SALESFORCE CASE STUDY | SERVICE CLOUD

Scaling High-Volume Support for an On-Demand Delivery Platform

Stabilising a busy Service Cloud support operation through smarter triage, bulk resolution, tighter access governance, and engineering-linked issue tracking.

Authored by: Parag Bhatt | Salesforce Technical Lead / Senior Developer | 8.5 years of CRM delivery experience

---

> *One-liner: We stabilised and optimised a high-volume Service Cloud support operation — streamlining case triage, enabling bulk case resolution, tightening access governance, and connecting Salesforce to the engineering bug-tracking workflow so customer issues were resolved faster and tracked end to end.*

Engagement Snapshot

Client	QuickBasket Technologies Inc. (United States)
Industry	On-Demand Logistics / Marketplace Technology
Region	US-based client, delivered from India (Jaipur)
Salesforce Cloud	Service Cloud
My Role	Senior Salesforce Developer / Support Engineer
Team Size	Embedded support pod within the client's wider support and engineering team
Duration	Ongoing managed support engagement
Key Tools	Service Cloud, Apex, Flows, JIRA, Okta, VS Code, Amazon Console

1. Project Context

QuickBasket Technologies Inc. operates a large on-demand delivery marketplace serving merchants, couriers, and consumers across multiple US metros. Its internal and partner-facing support teams rely on Salesforce Service Cloud to manage a continuous, high-volume stream of operational cases, partner access requests, issue escalations, and agent workflows.

At this scale, even small process inefficiencies multiplied into large backlogs. A few minutes lost per case translated into real SLA risk, slower merchant resolutions, and agent downtime. I joined the engagement as part of the embedded support and development pod responsible for keeping the Service Cloud platform stable, responsive, and aligned with business operations.

The brief was not just to close tickets, but to remove the root causes behind recurring issues: diagnose defects, resolve case-hygiene problems in bulk, improve access governance, and create a clean handoff into the engineering backlog when defects required deeper code changes.

2. Problem Statement

The high-velocity support environment exposed several operational pain points that were dragging down resolution speed and platform stability:

- **High inbound case volume.** Without disciplined triage and automation, queues regularly risked backlog, inconsistent prioritisation, and slower first-response times.
- **Recurring platform bugs.** Support agents repeatedly hit defects in Salesforce flows, validation behaviour, and case handling, generating repeat tickets and frustration.
- **Bulk case and lead hygiene gaps.** Large numbers of stale cases and leads needed to be closed safely and efficiently, but manual one-by-one updates were too slow and error-prone.
- **Access and permission blockers.** Users were frequently blocked from applications and features they needed, creating avoidable agent downtime and operational delay.
- **Disconnected engineering escalation.** Issues that genuinely required development work lacked a clean bridge from support into the engineering backlog, causing tracking gaps and duplicated effort.

3. What We Built — The Solution

We acted as the technical and operational backbone of the Service Cloud support platform, delivering a support model that combined fast resolution with structural improvement:

- **Defect resolution at source —** reproduced and fixed bugs affecting day-to-day support operations, reducing repeat incidents rather than repeatedly treating symptoms.
- **Bulk case and lead resolution —** designed and executed safe backlog-clearing routines using data tools, automation, and controlled validation so thousands of records could be closed without integrity risk.
- **Access governance —** corrected profiles, permission sets, and identity-layer mappings so users could reliably access the apps and features they were entitled to.
- **Engineering escalation bridge —** triaged deeper defects into JIRA, with clear ownership and status tracking through to closure so nothing disappeared between teams.
- **Triage discipline —** established clearer intake, classification, and routing logic so the right issues landed with the right resolver group quickly.
- **Pod coordination —** used JIRA to manage workload, track priorities, and keep support and engineering aligned on progress and blockers.

4. My Role on the Engagement

As the senior Salesforce developer and support engineer on the engagement, I combined hands-on issue resolution with solution design, backlog management, and stakeholder communication. My responsibilities included:

- **Hands-on issue resolution —** first- and second-line technical resolution of cases, defects, and support blockers within Salesforce.

- **Solution guidance —** translating customer-reported problems into practical resolution paths and recommending durable fixes where process or design needed improvement.
- **Bulk operations ownership —** designing and running safe bulk-close processes for cases and leads, with validation controls to prevent collateral changes.
- **Access administration —** managing profiles, permission sets, and identity-linked access corrections through the Okta-integrated environment.
- **Work management —** coordinating tasks in JIRA, tracking escalations, and maintaining visibility on throughput, blockers, and engineering dependencies.

5. How We Achieved It — Delivery Approach

1. **Triage discipline.** Established a consistent intake-and-classify model so high-volume cases were prioritised correctly, resolved by the appropriate team, and no longer languished in the wrong queue.
2. **Root-cause defect fixes.** Reproduced reported issues, traced them through configuration and Apex logic in VS Code, and deployed fixes that eliminated recurring ticket patterns.
3. **Safe bulk operations.** Built and ran controlled bulk-close routines for stale cases and leads, validating selection criteria carefully to avoid accidental mass updates.
4. **Access provisioning.** Used profiles, permission sets, and the Okta identity layer to resolve user access blockers quickly and consistently, reducing agent downtime.
5. **JIRA-linked escalation.** Created a clean handoff for defects that required engineering intervention, converting them into tracked JIRA items with visible ownership and lifecycle status.
6. **Continuous workload management.** Managed the pod's throughput and priorities through JIRA so SLA-sensitive work was handled first and backlog stayed under control.

6. Outcomes and Business Impact

The result was a more stable, better-governed support operation that could absorb high volume without losing control of quality or visibility. Indicative outcomes from the engagement included:

~35%	~25%	100%	Improved
Reduction in support backlog through bulk cleanup	Faster issue resolution from root-cause fixes	Engineering escalations tracked through JIRA	Access reliability for internal support users

- Reduced case backlog through safe bulk-resolution capability and better triage.
- Faster issue resolution and fewer repeat tickets through root-cause fixes in the platform.
- Cleaner access governance, removing a common source of agent downtime.
- End-to-end traceability for support issues needing engineering via the Salesforce-to-JIRA bridge.
- A stable, well-managed support operation able to handle high case volume without sacrificing visibility.

Note: percentages above are indicative portfolio figures and can be replaced with the client's measured support KPIs before external publication.

7. Tech Stack

Service Cloud	**Case Management**	**Apex / Flows**
Profiles & Permission Sets	**Okta**	**JIRA**
Data Loader / Bulk Ops	**VS Code**	**Amazon Console**

8. Architecture & Process Diagrams

The diagrams below illustrate the core process flows and architecture decisions behind the solution.

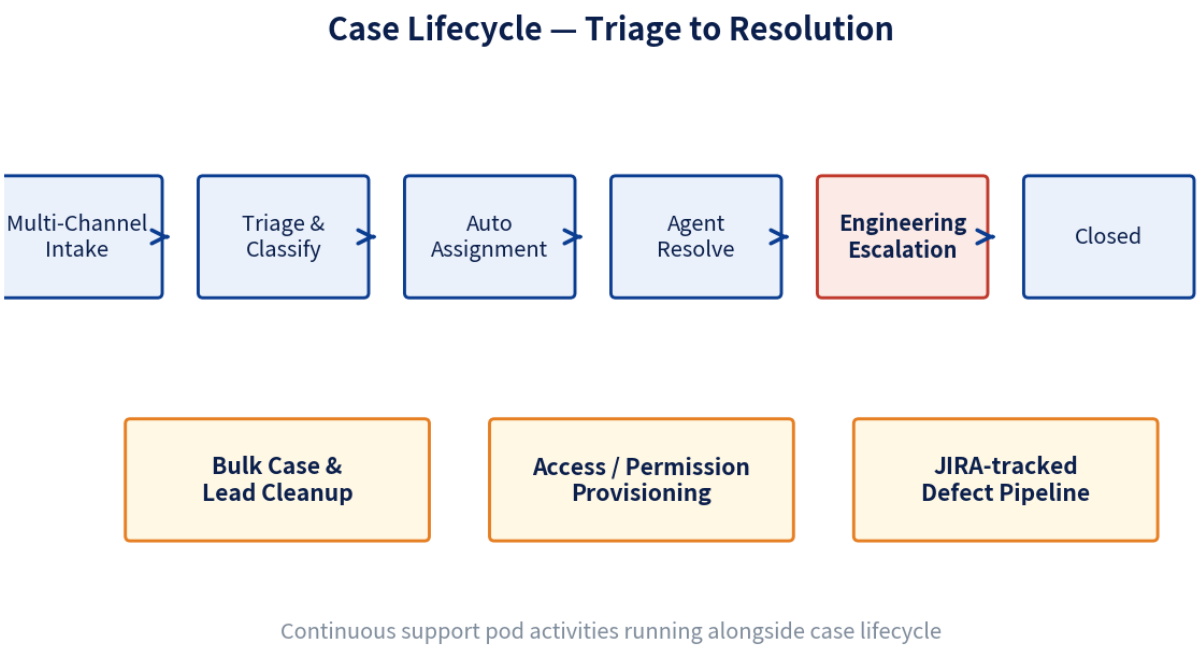

Figure 1 — Case lifecycle from multi-channel intake through to resolution, with parallel bulk-ops, access, and engineering tracks.

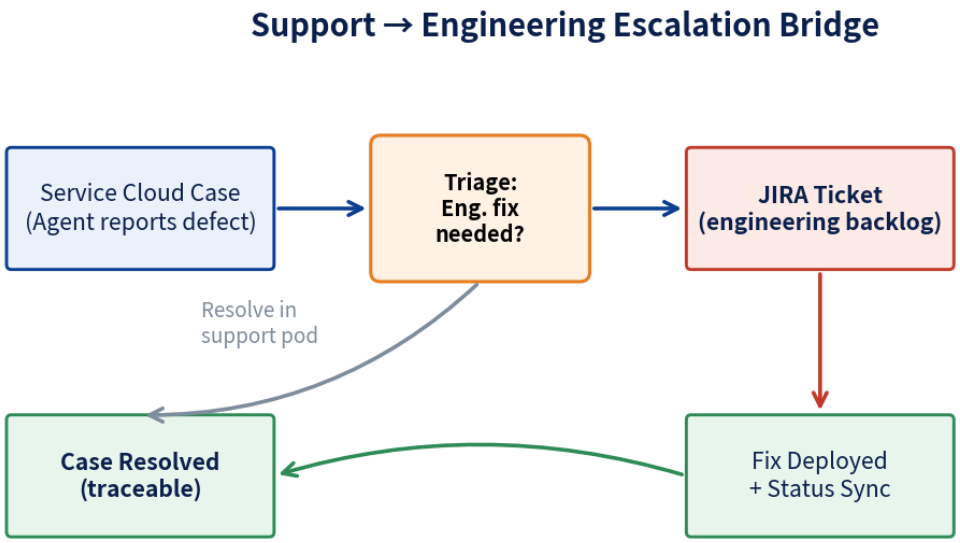

Figure 2 — Support-to-engineering escalation bridge connecting Service Cloud cases with the JIRA engineering backlog.

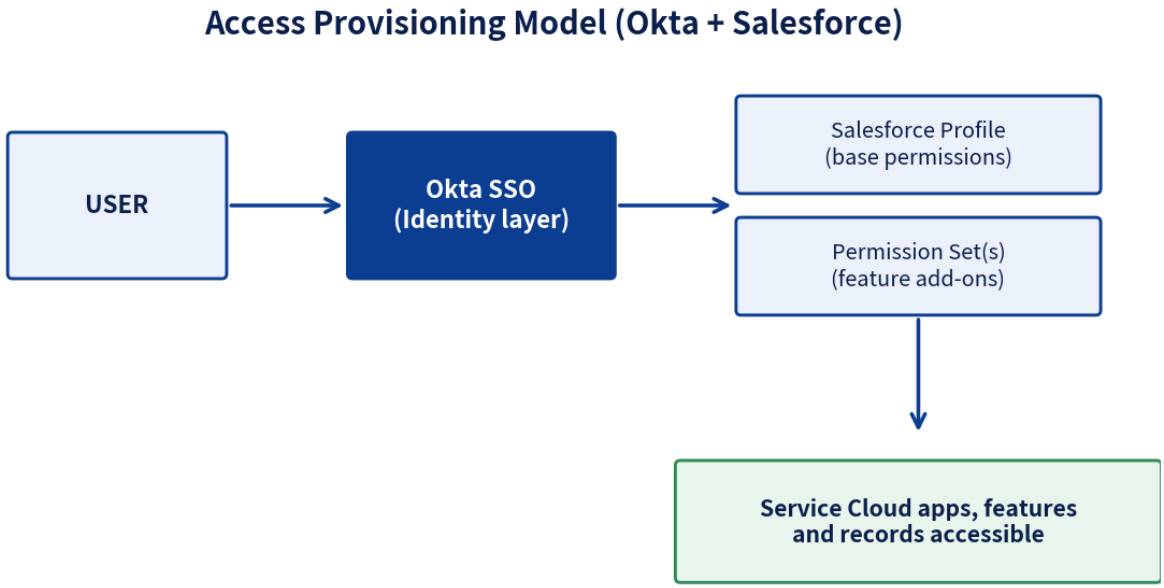

Figure 3 — Access provisioning model using Okta SSO, Salesforce profiles, and permission sets.

9. Key Takeaways

- **High-volume support needs structure, not heroics.** A disciplined triage and queueing model delivered more value than simply adding more hands to the problem.
- **Bulk operations must be safe by design.** Backlog clearance was only valuable because the update logic was validated carefully and executed with full data integrity control.
- **Access issues are operational issues.** Profiles, permission sets, and SSO alignment directly affect agent productivity and should be treated as a business-critical support layer.
- **Engineering handoff must be visible.** The JIRA bridge mattered because it turned vague escalations into accountable work with a trackable lifecycle.

About the Author

Parag Bhatt is a Salesforce Technical Lead and Senior Developer with 8.5 years of experience delivering CRM transformation programmes across Sales Cloud, Service Cloud, Experience Cloud, CPQ, Revenue Cloud, and Marketing Cloud. He has led offshore and hybrid delivery teams for clients across the GCC, North America, and APAC, with strong hands-on expertise in Apex architecture, automation design, integrations, e-signature, quote-to-cash, and customer engagement platforms.