Tech4Biz

Excellence Guaranted!

Tech4Biz
Excellence Guaranteed!

CASE STUDY 08

SOCIAL SENTIMENT & FEEDBACK INTELLIGENCE

Grounded Classification, Priority Scoring and Human Approved Response

A language layer that reads every piece of public customer feedback as it arrives, resolves it against the real customer, product and case records, and opens a scored ticket only when the evidence supports one. The model reads and proposes. Ticketing, customer contact and publication stay exactly where they are.

The Business Problem

How customer feedback reaches the business today

Customer feedback arrives on social platforms, review sites, app stores, marketplace listings and feedback forms. Most organisations watch those channels with a mix of manual reading, keyword alerts and a social listening dashboard. Each of those works for the volume that was anticipated, and each of them breaks when volume grows or when one issue spreads faster than the review cycle.

The practical consequence is that the business learns about a problem after the customer has already told everybody else about it. A team that wants to know which complaints are serious, which ones are the same complaint, and which customer is about to leave is reading text and forming a judgement by hand.

Where the cost accumulates

- **Detection delay.** A serious complaint sits unread because nobody was rostered on that channel that evening.
- **Fragmented view.** One defect appears on three platforms and gets counted as three unrelated comments.
- **Volume blindness.** The complaints that matter hide inside a large volume of low value chatter.
- **Manual triage.** A person reads, judges and retypes the issue into a ticket, and the judgement changes with the person.
- **Inconsistent voice.** Replies are drafted individually, so tone and commitment vary across the brand.
- **No operational signal.** Management sees counts of mentions rather than causes of complaint.

What the customer actually writes

CUSTOMER, PUBLIC REVIEW

"Third time I have contacted support and nobody has fixed this. Extremely disappointed."

That single sentence carries a sentiment, a repeat contact history, an unresolved support failure, a churn risk and an implied escalation. A listening dashboard reports it as one negative mention. The business needs it as a case, with an owner and a clock on it.

THE REQUIREMENT IN ONE SENTENCE

Read every piece of public feedback as it arrives, work out what happened and how serious it is, resolve it against the real customer, product and case records, and open a ticket only when the evidence supports one.

Why grounding, specifically

A sentiment score on its own cannot be acted on, because it does not say what broke or who it belongs to. A model that does not hold your catalogue will invent a product name, and a model that guesses at customer identity will attach a complaint to the wrong account. Both failures are worse than no automation at all, because both of them look confident. Grounding every named entity against a live record is what turns a classifier into something an operations team can be accountable for.

Solution Overview

The platform sits between the external feedback channels and the systems the organisation already runs. It collects feedback, resolves what it refers to against live customer and product records, produces a structured case, scores it, and hands that case to the existing ticketing and response workflow.

Layered architecture

LAYER 6 INTAKE

Channel connectors and normalisation

Platform APIs, review feeds and webhooks, normalised into one feedback record.

LAYER 5 UNDERSTANDING

Language models with a fixed output schema

Sentiment, intent, category, entities, issue summary and severity signals.

LAYER 4 GROUNDING

Live business model

Customers, products, orders, campaigns and open cases, refreshed from CRM and ticketing.

LAYER 3 SCORING

Deterministic priority engine

Fixed weights, thresholds and rules that decide what becomes a case and how urgent it is.

LAYER 2 HANDOFF

Schema validated case record

JSON contract, versioned, auditable, rejected if it does not validate.

LAYER 1 ACTION

Ticketing, CRM and response workflow

Unchanged. Assignment, SLA, escalation and publication behave exactly as they do today.

The boundary that makes this deployable

The most important design decision in this architecture is what the model is not allowed to do.

The model does	The model never does
Read and interpret the feedback	Publish a reply on its own
Resolve names to real records	Decide the SLA or the owner
Extract the issue and the entities	Close, merge or reopen a case
Propose a severity and a reply	Contact a customer directly
Emit a structured case record	Write to the CRM
Flag when it is not confident	Guess which customer wrote it

WHY THIS MATTERS COMMERCIALLY

Publication stays with the people who own the brand voice, and the ticketing system stays the single source of truth for work in progress. Nothing in the existing escalation path, the existing approval chain or the existing customer contract has to be renegotiated to switch the language layer on. That is what makes it deployable on a live brand rather than in a pilot sandbox.

Channels and Ingestion

Channel coverage

Channel class	What is collected
Social platforms	Posts, comments and replies on brand owned handles, plus mentions where the platform interface permits it.
Review sites	New reviews, edits, ratings and merchant replies.
App stores	Reviews by application version, country and rating.
Marketplaces	Product reviews and seller feedback against listed SKUs.
Owned channels	Feedback forms, in product surveys and post resolution surveys.
Support transcripts	Chat and email records where an integration exists and consent allows it.

Normalisation

Every item becomes one feedback record before any model sees it. The record carries the source, the platform identifiers, the author handle, the timestamp, the detected language, the text, any media references and the parent thread. One shape for every channel is what allows a single taxonomy and a single set of weights to work across all of them.

Source and platform ID · Author handle · Detected language · Thread position · Media references · Ingestion timestamp

Collection rules that keep this defensible

Collection is limited to what the platform terms allow and to what the organisation has a lawful basis to hold. The pipeline stores the minimum it needs, keeps personal identifiers separate from the text used for analysis, and applies a retention period per channel rather than a single global one.

Public data only · Platform terms respected · Webhooks preferred over polling · Rate limited · Retention per channel · Identifier minimisation

> **DESIGN DECISION TO TAKE EARLY**
>
> Whether private messages and support transcripts are in scope changes the consent position, the retention policy and the review workflow. It also changes who is allowed to read the queue. Decide this before detailed design rather than during integration.

Where the platform runs

The reference deployment runs inside the customer's own cloud account or data centre, in the region where the data is required to stay. The only outbound paths are the channel connectors themselves and, where a hosted model is used, the inference endpoint. Where residency rules forbid that, the same pipeline runs against models hosted in region or on the customer's own hardware, and nothing else in the architecture changes.

The Business World Model

A language model that does not know your catalogue will invent a product, and a model that does not know your customers will attach a complaint to the wrong one. The world model is the component that prevents both, and it is the difference between a demonstration and a deployable system.

What it holds

Entity class	Attributes tracked
Customers	Identifier, known handles, value tier, contact history, open and recent cases
Products and SKUs	Names, aliases, model numbers, current version, known defects
Orders and shipments	Reference, status and delivery dates where an integration exists
Locations	Branch, store and service area identifiers
Campaigns	Active promotions and launch dates, so an expected spike is not read as a crisis
Cases	Open and recently closed tickets, so a repeat is recognised as a repeat

How it stays current

- Subscribed to CRM and ticketing events over the organisation's existing message broker.
- Catalogue synchronised on a schedule and on product events, so a launch is visible the day it happens.
- Staleness is tracked explicitly. A record older than a configured threshold is treated as unknown rather than as last known good.
- Aliases are first class. Customers write "the blue one" and the model number interchangeably, and the registry has to resolve both.

Grounding contract

Every entity the model names in a case must resolve to a real record. There is no path by which an unresolved name becomes a link in the ticketing system or the CRM.

Condition	Behaviour
Resolves uniquely	Proceed and link.
Resolves to several records	Record the candidates and route to a human for selection.
Does not resolve	Raise the case unlinked and say so. Never attach the nearest match.
Author is not a known customer	Treat as anonymous. No customer record is created or updated.
Record is stale	Treat as unknown and ask for confirmation before linking.

FAIL CLOSED, ALWAYS

A complaint linked to the wrong customer is far more expensive than a case that arrives unlinked. The unlinked case costs a reviewer thirty seconds. The wrong link contaminates the account history, triggers the wrong retention offer and is very hard to unwind once other systems have read it. The system is designed so that uncertainty produces a question, never a guess.

From Comment to Case

Constrained generation

The model does not write free text that is then parsed. It generates directly into a schema, so structurally invalid output is not merely rejected downstream but is impossible to produce.

01
Feedback record plus a filtered snapshot
The model sees only the customers, products and open cases that plausibly match, not the whole database.

02
Schema constrained decoding
Output is forced to conform to the case grammar at generation time, including the allowed category values.

03
Schema validation
Types, enumerations, required fields and value ranges are checked against the contract.

04
Semantic validation
The named product exists, the order belongs to the named customer, and the category fits the product line.

05
Human review
Every case above a configured significance threshold is read by a person before anything is published.

What the model is given

- The feedback text, its language and its channel.
- A filtered snapshot of candidate customers, products and open cases.
- The active taxonomy and the brand response policy.
- Nothing else. No open browsing, no unbounded account history, no free text memory.

Case record contract

```
{
  "case_id"        : "FB-2026-08-10-0471",
  "source"         : "review-site",
  "received_at"    : "2026-08-10T09:14:22Z",
  "language"       : "en",
  "sentiment"      : "highly_negative",
  "intensity"      : 0.91,
  "intent"         : "complaint",
  "category"       : "support",
  "subcategory"    : "unresolved_ticket",
  "entities": {
      "customer"   : "CUST-40218",
      "product"    : "SKU-7741",
      "order"      : null },
  "issue_summary"  : "Third contact on the same fault, still open.",
  "severity"       : "high",
  "priority"       : "critical",
  "signals"        : ["repeat_contact", "churn_language",
                      "public_visibility"],
  "cluster_id"     : "CL-2261",
  "suggested_reply": "tmpl-14",
  "requires_human" : true,
  "model_snapshot" : "bm-33107",
  "schema_version" : "1.4"
}
```

WHY THE SNAPSHOT REFERENCE IS IN THE RECORD

Recording which business model snapshot the case was built against makes the decision reproducible after the fact. When a response is questioned weeks later, the first question is always what the system believed at the time, and this answers it exactly.

The Priority Engine

Why scoring is separate from the model

Language models are good at understanding what someone meant. They are unreliable at applying a consistent weighting across thousands of items, and unreliable in a way that is hard to test. The priority engine is therefore a conventional deterministic component that takes the model's reading and turns it into a number the business can defend.

Engine responsibility	What it checks
Severity scoring	Sentiment intensity, escalation language and stated harm are combined with fixed weights.
Repeat detection	Does this author, product or defect already have an open or recently closed case.
Reach and visibility	Platform, audience size, whether the item is public and whether it is being shared.
Customer value	Contract tier, lifetime value and renewal proximity where the CRM supplies them.
Regulatory triggers	Safety, injury, fraud, discrimination and data protection language route on a separate path.
Clustering	Does this belong to an existing issue cluster rather than opening a new case.

Routing

Where the case does not name an owner, the engine proposes one. The routing policy is configuration rather than model behaviour, so a business can tune it without retraining anything.

- Queue selected by category, with product line ownership taking precedence over channel ownership.
- Region and language decide the team, so a reply is drafted by people who work in that market.
- Regulatory triggers bypass the standard queue and go straight to the compliance path.
- Load balancing across reviewers over a shift, so the queue does not pile onto one person.

> **ARBITRATION STAYS WITH THE TICKETING SYSTEM**
>
> The engine proposes. The existing ticketing system remains the single authority that assigns work, runs the SLA clock and resolves contention between teams. Introducing a second scheduler would be the fastest way to lose track of who owns a customer, so the architecture deliberately avoids it.

Suppression paths

Not everything becomes a case. Spam, bot activity, competitor noise, resolved duplicates and off topic mentions are recorded with a suppression reason rather than silently dropped. Suppression is auditable and reversible, because the item a reviewer wants to see six weeks later is usually the one an early rule decided was noise.

Model Estate and Cost Budget

Why a cascade rather than one large model

Volume is the cost driver in this workload. Most items are short, most of them are not complaints, and sending every one of them to a large model turns a fixed problem into a bill that grows with the brand. A cascade avoids that. A small classifier filters, a mid sized model does the structured extraction, and the largest model is reserved for the small share of items that are genuinely ambiguous or genuinely severe.

Stage	Model class	Share of volume
Language and spam filter	Small encoder	100 percent
Sentiment and intent	Fine tuned small model	100 percent
Embedding and clustering	Embedding model	100 percent
Structured extraction	Mid size instruction model	~20 to 30 percent
Ambiguous or severe review	Largest model	~2 to 5 percent
Response drafting	Mid size with policy in context	Cases only

Where the engineering effort actually goes

The published price per token is a ceiling that nobody pays in practice. Closing the gap between that ceiling and the real cost per case is where the work sits.

BATCHING

Items are processed in windows, not one at a time

Window length is tuned per channel against the detection latency target.

CACHING

Repeat text and known clusters skip the model entirely

Campaign replies and copied review text are a larger share of volume than most teams expect.

ROUTING

The cheapest stage that can decide is the stage that decides

Escalation to a larger model happens on confidence and severity, never by default.

CONTEXT DISCIPLINE

The snapshot is filtered before it reaches the model

Candidate entities are retrieved first, so context length stays flat as the catalogue grows.

STATUS OF FIGURES

Volume shares and stage allocations here are design targets for the reference configuration. They are confirmed against the customer's own channel mix, language spread and complaint rate during the engineering phase, because the cascade economics depend entirely on that mix.

Execution and Enterprise Integration

Integration surface

Component	Role
Ticketing system	Case of record, assignment, SLA clock and closure.
CRM	Customer identity, value tier and interaction history.
Product catalogue	SKU, version, alias and known defect references.
Message broker	Event distribution between ingestion, scoring and handoff.
Data warehouse	Trend analysis, cluster history and management reporting.
Social management tool	Publication of approved replies through the existing account and approval chain.
Identity provider	Single sign on and role based access for reviewers and approvers.
Notification	Email, chat and paging for critical and regulatory cases.

What stays under human control

Everything above the case record. Assignment, commitment to a customer, compensation, closure and publication operate exactly as they did before the language layer was introduced, and they are audited the same way.

Control architecture

The controls that carry brand and legal risk are not implemented in software a model can influence. They stay where they already are, and the language layer sits entirely outside that boundary.

Control	Where it lives
Publication approval	Human reviewer inside the existing social management tool.
Legal and regulatory escalation	Compliance queue, unchanged.
Direct customer contact	Support agent, through existing channels.
Changes to a customer record	Agent action only. The pipeline has no write path.
Case closure and SLA	Ticketing system workflow.
Classification and drafting	The only place the language layer participates.

PRIVACY AND REGULATORY CONTEXT

Feedback that identifies a person is personal data, and it is handled under the applicable data protection regime for each market in scope. Lawful basis, retention period, access control and the process for a deletion request are agreed in Phase 0 rather than retrofitted. Because the pipeline holds no write path into customer records, the surface that has to be assessed stays small.

Worked Example, End to End

CUSTOMER, PUBLIC REVIEW, 4.2K FOLLOWER ACCOUNT

"Third time I have contacted support about the same fault on my SKU-7741 and nobody has fixed it. Extremely disappointed, I am cancelling."

STEP 1 INGEST

Item collected and normalised

Review webhook fires, record created with source, handle, language and timestamp.

STEP 2 RESOLVE

Entities matched to the business model

Handle matches CUST-40218. SKU-7741 exists and is on the current version. Two support tickets in 21 days.

STEP 3 INTERPRET

Issue and signals extracted

Complaint, unresolved support, explicit cancellation language, public and visible.

STEP 4 SCORE

Severity and priority computed

Severity high, repeat contact multiplier applied, renewal in 40 days, priority critical.

STEP 5 REVIEW

Ticket created and reply drafted

Case opens in the existing queue with the draft attached. A person approves before anything is published.

SYSTEM, TO THE SUPPORT QUEUE

Critical case FB-2026-08-10-0471. Customer CUST-40218, renewal in 40 days, third contact on SKU-7741 in 21 days with tickets 88214 and 88907 still open. Public review, cancellation stated. Routed to the support escalation queue and to the product line owner. Draft reply attached and held for approval.

And when it cannot be grounded

CUSTOMER, PUBLIC POST

"Your new dryer burned out after two weeks and support is ignoring me."

SYSTEM, TO THE REVIEWER

No product matching "dryer" exists in the catalogue for this brand. The author handle does not match any customer record. Case raised as unlinked, severity high on the safety language, routed to product support for identification. No customer record was touched and no reply was drafted.

THIS IS THE BEHAVIOUR THAT MATTERS MOST

Two separate failures were caught in one item, both were reported specifically, and the case still reached a human because the safety language on its own justified it. Nothing was published and nothing was linked on a guess. Systems that get this wrong are not merely unhelpful, they attach a stranger's complaint to a real customer's account.

Audit, Deployment and Lifecycle

Audit trail

Every item produces a record. This is what makes the system defensible when a published reply or a missed complaint is questioned months later.

Recorded	Purpose
Original text and source	What the customer actually wrote, as collected.
Model and prompt version	Which configuration produced the reading.
Snapshot reference	What the system believed about the customer and catalogue at that moment.
Generated case	The exact structured output, before any human edit.
Validation outcome	Accepted, rejected, suppressed or amended, and why.
Reviewer decision	Who approved, edited or rejected, and when.
Published response	The text that actually went out, on which channel.
Outcome	Resolution and closure returned from the ticketing system.

Release management

The models, prompts, taxonomy, weights, schema and application are released as one signed bundle with a single version number. The schema version travels inside every case, so a ticketing system receiving a case always knows which contract it was built against.

Delivery phases

Phase	Scope	Exit criteria
Phase 0 **Assessment**	Channel inventory, taxonomy workshop, CRM and ticketing interface review, lawful basis and retention agreed.	Agreed taxonomy, schema and integration approach.
Phase 1 **Shadow mode**	Cases generated and scored but never dispatched, running alongside the current manual process.	Classification and grounding accuracy measured against human triage on real feedback.
Phase 2 **Supervised operation**	Cases created in the ticketing system, with mandatory human review on every response.	Zero ungrounded links and an agreed miss rate on critical items.
Phase 3 **Routine operation**	Review required only above a significance threshold, tuned per category and per market.	Detection latency and reviewer throughput targets met.
Phase 4 **Expansion**	Additional channels, languages, brands and markets.	Repeatable deployment package.

Benefits and Measurement

Benefits realised

Faster detection
Negative feedback is read within minutes of arrival rather than at the next review cycle.

Consistent triage
The same taxonomy and the same weights apply to every item, so severity does not depend on who is on shift.

One view across channels
The same defect reported on three platforms clusters into one issue with one owner.

Less manual monitoring
Staff review exceptions instead of reading everything that arrives.

Consistent brand voice
Drafts follow the response policy, and a person still approves before publication.

Earlier product signal
Recurring complaints reach engineering and quality as a measured trend rather than as anecdote.

Auditable decisions
Every case carries the source text, the model version and the reviewer who approved it.

Predictable cost
The cascade keeps cost per case roughly flat as feedback volume grows.

KPI framework

Measure	What it tells you
Detection latency	Time from the customer posting to the case being created.
Grounding accuracy	Share of cases correctly linked to the real customer and product.
Agreement with human triage	Category and severity agreement on a sampled set, per language.
Critical miss rate	Critical items not raised as critical. This is the number that matters.
False case rate	Cases raised that a reviewer rejects as noise.
Time to approved reply	How long the customer actually waits for a public response.
Open cluster count	Distinct recurring issues live at any one time.
Reviewer load	Items handled per reviewer hour, which tells you whether the thresholds are tuned.

BASELINE FIRST

Capture current detection latency, triage effort, manual ticket volume and complaint recurrence before deployment. Without a baseline the pilot produces numbers that cannot be defended, and that is the usual reason a working pilot fails to secure production funding.

Engineering Considerations and Risks

Known hard problems

Risk	Mitigation
Sarcasm and idiom	Evaluation set built from the brand's own feedback, and low confidence items routed to a human rather than scored.
Language coverage	Accuracy measured per language before a market goes live, never as a single global figure.
Wrong customer linkage	Grounding contract, unlinked by default, and no write path from the pipeline into the CRM.
Model drift	Sampled human audit every cycle, with the audit set refreshed as products and complaint patterns change.
Review fatigue	Thresholds tuned so reviewers see exceptions. A queue nobody reads is worse than no queue at all.
Platform interface change	Connectors isolated behind one internal interface, so a channel change never reaches the pipeline.
Brand and legal risk in drafts	Response policy carried in context, prohibited commitments blocked by rule, human approval before publication.
Coordinated campaigns	Burst detection and author clustering, so brigading is recognised rather than scored as many independent complaints.

Open questions for the customer

- Which channels are in scope for the first release, and are private messages and support transcripts included?
- Which system is the case of record, and what interface does it expose for case creation and status?
- Does a CRM integration exist that can supply customer identity and value tier, or is the world model limited to the catalogue?
- Which languages and markets are in scope, and what accuracy is acceptable in each of them?
- Who approves a public response, and does that approver change with severity or with market?
- What is the retention period for collected feedback, and which lawful basis applies in each market?
- Which complaint categories carry a regulatory reporting obligation for this business?

> **RECOMMENDED FIRST STEP**
>
> A shadow mode pilot on one brand and two channels. It generates, grounds and scores cases against real feedback without creating a single ticket, which produces measured classification and grounding accuracy at effectively zero operational risk. It also gives the customer experience team the evidence it needs before anything is allowed to reach a customer.

Contact

Website

https://tech4bizsolutions.com

Contact details

If you would like to know more about Tech4Biz Solutions
and our products please contact us via email contact@tech4biz.io

Address

Bangalore
NGR Arcade,
HSR Layout Sector 1,
Bangalore, Karnataka

Surat
306-307,
Millionaire Business Park,
LP Savani Road, Adajan,
Surat, Gujarat - 395009



Tech4Biz

Excellence Guaranteed!

Tech4Biz is a provider of comprehensive engineering and IT solutions for businesses of all sizes. From cloud computing and data management to security, networking and embedded intelligence, we have the expertise and experience to help your business stay ahead of the curve.

Our mission is to provide businesses with the best possible technology solutions. We understand that a reliable and efficient technology foundation is crucial in an increasingly competitive marketplace. That is why we offer a wide range of services, from managed IT and cloud computing to applied AI and data engineering.

This communication contains general information only, and Tech4Biz Solutions Private Limited is not, by means of this communication, rendering professional advice or services. Architecture descriptions, model estate, volume shares and integration details in this document represent a reference design and design targets for the stated configuration. They are confirmed against the specific customer channel mix, data volume and language coverage during the engineering phase. Third party product names are the property of their respective owners. Before making any decision or taking any action that may affect your finances or your business, you should consult a qualified professional adviser. Tech4Biz Solutions Private Limited shall not be responsible for any loss whatsoever sustained by any person who relies on this communication.

© 2026 Tech4Biz Solutions Private Limited