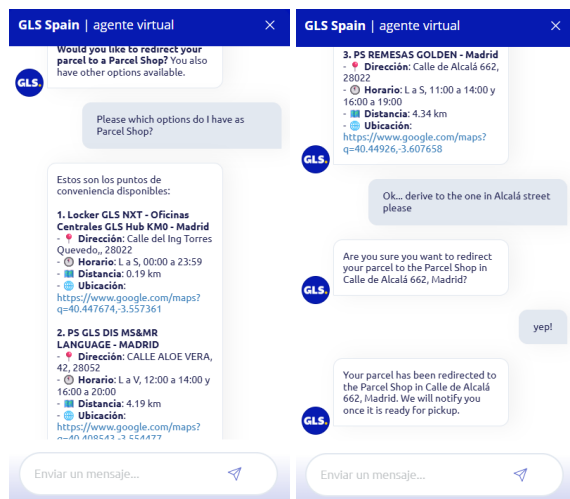


Figure 2: Derivation to a Parcel Shop successfully triggered by the user.

livery date changes.

Safeguards and validation: The system enforces action-specific constraints: incomplete parameters (e.g., missing Parcel Shop selection) block execution until resolved; abusive content in notes or derivations is rejected with a request for compliant rephrasing; and neighbour derivations require

sufficient identification (name and address) before proceeding.

B.3 Change Delivery Date

Goal: Execute a delivery date change for a parcel.

Pre-requisite: Successful parcel identification (parcel ID + postal code).

Information required: New delivery date selected by the user.

System Outcome: The assistant presents available dates, confirms the user's selection, and completes the date change upon explicit confirmation.

Safeguards and validation: The system enforces date-specific constraints: date changes are blocked when not available for the given parcel; selections outside the permitted date range are rejected with a prompt to choose a valid alternative. The system also resolves natural language date expressions (e.g., "tomorrow") into concrete dates before validation.

B.4 Escalation Process

Goal: Escalate a customer case to human support when automated resolution is not appropriate.

Pre-requisite: A parcel/customer case eligible for escalation under business rules.¹¹

Information Required: User's email.

System Outcome: The assistant collects the necessary context and creates a support escalation request only when eligibility conditions are met.

Safeguards and validation: The system prevents unnecessary escalations: ineligible cases (e.g., status "in delivery") are refused with an explanation; missing identifiers or context block the request until provided; and duplicate escalations are detected, informing the user that a ticket already exists.

¹¹ Escalation rules are defined internally and cannot be disclosed. They are designed to prevent unnecessary escalations when the issue can be resolved through automated actions.