Proactive support playbook template

Reactive support solves problems after they happen. Proactive support prevents them. This template helps support teams identify early warning signals, monitor customer sentiment, and act before issues escalate. Use it to create a repeatable system for detecting patterns, reducing repeat tickets, and improving the overall customer experience.

How to use this template:

- Fill this out for one product area or one segment first.
- Keep the first version simple. Add sophistication later.
- Proactive support = detection + action + communication (not just “fewer tickets”).

1) Your definition of proactive support

Create a clear, shared definition so your team isn't guessing what "proactive" means.

Proactive support definition (choose what applies)	
Identifying issues before customers report them	☐
Preventing repeat issues by giving prevention steps in every resolution	☐
Monitoring "silent failures" (connections breaking, data out-of-bounds, etc.)	☐
Reducing escalations by intervening before frustration spikes	☐
Enabling teams across time zones to spot patterns early ("follow-the-sun")	☐

Our definition:

e.g., Proactive support means identifying patterns early and taking action before customers experience impact.

What success looks like	
Fewer repeat tickets in top issue categories	☐
Faster detection → faster mitigation (issue addressed before peak hours)	☐
Lower escalation rate/fewer “ask for manager” moments	☐
Higher customer confidence (“you knew before we did”)	☐
Better internal alignment (no “loop of despair” between teams)	☐

Primary success metric:

e.g., Time to Detect (TTD) for the top 5 issue categories

Secondary success metric:

e.g., Escalation rate (% of tickets escalated to engineering)

2) Proactive detection signals

Define the “early warning system” inputs you’ll watch to spot patterns before customers do.

Signal checklist (pick 3–6 to start):

Operational signals	
Monitoring alerts (uptime, latency, error rates)	☐
Connection health/integration failures	☐
Log anomalies (Datadog, platform logs, etc.)	☐
Spikes in ticket volume by tag/feature	☐
Repeat ticket patterns (same root cause)	☐
Customer signals	
Frustration score/negative sentiment	☐
Escalation triggers (keywords, tone, threats to churn)	☐
High-impact customers reporting similar issues	☐

Signals we will track first (list 3):

---

e.g., Ticket spikes by feature area; Integration failures/hour; Frustration score > 60

---

3) The “tag and track” workflow

A lightweight process to capture patterns as they happen (not month-end).

Tagging rules:

- Every ticket gets a feature area tag
- Every escalation gets an escalation reason tag
- Every repeat issue gets a root cause tag (when known)
- Tagging happens during resolution, not after

Tracking cadence:

- Weekly pattern review (30 minutes)
- Real-time alert when thresholds hit

4) Proactive thresholds and alerts

Clear “when to act” rules so teams don’t hesitate or over-escalate.

Signal	Threshold	What happens when triggered	Owner
Ticket spike (tagged)	*e.g., +30% WoW*	*e.g., Create incident thread and notify Eng*	*e.g., Support Ops*
Connection failures	*e.g., >10/hour*	*e.g., Notify on-call and prep customer comms*	*e.g., On-call lead*
Frustration score	*e.g., >60*	*e.g., Manager review and intervene within 2 hrs*	*e.g., Team lead*
Repeat issue count	*e.g., >5/week*	*e.g., Create KB and macro and product bug intake*	*e.g., KB owner*

5) Frustration radar worksheet

A repeatable way to detect “this is about to blow up” and intervene early.

Frustration scoring rubric (customize):

- 0–30: neutral/normal support tone
- 31–60: mild frustration (risk rising)
- 61–120: high frustration (needs intervention)
- 121–200: critical (escalation likely; churn language possible)

Our threshold for manager review:

---

e.g., Score > 60 OR keywords indicating churn risk

Frustration dashboard

Customer/account	Ticket	Score	Why it scored high	What we’ll do now	Owner
e.g. company name	*#4562*	*65*	*Customer mentioned “this keeps happening” and referenced previous tickets.*	*Investigate recurring root cause, create KB article and macro to prevent repeat issue.*	*Support ops*

Intervention checklist	
Manager reviews context (not just the score)	☐
Acknowledge emotion and restate impact	☐
Offer next step and timeline (clear expectations)	☐
Update internal stakeholders (AM/CSM/SE)	☐
Add prevention steps and KB/macro follow-up	☐

6) Completeness macro builder

A standard to ensure every resolution includes prevention, not just a fix.

Completeness checklist (for every response)	
Answered the immediate question/fixed the issue	☐
Included prevention steps (“next time, do X”)	☐
Explained why we need this info (screenshots/logs)	☐
Included next steps and what happens if it persists	☐
Used a human tone (not robotic)	☐

Macro template (copy/paste and customize)

Fix/answer

Here’s what caused the issue and how to resolve it:

e.g., The token expired; re-authenticate the integration and retry.

Prevention (the proactive part)

To prevent this happening again, please:

---

e.g., Rotate tokens every 90 days and store expiry dates in your runbook.

Why we're asking for info

To confirm root cause, I'm asking for [X] because [X].

---

e.g., a screenshot of the error banner because it contains the error code and timestamp.

Next step and timing

Next, we'll [X]. You can expect an update by [X].

---

e.g., validate logs and share a fix. You can expect an update by EOD Friday.

---

7) Cross-functional proactive comms

Prevents internal chaos (AM doesn't know, sales escalates, engineering gets pinged twice).

"Overcommunicate" rules	
One shared internal channel for major issues (per account or incident)	☐
Stakeholders included early: Support, Eng, AM/CSM, SE, Finance/Legal if needed	☐
Updates posted on a schedule (not ad hoc)	☐

Stakeholder channel setup

Channel/thread name	Used for	Who must be included	Update cadence
e.g., #incident-payments-api	*Incident/widespread issue*	*Support + Eng + CS/AM*	*e.g., every 2 hrs*
e.g., #acct-ACME-risk	*Strategic account risk*	*Support + CS/AM + SE*	*e.g., daily*
e.g., #bugs-top-patterns	*Product bug pattern review*	*Support + Product + Eng*	*e.g., weekly*

8) “Shared responsibility” customer messaging

A respectful way to handle cases where customer inputs/config cause issues.

Shared responsibility script (adapt):

It looks like the platform behaved unexpectedly because of the data/config being sent into it.

Here’s what we can do on our side:

e.g., Add validation and improve error messaging for this endpoint.

Here’s what you can adjust on your side to prevent this:

e.g., Validate payloads against the schema before sending; cap batch size to 5,000 records.

If helpful, we can share a checklist/template to validate inputs before sending them.

Customer-side prevention checklist we’ll provide

- Validation rule/schema check
- Example “good input” and “bad input”
- Limits/guardrails (out-of-bounds prevention)
- Monitoring suggestion (optional)

9) Team enablement to “think proactive”

Training and habits that help teams spot patterns, not just close tickets.

Training principles	
Train beyond tier boundaries where needed (especially startups/small teams)	☐
Equip everyone with the tools (logs, SQL, monitoring dashboards)	☐
Teach confidence in escalation decisions (reduce hesitation)	☐

Proactive habit loop (weekly)

Proactive habit loop (weekly)	
Team topic discussion: What does proactive mean this week?	☐
Share 1 example of proactive prevention from a resolved ticket	☐
Create or improve 1 macro/KB article from a recurring issue	☐
Review 3 random tickets for completeness and tone	☐

10) Prioritization matrix (who gets proactive help first)

A consistent way to choose where proactive resources go.

Scoring model (recommended: blend)

Use these three inputs: ARR/tier (revenue impact), sentiment/frustration (reputational and churn risk), feature area pattern (systemic product issue).

Prioritization table:

Account/segment	ARR tier (1–5)	Sentiment risk (1–5)	Feature pattern severity (1–5)	Total	Action

Action rules (example):

- 12–15: Immediate proactive outreach and Eng escalation
- 8–11: Monitor and preventative comms and macro/KB update
- <7: Standard handling and track for trends

11) Metrics dashboard (short and practical)

Proves proactive is working without overcomplicating measurement.

Choose 4–6 to track:

- Time to detect (TTD)
- Time to mitigate (TTM)
- Ticket spikes by feature area
- Repeat ticket rate for top 5 issues
- Escalation rate
- Frustration score trend

Metric	Baseline	Target	How we measure	Cadence
e.g., Time to Detect (TTD)	*e.g., 3 hrs*	*e.g., <1 hr*	*e.g., monitoring alert → first triage*	*e.g., weekly*
e.g., Escalation rate	*e.g., 18%*	*e.g., 12%*	*e.g., escalated tickets/total*	*e.g., monthly*
e.g., Repeat ticket rate (Top 5)	*e.g., 22%*	*e.g., 15%*	*e.g., repeat-tagged/total*	*e.g., monthly*
e.g., Frustration score avg	*e.g., 48*	*e.g., 35*	*e.g., LLM rubric and manual QA*	*e.g., weekly*

e.g., Ticket spike events	*e.g., 6/mo*	*e.g., 3/mo*	*e.g., threshold breaches*	*e.g., monthly*
e.g., "Prevented escalations"	*e.g., 0*	*e.g., 10/mo*	*e.g., manager intervened pre-escalation*	*e.g., monthly*

12) Your first 30-day rollout plan

A simple “start small” implementation plan.

Week 1: Foundation

- Define proactive and success metric
- Choose 3–6 signals
- Standardize tags (feature, root cause, escalation reason)

Week 2: Early alerts and comms

- Set thresholds and alert workflow
- Create stakeholder channel process

Week 3: Frustration radar and macros

- Launch frustration scoring and manager review threshold
- Implement completeness checklist and 3 core macros

Week 4: Review and refine

- Identify top 3 recurring issues and create KB/macro fixes
- Adjust thresholds and ownership
- Share results with Product/Eng/CS