CLAUDE COWORK

Customer Support Plugin Handbook

Version:

VERSION	1.0.0
PLUGIN	customer-support

RationalEyes.ai
contact@rationaleyes.ai

February 2026

Table of Contents

Version: 1.0.0

Author: Anthropic

Last Updated: February 14, 2026

Part I: Overview & Quick Start

What Is This Plugin?

The **customer-support** plugin transforms Claude into an AI-powered support co-pilot for customer service teams. It provides structured workflows for ticket triage, customer research, response drafting, escalation management, and knowledge base authoring — reducing resolution times, improving response quality, and building institutional knowledge.

Official Description: "Triage tickets, draft responses, escalate issues, and build your knowledge base. Research customer context and turn resolved issues into self-service content."

Business Problems This Plugin Solves

PROBLEM	HOW THIS PLUGIN HELPS
Inconsistent ticket prioritization	Structured P1-P4 framework with clear routing rules
Time spent researching customer questions	Multi-source research with confidence scoring and attribution
Poorly written customer responses	Professional response templates adapted to situation and tone
Lost context during escalations	Structured escalation briefs with impact assessment and reproduction steps
Repeated questions that should be self-service	KB article generation from resolved tickets to reduce future volume
New hires ramping slowly	Codified best practices in ticket triage, research, and communication
Tribal knowledge locked in individuals' heads	Institutional knowledge captured in searchable, maintainable articles

Who Should Use This Plugin

Primary Users:

- **Customer support representatives** (L1/L2 support) — triage, research, respond
- **Support managers** — standardize workflows, maintain knowledge base

- **Customer success teams** — draft high-touch customer communications
- **Technical support engineers** — package escalations with full technical context

Secondary Beneficiaries:

- **Engineering teams** — receive better-structured bug reports with reproduction steps
- **Product teams** — see aggregated feature request patterns with business impact
- **Leadership** — visibility into support patterns and escalation trends

Quick Start: Install, Activate, Run

Step 1: Install the Plugin

Cowork (recommended): Open **Plugin Settings** in the Cowork desktop app, find **Customer Support**, and click **Install**. The plugin activates immediately — no CLI required.

Claude Code CLI (alternative): If you are using Claude Code in the terminal rather than Cowork, install via:

```
claude plugins add knowledge-work-plugins/customer-support
```

> **Note:** *All standard Cowork plugins, including Customer Support, are available from Plugin Settings with a single click. The CLI command above is only needed for Claude Code terminal users.*

No additional configuration is required to start using it.

Step 2: Connect Your Tools (Optional but Recommended)

The plugin works standalone but shines when connected to your support stack. Connect via Claude settings:

- **Support platform** (Intercom, Zendesk, Freshdesk, HubSpot Service Hub)
- **Knowledge base** (Guru, Notion, Confluence, Help Scout)
- **Project tracker** (Atlassian/Jira, Linear, Asana)
- **CRM** (HubSpot, Salesforce, Pipedrive)
- **Chat** (Slack, Microsoft Teams)
- **Email** (Microsoft 365, Google Workspace)

See Part IV: Connectors & MCP Servers for setup instructions.

Step 3: Run Your First Command

Try triaging a sample ticket:

```
/triage Customer says their dashboard has been showing a blank page since this morning
```

Claude will:

1. Categorize the issue (Bug)
2. Assign priority (P2 — High, since it's an Enterprise customer with a broken core feature)
3. Recommend routing (Tier 2 Support → investigate, then escalate to Engineering if reproducible)
4. Draft an initial response acknowledging the issue
5. Suggest next steps

Expected output structure:

```
## Triage: Dashboard blank page — Enterprise customer

**Category:** Bug
**Priority:** P2 — High (core feature broken, enterprise customer)
**Product area:** Dashboard / Frontend

### Routing Recommendation
Route to: Tier 2 Support (investigate browser/account specifics,
then escalate to Engineering if reproducible)

### Suggested Initial Response
[Draft acknowledging the issue, asking clarifying questions
about browser and any recent changes, offering status page link]
```

Step 4: Explore Other Commands

```
# Research a customer question across all connected sources
/research Does our platform support SSO with Okta?

# Draft a professional response for any situation
/draft-response Customer escalation — their integration has been down for 2 days

# Package an escalation for engineering
/escalate API returning 500 errors intermittently — 3 Enterprise customers reported

# Create a KB article from a resolved issue
/kb-article How to configure webhook notifications — resolved this for 3 customers thi
```

Plugin Architecture Diagram

```
┌──────────────────────────────────────────────────────────────────┐
│                     CUSTOMER SUPPORT PLUGIN                      │
├──────────────────────────────────────────────────────────────────┤
│                                                                  │
│                                                                   │
│  COMMANDS (User-Invoked Actions)                                 │
│                                                                 │
│  ┌────────────┬─────────────┬──────────────┬────────────────┐   │
│  │ /triage    │ /research   │ /draft-resp  │ /escalate      │   │
│  │            │             │              │ /kb-article    │   │
│  └─────┬──────┴──────┬──────┴───────┬──────┴───────┬────────┘   │
│        │             │              │              │             │
│        └─────────────┼──────────────┼──────────────┘             │
│                      │              │                            │
│                      │              │                            │
│  SKILLS (Auto-Activated Domain Knowledge)                        │
│                      │              │                          │
│  ┌──────────────────┬─────────────────┬──────────────────────┐   │
│  │ ticket-triage    │ customer-       │ response-drafting    │   │
│  │ • Categories     │   research      │ • Tone guidelines    │   │
│  │ • P1-P4 priority │ • Multi-source  │ • Templates          │   │
│  │ • Routing rules  │ • Confidence    │ • Situation types    │   │
│  │                  │   scoring       │                      │   │
│  ├──────────────────┼─────────────────┼──────────────────────┤   │
│  │ escalation       │ knowledge-      │                      │   │
│  │ • Tiers          │   management    │                      │   │
│  │ • Impact assess  │ • Article types │                      │   │
│  │ • Repro steps    │ • Searchability │                      │   │
│  └──────────────────┴─────────────────┴──────────────────────┘   │
│                               │                                  │
│                               ▼                                  │
│  MCP SERVERS (External Tool Connections)                         │
│                                                                  │
│  ┌───────────────────────────────────────────────────────────┐   │
│  │ Slack    Intercom   HubSpot    Guru     Atlassian         │   │
│  │ Notion   MS365                                              │   │
│  └───────────────────────────────────────────────────────────┘  │
│                                                                   │
│  DATA FLOW EXAMPLE:                                              │
│  User invokes → /triage "Customer can't export data"            │
│        ↓                                                         │
│  ticket-triage skill loads → applies P1-P4 framework            │
│        ↓                                                         │
│  Searches Intercom → finds similar tickets                      │
│        ↓                                                         │
│  Searches Guru → finds known issue article                      │
│        ↓                                                         │
│  Generates structured triage with category, priority,           │
│  routing recommendation, and suggested response                 │
│                                                                   │
└─────────────────────────────────────────────────────────────────┘
```

Key Relationships:

- **Commands invoke skills automatically** — `/triage` triggers `ticket-triage`, `/research` triggers `customer-research`, etc.
- **Skills provide domain knowledge** — frameworks, taxonomies, best practices, templates
- **MCP servers provide data** — ticket history, KB articles, customer context, internal docs
- **Commands orchestrate workflows** — combine skill knowledge + MCP data → structured output

Part II: Commands Reference

The plugin provides five slash commands covering the core support workflows: triage, research, response drafting, escalation, and knowledge capture.

Command 1: `/triage`

File: `/commands/triage.md`

Syntax

```
/triage <ticket text, customer message, or issue description>
```

What It Does

Categorizes, prioritizes, and routes an incoming support ticket or customer issue. Produces a structured triage assessment including:

- Primary and secondary category (bug, how-to, feature request, billing, account, integration, security, data, performance)
- Priority level (P1–P4) with justification
- Product area identification
- Routing recommendation (which team/queue should handle this)
- Suggested initial customer response
- Internal notes for the next agent

Required Inputs

- **Ticket description:** The customer's issue, question, or request (as free text)

Optional Inputs

- Customer name or account (helps pull context from CRM)
- Ticket ID or reference (pulls full thread from support platform)

Expected Output

```
## Triage: [One-line issue summary]

**Category:** [Primary] / [Secondary if applicable]
**Priority:** [P1-P4] — [Brief justification]
**Product area:** [Area/team]

### Issue Summary
[2-3 sentence summary of what the customer is experiencing]

### Key Details
- **Customer:** [Name/account if known]
- **Impact:** [Who and what is affected]
- **Workaround:** [Available / Not available / Unknown]
- **Related tickets:** [Links to similar issues if found]
- **Known issue:** [Yes — link / No / Checking]

### Routing Recommendation
**Route to:** [Team or queue]
**Why:** [Brief reasoning]

### Suggested Initial Response
[Draft first response — acknowledges issue, sets expectations,
provides workaround if available]

### Internal Notes
- [Additional context for the agent picking this up]
- [Reproduction hints if it's a bug]
- [Escalation triggers to watch for]
```

Realistic Walkthrough Example

User input:

```
/triage Customer says they were charged twice for their subscription this month. Accou
```

Claude's process:

1. **Parse the issue:** Billing error, duplicate charge, Enterprise customer, already waited 1 day
2. **Check category taxonomy (from ticket-triage skill):** Category = Billing
3. **Assess priority (from ticket-triage skill):**
   - Impact: Financial issue, Enterprise customer
   - Urgency: Customer already waiting 1 day, following up

   - Priority: **P2 — High** (billing issues need prompt attention, especially when customer is already waiting)
4. **Search for context (if CRM/support platform connected):**
   - Pull Acme Corp account details from HubSpot
   - Search Intercom for previous billing tickets from this customer
5. **Determine routing:** Billing/Finance team (not frontline support — requires account access)
6. **Draft response** using billing template from ticket-triage skill

Claude's output:

```
## Triage: Duplicate subscription charge — Acme Corp (Enterprise)

**Category:** Billing
**Priority:** P2 — High (financial issue, Enterprise customer, already waiting 1 day)
**Product area:** Billing / Payments

### Issue Summary
Acme Corp was charged twice for their subscription this month. They reported
this yesterday via email and are following up today. This is an Enterprise
account and the duplicate charge needs immediate investigation and resolution.

### Key Details
- **Customer:** Acme Corp (Enterprise plan, ARR: $120K)
- **Impact:** Financial — double charge for monthly subscription
- **Workaround:** Not applicable (billing issue)
- **Related tickets:** No recent billing issues for this account
- **Known issue:** No — appears to be account-specific

### Routing Recommendation
**Route to:** Billing/Finance team
**Why:** Requires account access to verify charges, process refund if needed.
Not a frontline support issue.

### Suggested Initial Response

Hi [Contact Name],

Thank you for reaching out, and I apologize for the delay in responding.
I can see you were charged twice for your subscription this month, and
I understand how concerning that is.

I've escalated this to our billing team for immediate investigation.
They'll review your account and process a refund if a duplicate charge
occurred.

You'll hear from our billing team within 4 hours with an update. If you
need to speak with someone urgently, please let me know and I'll arrange
```

```
a call.

I'm sorry for the inconvenience.

Best,
[Your name]

### Internal Notes
- Customer already waited 1 day — SLA breach risk if not handled today
- Enterprise account — high retention priority
- Check if this is a payment processor issue vs. subscription system bug
- If duplicate charge confirmed, process refund immediately and consider
  proactive credit for the inconvenience
```

Tips and Common Pitfalls

Tips:

- **Always paste the full customer message** when available — context in later sentences often changes the assessment
- **Include customer/account name** if you know it — Claude will pull CRM context automatically
- **Mention time pressure explicitly** — "customer's deadline is Friday" or "third time they've reported this"
- **Use `/triage` even for seemingly simple tickets** — it catches priority signals you might miss

Common Pitfalls:

- **Categorizing by symptom instead of root cause** — "can't log in" might be Bug (authentication broken) or Account (password reset needed). Let Claude investigate.
- **Under-prioritizing based on politeness** — a calm, polite message about a production outage is still P1
- **Skipping duplicate detection** — Claude checks for similar tickets automatically, but only if your support platform is connected
- **Not providing enough context** — "customer has an issue with export" is too vague. Include the symptom: "export returns 0 rows when data exists"

Command 2: `/research`

File: `/commands/research.md`

Syntax

```
/research <question or topic>
```

What It Does

Conducts multi-source research on a customer question, product topic, or account-related inquiry. Synthesizes findings from all available sources (documentation, knowledge base, CRM, chat, email, support platform, web) with clear attribution and confidence scoring.

Required Inputs

- **Question or topic**: What you need to find out (can be phrased as a question or topic description)

Optional Inputs

- Customer or account name (focuses research on customer-specific context)
- Specific source to prioritize (e.g., "check our internal wiki first")

Expected Output

```
## Research: [Question/Topic]

### Answer
[Clear, direct answer to the question — bottom-line first]

**Confidence:** [High / Medium / Low]
[Explanation of what drives the confidence level]

### Key Findings

**From [Source 1]:**
- [Finding with specific detail]
- [Finding with specific detail]

**From [Source 2]:**
- [Finding with specific detail]

### Context & Nuance
[Any caveats, edge cases, or additional context that matters]

### Sources
1. [Source name/link] — [what it contributed]
2. [Source name/link] — [what it contributed]
3. [Source name/link] — [what it contributed]

### Gaps & Unknowns
- [What couldn't be confirmed]
- [What might need verification from SME]

### Recommended Next Steps
- [Action if answer needs to go to customer]
- [Action if further research needed]
- [Who to consult for verification if needed]
```

Realistic Walkthrough Example

User input:

```
/research Does our platform support SSO with Okta?
```

Claude's process:

1. **Understand the question type (from customer-research skill):** Customer question — needs definitive answer about product capability

2. **Plan search strategy (from customer-research skill):**
    - Tier 1: Product documentation, knowledge base
    - Tier 2: Support tickets (has anyone set this up before?)
    - Tier 3: Internal chat (recent discussions about SSO)
    - Tier 4: Web search (general Okta integration info)
3. **Execute searches systematically:**
    - Search Guru knowledge base → finds article "Configuring SSO with SAML Providers"
    - Search Notion docs → finds "Supported SSO Providers" page
    - Search Intercom → finds 3 resolved tickets where customers set up Okta SSO
    - Search Slack #support channel → finds discussion from last week about Okta setup
4. **Synthesize and validate:**
    - All sources confirm: Yes, Okta is supported via SAML 2.0
    - Guru article has step-by-step setup instructions
    - Intercom shows successful implementations
5. **Assign confidence:** High (official docs + multiple customer successes)

Claude's output:

```
## Research: Does our platform support SSO with Okta?

### Answer
Yes, our platform supports SSO with Okta via SAML 2.0. Setup takes
approximately 30-45 minutes and requires Admin access in both your
platform account and your Okta admin console.

**Confidence:** High

This is confirmed in our official documentation and we have multiple
customers successfully using Okta SSO (verified in support ticket history).

### Key Findings

**From Guru Knowledge Base:**
- Article "Configuring SSO with SAML Providers" includes Okta-specific
  setup instructions
- SAML 2.0 is the supported protocol
- Requires Enterprise plan or higher
- Setup involves exchanging metadata between platforms

**From Notion Documentation:**
- "Supported SSO Providers" page explicitly lists Okta
- Also supports Azure AD, Google Workspace, OneLogin
- JIT (Just-In-Time) provisioning is supported
```

```
**From Intercom Support Tickets:**
- 3 customers successfully set up Okta SSO in the past 6 months
- Average setup time: 30-45 minutes
- Common issue: certificate mismatch (resolved by re-downloading metadata)

**From Slack #support:**
- Discussion last week confirmed Okta SSO works with MFA enabled
- Engineering confirmed no known issues with Okta integration

### Context & Nuance
- **Plan requirement:** Enterprise plan or higher (not available on Pro/Standard)
- **Setup complexity:** Moderate — requires admin access in both systems
- **MFA compatibility:** Works with Okta MFA enabled
- **Common issues:** Certificate mismatches during setup (easily resolved)

### Sources
1. Guru KB: "Configuring SSO with SAML Providers" (verified current as of Jan 2026)
2. Notion: "Supported SSO Providers" page (updated Dec 2025)
3. Intercom: Tickets #4521, #4890, #5123 (successful Okta setups)
4. Slack #support: Thread from Feb 8, 2026 (MFA compatibility confirmed)

### Gaps & Unknowns
- None — this is a well-documented, commonly used integration

### Recommended Next Steps
- **If going to customer:** Share the Guru KB article link directly
- **If customer needs help:** Offer to walk them through setup or
  connect them with a solutions engineer
- **No verification needed:** High confidence, ready to share
```

Tips and Common Pitfalls

Tips:

- **Be specific with your question** — "SSO support" is vague; "SSO with Okta" is specific and yields better results
- **Include customer name** if the question is account-specific — "Does Acme Corp's plan include API access?"
- **Let Claude search all sources** — don't prematurely limit to one source unless you have good reason
- **Use confidence level to decide next steps** — High confidence → share with customer; Low confidence → verify with SME first

Common Pitfalls:

- **Asking multiple questions at once** — "Does it support SSO and SCIM and MFA?" Split into separate `/research` calls
- **Not specifying the use case** — "How does the API work?" is too broad. "How do I authenticate API requests?" is specific.
- **Assuming web search is enough** — internal docs and customer history often contain nuances that public info doesn't
- **Sharing low-confidence answers with customers** — if confidence is Low, verify internally first

---

Command 3: `/draft-response`

File: `/commands/draft-response.md`

Syntax

```
/draft-response <situation description>
```

What It Does

Drafts a professional, customer-facing response tailored to the situation, customer relationship stage, stakeholder level, and communication channel. Applies tone guidelines, templates, and best practices from the response-drafting skill.

Required Inputs

- **Situation description**: What you're responding to (customer question, issue, escalation, announcement, etc.)

Optional Inputs

- Customer name or account (pulls context and relationship history)
- Channel (email, ticket, chat) — Claude infers if not specified
- Desired tone adjustment (e.g., "make this more empathetic" or "more technical")

Expected Output

```
## Draft Response

**To:** [Customer contact name]
**Re:** [Subject/topic]
**Channel:** [Email / Ticket / Chat]
**Tone:** [Empathetic / Professional / Technical / Celebratory / Candid]

---

[Draft response text]

---

### Notes for You (internal — do not send)
- **Why this approach:** [Rationale for tone and content choices]
- **Things to verify:** [Any facts or commitments to confirm before sending]
- **Risk factors:** [Anything sensitive about this response]
- **Follow-up needed:** [Actions to take after sending]
- **Escalation note:** [If this should be reviewed by someone else first]
```

Realistic Walkthrough Example

User input:

```
/draft-response Customer escalation — their integration has been down for 2 days and t
```

Claude's process:

1. **Understand context (from response-drafting skill):**
   - Situation type: Escalation, bad news (it's our bug), issue resolution
   - Customer: Enterprise, high ARR, already frustrated
   - Urgency: High (2 days is too long)
   - Channel: Email (likely, given the escalation)
   - Tone needed: Accountable, empathetic, action-oriented
2. **Research customer history (if CRM connected):**
   - Pull TechCorp account from HubSpot
   - Check previous escalations (have they escalated before?)
   - Identify primary contact and role
3. **Select response template** (from response-drafting skill): "Escalation response" template
4. **Draft response** following principles:

- Lead with empathy and acknowledgment
- Take ownership (no deflecting)
- Be specific about what happened and what's being done
- Provide timeline
- Offer direct escalation path

Claude's output:

```
## Draft Response

**To:** Sarah Chen, CTO @ TechCorp
**Re:** Integration Downtime — Status Update and Resolution Plan
**Channel:** Email
**Tone:** Accountable, empathetic, action-oriented

---

Hi Sarah,

I want to start by apologizing for the disruption to your integration over
the past two days. I know your team relies on this connection, and I
understand how frustrating it is when something that should "just work"
doesn't.

Here's what happened and what we're doing about it:

**What Happened**
We identified a rate limiting bug in our API that was incorrectly flagging
your integration's requests as exceeding limits, even though they were well
within your quota. This started on Tuesday morning and wasn't caught by our
monitoring until your team reported it.

**What We're Doing**
- Engineering deployed a fix 2 hours ago that corrects the rate limit
  calculation
- We're monitoring your integration specifically to confirm it's stable
- We've added additional monitoring to catch this type of issue faster in
  the future

**Current Status**
Your integration should be working normally now. We're seeing successful
API calls flowing as of 3:00 PM today. Can you confirm on your end that
things are back to normal?

**What's Next**
- I'll check in with you tomorrow to make sure everything remains stable
- We're conducting a root cause analysis this week to understand how this
```

```
  slipped through our testing
- I'd like to schedule a call next week to discuss how we can prevent this
  from affecting you in the future — would Thursday or Friday work?

I'm personally tracking this and am available if you need anything. My
direct number is [phone] if you'd prefer to discuss by phone.

Again, I'm sorry for the disruption. We take this seriously and we're
committed to earning back your trust.

Best,
[Your name]

---

### Notes for You (internal — do not send)

**Why this approach:**
- Heavy empathy up front given 2-day outage and customer frustration
- Technical specificity (rate limit bug) shows we understand the issue
- Personal accountability ("I'm personally tracking") given high ARR
- Proactive follow-up offer (call next week) to rebuild relationship

**Things to verify before sending:**
- ✓ Confirm the fix is deployed (check with engineering)
- ✓ Confirm their integration is actually working now (check logs)
- ✓ Get approval for phone number sharing (or use general support line)
- ✓ Check if Sarah is the right contact or if there's a technical lead

**Risk factors:**
- High-value customer ($200K ARR) — churn risk if not handled well
- This is their second escalation this quarter (per CRM notes) — pattern
  of issues eroding trust
- May ask for SLA credit or contract concession — be prepared to escalate
  to account executive if they request compensation

**Follow-up needed:**
- Send this response TODAY (already 2 days in)
- Set reminder to check in tomorrow
- Schedule internal post-mortem with engineering
- Brief account executive on situation (high churn risk)

**Escalation note:**
- Loop in VP of Support before sending (high-value account)
- Have account executive standing by in case Sarah wants to discuss
  contract/credit
```

Tips and Common Pitfalls

Tips:

- **Provide full context** — customer name, account tier, issue type, timeline all affect tone
- **Specify the channel** if it matters — chat responses are shorter, emails are more formal
- **Use the internal notes** — Claude flags risks and verification steps you shouldn't skip
- **Ask for tone adjustments** — "make this more empathetic" or "more technical for an engineer"
- **Reference customer history** — if you know they've escalated before, mention it so Claude adjusts approach

Common Pitfalls:

- **Sending the first draft without reading the internal notes** — Claude often flags things to verify or people to loop in
- **Over-promising to calm a customer** — Claude's drafts are conservative; don't add commitments Claude didn't include
- **Using the same tone for all situations** — a billing inquiry needs precision, an outage needs empathy, good news can be enthusiastic
- **Forgetting to verify facts** — if Claude says "this should be fixed," confirm it actually is before sending

---

Command 4: `/escalate`

File: `/commands/escalate.md`

Syntax

```
/escalate <issue description> [customer name or account]
```

What It Does

Packages a support issue into a structured escalation brief for engineering, product, security, or leadership. Gathers context from connected sources, structures reproduction steps (for bugs), assesses business impact, and identifies the appropriate escalation target.

Required Inputs

- **Issue description:** What's broken, needed, or requires decision

Optional Inputs

- Customer name or account (pulls account context, ARR, previous escalations)
- Ticket ID or reference (pulls full troubleshooting history)

Expected Output

```
## ESCALATION: [One-line summary]

**Severity:** [Critical / High / Medium]
**Target team:** [Engineering / Product / Security / Leadership]
**Reported by:** [Your name/team]
**Date:** [Today's date]

### Impact
- **Customers affected:** [Who and how many]
- **Workflow impact:** [What they can't do]
- **Revenue at risk:** [If applicable]
- **Time in queue:** [How long this has been an issue]

### Issue Description
[Clear, concise description — 3-5 sentences]

### What's Been Tried
1. [Action] → [Result]
2. [Action] → [Result]
3. [Action] → [Result]

### Reproduction Steps
[If applicable — structured with environment details and evidence]

### Customer Communication
- **Last update to customer:** [Date and what was communicated]
- **Customer expectation:** [What they expect and by when]
- **Escalation risk:** [Will they escalate further if not resolved?]

### What's Needed
- [Specific ask]
- **Deadline:** [When this needs resolution or update]

### Supporting Context
- [Links to tickets, threads, logs]
```

Realistic Walkthrough Example

User input:

```
/escalate API returning 500 errors intermittently — 3 Enterprise customers reported th
```

Claude's process:

1. **Understand the issue (from escalation skill):**
    - What's broken: API 500 errors
    - Who's affected: 3 Enterprise customers
    - Timeline: Started Tuesday, worsening
    - Why escalate: Multiple customers, Enterprise tier, growing pattern
2. **Assess impact (from escalation skill):**
    - Breadth: 3 customers, potentially more unreported
    - Depth: Blocking (API calls failing)
    - Duration: 3 days and growing
    - Revenue: 3 Enterprise accounts at risk
    - Severity: **High** (multiple customers, production-blocking)
3. **Determine target (from escalation skill):** Engineering (confirmed bug, needs code fix)
4. **Gather context:**
    - Search Intercom for tickets from all 3 customers
    - Pull CRM data for ARR and contacts
    - Search Slack for any internal discussion
    - Check Jira for related bugs
5. **Structure reproduction steps (from escalation skill):** Gather error patterns, affected endpoints, frequency
6. **Generate escalation brief**

Claude's output:

```
## ESCALATION: API 500 Errors Affecting Multiple Enterprise Customers

**Severity:** High
**Target team:** Engineering (Backend API)
**Reported by:** Support Team
**Date:** February 14, 2026

### Impact
- **Customers affected:** 3 Enterprise customers (Acme Corp, TechFlow,
  DataSync) — likely more unreported
- **Workflow impact:** API calls failing intermittently, blocking data
  syncs and integrations
- **Revenue at risk:** Combined ARR ~$350K across 3 known affected customers
```

~~~
- **Time in queue:** Started Tuesday Feb 11, reported Wednesday Feb 12,
  worsening since

### Issue Description
Multiple Enterprise customers are experiencing intermittent 500 errors when
calling our API. The errors started Tuesday morning and frequency has
increased over the past 3 days. Errors are not consistent — same endpoint
can succeed or fail for the same customer. No recent deployments or
configuration changes correlate with the start date. Customers are unable
to reliably sync data, which is blocking critical workflows.

### What's Been Tried
1. **Checked customer API keys** → All valid and within rate limits
2. **Reviewed recent deployments** → No API changes deployed on or around
   Feb 11
3. **Tested affected endpoints** → Successfully reproduced 500 errors on
   `/data/export` endpoint (~30% failure rate)
4. **Checked server logs** → Found "Database connection pool exhausted"
   errors correlating with 500 responses
5. **Temporary mitigation** → Advised customers to implement retry logic
   with exponential backoff (partial relief only)

### Reproduction Steps

**Environment:**
- Endpoint: `POST /api/v2/data/export`
- Authentication: OAuth 2.0 (Bearer token)
- Frequency: ~30% of requests fail with 500 error
- Pattern: No consistent trigger — same request payload succeeds sometimes,
  fails other times

**Steps to Reproduce:**
1. Authenticate via OAuth 2.0 to get bearer token
2. Make POST request to `/api/v2/data/export` with payload:
   ```json
   {
     "format": "json",
     "date_range": "last_30_days",
     "filters": { "status": "active" }
   }
~~~

3. Repeat request 10 times
4. **Expected:** All requests return 200 OK with export data
5. **Actual:** ~3 out of 10 requests return 500 Internal Server Error

Evidence:

- Server logs show "Database connection pool exhausted" at timestamp of failed requests

- Error rate increased from 10% on Feb 12 to 30% today (Feb 14)
- Affects multiple endpoints, but `/data/export` most impacted

Customer Communication

- **Acme Corp:** Last update Feb 13 — told them we're investigating, advised retry logic. They expect resolution by Friday Feb 14 (TODAY).
- **TechFlow:** Last update Feb 13 — same message. CTO said if not resolved by Monday they'll escalate to their exec team.
- **DataSync:** Last update Feb 14 morning — newer customer, very frustrated. Mentioned considering alternatives if we can't stabilize the API.

What's Needed

- **Immediate investigation:** Root cause analysis of database connection pool issue
- **Short-term fix:** Increase connection pool limits or optimize queries to reduce pool pressure
- **Long-term fix:** Architect for better connection pool management and error handling
- **Deadline:** Need resolution or mitigation by end of day Friday Feb 14 (Acme Corp's deadline). Need update for all 3 customers TODAY.

Supporting Context

- Intercom tickets: #5234 (Acme), #5241 (TechFlow), #5250 (DataSync)
- Slack thread: #engineering discussed database connection pool on Feb 13 (no resolution yet)
- Server logs: Attached in ticket #5234 (connection pool errors)
- No related Jira issues found — this is a new problem

Escalation Risk: High. If not resolved by Monday, TechFlow will escalate to exec level and may involve their account executive in contract discussions. DataSync mentioned evaluating alternatives.

```
#### Tips and Common Pitfalls

**Tips:**
- **Include multiple customer names** if it's a pattern — "3 customers reported" is mu
- **Quantify business impact** — ARR at risk, number of users blocked, timeline pressu
- **Document what you've tried** — engineering needs to know what to skip
- **Be specific about deadlines** — "ASAP" is vague; "by Friday because Acme's CTO exp
- **Include exact error messages and logs** — paste them into the escalation or attach

**Common Pitfalls:**
- **Vague impact statements** — "customers are having issues" vs. "3 Enterprise custom
- **Skipping reproduction steps** — even if you can't fully reproduce, document what y
- **Escalating too early** — check the escalation skill's "When to Escalate vs. Handle
- **Forgetting to follow up** — escalating doesn't mean you're off the hook; maintain

---

### Command 5: `/kb-article`

**File:** `/commands/kb-article.md`

#### Syntax
```

/kb-article <resolved issue, ticket reference, or topic description>

```
#### What It Does

Drafts a publish-ready knowledge base article from a resolved support issue, common qu

#### Required Inputs

- **Source material**: Description of the resolved issue, ticket ID, or topic to docum

#### Optional Inputs

- Article type preference (how-to, troubleshooting, FAQ, known issue, reference)
- Target audience (all users, admins, developers, specific plan level)

#### Expected Output
```

KB Article Draft

Title: [Article title] **Type:** [How-to / Troubleshooting / FAQ / Known Issue / Reference] **Category:** [Product area or topic] **Tags:** [Searchable tags] **Audience:** [All users / Admins / Developers / Specific plan]

[Full article content using appropriate template]

Publishing Notes

- **Source:** [Ticket #, customer conversation, or internal discussion]
- **Existing articles to update:** [If this overlaps with existing content]
- **Review needed from:** [SME or team if technical accuracy needs verification]
- **Suggested review date:** [When to revisit for accuracy]

```
#### Realistic Walkthrough Example

**User input:**
```

/kb-article How to configure webhook notifications — just resolved this for the third customer this month, clear there's a documentation gap

```
**Claude's process:**

1. **Understand the source (from knowledge-management skill):**
   - Topic: Webhook notifications configuration
   - Frequency: Common (3 times this month)
   - Purpose: Fill documentation gap, reduce future tickets
   - Article type: **How-to** (step-by-step guidance)
2. **Check for existing content (if KB connected):**
   - Search Guru for "webhook" articles
   - Find partial coverage but no complete setup guide
3. **Structure using how-to template (from knowledge-management skill):**
   - Prerequisites
   - Step-by-step instructions
   - Verification
   - Common issues
4. **Optimize for searchability (from knowledge-management skill):**
   - Title includes exact term customers search: "webhook notifications"
   - Opening sentence restates the task in plain language
   - Include error messages customers might encounter
5. **Generate article**

**Claude's output:**
```

KB Article Draft

Title: How to Configure Webhook Notifications **Type:** How-to **Category:** Integrations **Tags:** webhooks, notifications, API, integrations, events, callbacks **Audience:** Developers, Integration Admins

How to Configure Webhook Notifications

This guide shows you how to set up webhook notifications to receive real-time updates when events occur in your account (e.g., new data created, records updated, or exports completed).

Prerequisites

Before you begin, make sure you have:

- **Admin or Developer role** in your account (required to access API settings)
- **HTTPS endpoint URL** where you want to receive webhook payloads
- **Server capable of receiving POST requests** at that URL

Steps

1. Navigate to API Settings

1. Click **Settings** in the top navigation
2. Select **Integrations** from the left sidebar
3. Click **Webhooks** under the API section

You should see a page titled "Webhook Configuration" with a list of any existing webhooks.

2. Create a New Webhook

1. Click the **New Webhook** button in the top-right
2. Fill in the configuration form:
   - **Webhook Name:** Descriptive name (e.g., "Production Data Sync")
   - **Endpoint URL:** Your HTTPS URL (e.g., `https://yourapp.com/webhooks/data`)
   - **Events:** Select which events should trigger this webhook:
     - `data.created` — fired when new data is created
     - `data.updated` — fired when existing data is updated
     - `data.deleted` — fired when data is deleted
     - `export.completed` — fired when an export job finishes
   - **Secret Key** (optional but recommended): Used to verify webhook authenticity

3. Click **Save**

3. Verify the Webhook

After saving, your webhook is active immediately. To verify it's working:

1. Trigger one of the events you selected (e.g., create new data if you selected `data.created`)
2. Check your server logs for an incoming POST request from our platform
3. The request should include:
    - `X-Webhook-Signature` header (if you set a secret key)
    - JSON payload with event details

Example payload:

```
{
  "event": "data.created",
  "timestamp": "2026-02-14T10:30:00Z",
  "data": {
    "id": "rec_abc123",
    "type": "contact",
    "created_by": "user_xyz789"
  }
}
```

4. Verify Webhook Signature (Recommended)

If you set a secret key, verify the signature to confirm requests are from our platform:

```
import hmac
import hashlib

def verify_webhook(payload, signature, secret):
    expected = hmac.new(
        secret.encode(),
        payload.encode(),
        hashlib.sha256
    ).hexdigest()
    return hmac.compare_digest(signature, expected)
```

Common Issues

"Webhook endpoint unreachable" error

Cause: Your server isn't accessible or is rejecting the requests.

Solution:

- Verify your endpoint URL is correct and publicly accessible
- Ensure your server is running and responding to POST requests
- Check your firewall isn't blocking requests from our IP ranges
- Test your endpoint with curl: `curl -X POST https://yourapp.com/webhooks/data`

Webhooks are delayed by several minutes

Cause: We retry failed deliveries with exponential backoff.

Solution:

- Make sure your endpoint returns 200 OK quickly (< 3 seconds)
- Process webhook payloads asynchronously — acknowledge receipt immediately, then process in background
- Check your server logs for errors that might cause retries

Receiving duplicate events

Cause: Network issues can cause us to retry delivery.

Solution:

- Implement idempotency — use the event `id` field to deduplicate
- Example: Store processed event IDs and skip if already seen

Related Articles

- Webhook Event Reference — full list of events and payload schemas
- API Authentication — how to authenticate API requests
- Troubleshooting Webhooks — common webhook issues

Publishing Notes

Source: Resolved this for 3 customers in the past month (tickets #5100, #5201, #5234). Clear pattern of confusion about webhook setup process.

Existing articles to update:

- "API Overview" article should link to this as a related resource
- "Integration Getting Started" guide should mention webhooks as an option

Review needed from: Engineering team to verify signature verification code example is current (Python example may need updating for latest library versions)

Suggested review date: June 2026 (quarterly review) — webhook configuration UI may change

```
#### Tips and Common Pitfalls

**Tips:**
- **Write articles immediately after resolving repeat issues** — fresh in your mind, p
- **Use exact customer language** in titles — if customers say "webhooks aren't firing
- **Include code examples** when relevant — customers copy-paste, so make examples rea
- **Front-load the answer** — customers skim; put the key info in the first paragraph
- **Add related articles** — help customers discover adjacent content

**Common Pitfalls:**
- **Writing for yourself, not customers** — avoid internal jargon and assume less know
- **Skipping verification steps** — "to verify it worked..." sections are critical
- **Making articles too long** — if it's over 1,000 words, consider splitting into mul
- **Forgetting to tag and categorize** — articles customers can't find don't reduce ti
- **Not updating stale articles** — if the UI changed since the article was written, c

---

## Part III: Skills Deep Dive

Skills are packages of domain knowledge that Claude loads automatically when conversat

The customer-support plugin includes five skills covering the core competencies of sup

### Skill 1: `ticket-triage`

**File:** `/skills/ticket-triage/SKILL.md`

#### Trigger Conditions

This skill automatically loads when the conversation involves:
- Categorizing or classifying customer issues
- Assigning priority or severity to tickets
- Determining which team should handle an issue
- Deciding if something is a bug vs. feature request vs. how-to question
- Assessing SLA risk or urgency
- Checking for duplicate tickets or known issues

**Trigger phrases:**
- "How should I categorize this ticket?"
- "What priority is this?"
- "Which team should handle this?"
- "Is this a P1 or P2?"
- "Should this go to engineering or product?"

#### Knowledge & Capabilities Provided

**1. Category Taxonomy**

Nine primary categories with signal words for rapid classification:

| Category | Description | Signal Words |
├──────────┼──────────────┼─────────────┤
| Bug | Product behaving incorrectly | error, broken, crash, not working, failing |
| How-to | Needs guidance on using product | how do I, can I, where is, configure, hel
| Feature request | Wants non-existent capability | would be great if, wish I could, a
| Billing | Payment/subscription issues | charge, invoice, payment, refund, upgrade |
| Account | Access, permissions, user mgmt | login, password, access, permission, SSO,
| Integration | Third-party connections/APIs | API, webhook, integration, connect, OAu
| Security | Security, data, compliance concerns | data breach, unauthorized, complian
| Data | Data quality, migration, import/export | missing data, export, import, migrat
| Performance | Speed, reliability, availability | slow, timeout, latency, down, unava

**2. Priority Framework (P1–P4)**

Clear criteria for each priority level with SLA expectations:

- **P1 — Critical:** Production down, data loss/corruption, security breach, all/most
- **P2 — High:** Major feature broken, significant workflow blocked, many users affect
- **P3 — Medium:** Feature partially broken, workaround available, single user or smal
- **P4 — Low:** Minor inconvenience, cosmetic issue, general question, feature request

**Priority escalation triggers:**
- Customer waiting longer than SLA allows
- Multiple customers report same issue
- Customer explicitly escalates
- Workaround stops working
- Issue expands in scope

**3. Routing Rules**

Guidelines for which team/queue should handle each type:
```

```
| Route to | When |
├──────────┼──────┤
| Tier 1 (frontline) | How-to, known issues with docs, billing inquiries, password res
| Tier 2 (senior support) | Bugs needing investigation, complex config, integration tr
| Engineering | Confirmed bugs needing code fixes, infrastructure issues, performance 
| Product | Feature requests with demand, design decisions, workflow gaps |
| Security | Data access concerns, vulnerability reports, compliance questions |
| Billing/Finance | Refund requests, contract disputes, complex billing adjustments |

**4. Duplicate Detection Process**

Before routing, check:
1. Search by symptom (similar error messages, descriptions)
2. Search by customer (open tickets for same customer)
3. Search by product area (recent tickets in same feature)
4. Check known issues documentation

If duplicate found:
- Link to existing ticket
- Notify customer it's known and tracked
- Add new information to master ticket
- Bump priority if new report adds urgency

**5. Auto-Response Templates**

Pre-written initial responses for each category (Bug, How-to, Feature Request, Billing

#### How It Changes Claude's Behavior

**Without this skill:** Claude might suggest vague categorization or inconsistent prio

**With this skill:** Claude applies your organization's specific category taxonomy, fo

#### Inventory of Additional Resources

This skill contains only `SKILL.md` — no additional reference files, scripts, or asset

#### Practical Scenario

**Situation:** A support rep receives this message: "Our dashboard isn't loading — jus

**Without ticket-triage skill:** Rep might categorize as "Performance" and assign P3 (

**With ticket-triage skill:** Claude recognizes:
- Signal words: "not loading," "spinner" → likely **Bug** (not Performance)
- Urgency signal: "client demo in 2 hours" → time pressure
- Impact: Single user but high stakes (demo)
- Priority: **P2 — High** (major feature broken, time-sensitive, no workaround)
- Routing: **Tier 2 Support** (investigate if it's account-specific or broader, then e
  - Initial response: Use "Bug — Initial Response" template, acknowledge urgency, commit

The skill ensures consistent, appropriate triage regardless of which rep handles the t

---

### Skill 2: `customer-research`

**File:** `/skills/customer-research/SKILL.md`

#### Trigger Conditions

This skill automatically loads when:
- Answering a customer question that requires investigation
- Building background on a customer situation or account
- Synthesizing information from multiple sources
- Assessing confidence in an answer before sharing with customers
- Verifying facts before escalating or committing

**Trigger phrases:**
- "Does our product support...?"
- "What did we tell this customer before?"
- "Has anyone else asked about...?"
- "Is this documented anywhere?"
- "What's the history with this account?"

#### Knowledge & Capabilities Provided

**1. Multi-Source Research Methodology**

A systematic 5-step process:
1. **Understand the question** — factual, contextual, exploratory, or policy-based?
2. **Plan search strategy** — map question to likely source types
3. **Execute searches systematically** — tier-based prioritization
4. **Synthesize and validate** — combine findings, check for contradictions
5. **Present with attribution** — cite sources, assign confidence level

**2. Source Prioritization (5-Tier System)**

| Tier | Sources | Confidence | When to Use |
├──────┼─────────┼────────────┼─────────────┤
| **Tier 1** | Product docs, KB, policy docs, roadmap | High | Official, authoritative
| **Tier 2** | CRM records, support tickets, internal docs | Medium-High | Contextual,
| **Tier 3** | Chat, email, meeting notes | Medium | Informal, recent, may lack full c
| **Tier 4** | Web search, community forums, third-party docs | Low-Medium | General k
| **Tier 5** | Inferred from analogies or similar situations | Low | Use when direct s

**3. Confidence Scoring Framework**

- **High Confidence:** Confirmed by official docs, multiple sources corroborate, infor
- **Medium Confidence:** Found in informal sources, single source without corroboratio
- **Low Confidence:** Inferred from related info, sources outdated or unreliable, cont
- **Unable to Determine:** No relevant information found, requires specialized knowled

**4. Answer Synthesis Structure**

Template for presenting research findings:
- Direct answer (bottom-line first)
- Confidence level with rationale
- Supporting evidence by source
- Caveats and nuance
- Recommendation (ready to share with customer? needs verification?)

**5. Escalation Guidelines**

When to answer directly vs. escalate to SME:
- **Answer directly:** Official docs address it, multiple reliable sources corroborate
- **Escalate/verify:** Involves roadmap commitments, pricing/legal/contract questions,

**6. Knowledge Documentation Process**

After research, capture findings for team benefit:
- When to document (common questions, took significant effort, nuanced answers)
- Documentation format (question, answer, sources, review notes)
- KB hygiene (date-stamp, flag version-specific info, quarterly review)

#### How It Changes Claude's Behavior

**Without this skill:** Claude might search randomly, miss authoritative sources, or p

**With this skill:** Claude follows a systematic tier-based search strategy, cross-ref

#### Inventory of Additional Resources

This skill contains only `SKILL.md` — no additional reference files. All methodology i

#### Practical Scenario

**Situation:** Customer asks: "Does your platform support exporting data to Google She

**Without customer-research skill:** Claude might do a quick web search, find a third-

**With customer-research skill:** Claude:
1. **Understands the question:** Product capability question, definitive answer needed
2. **Plans strategy:** Tier 1 (docs, KB) → Tier 2 (support tickets) → Tier 4 (web searc
3. **Executes systematically:**
   - Search Guru KB → finds article "Supported Export Formats" (lists CSV, JSON, XML —
   - Search Notion docs → finds "Integrations Roadmap" (Google Sheets on Q3 2026 roadma
   - Search Intercom → finds 2 tickets where customers asked, told "not yet supported,
   - Web search → finds Zapier integration that can automate CSV export → Google Sheets
```

```
4. **Synthesizes:**
   - Direct answer: "Not natively, but possible via third-party automation (Zapier)"
   - Confidence: **High** (official docs confirm no native support, roadmap shows futu
   - Sources: Guru KB "Supported Export Formats," Notion "Integrations Roadmap," Inter
5. **Presents:**
   - Ready to share with customer
   - Suggests documenting as FAQ since 2 customers asked

Result: Accurate answer with workaround, customer knows it's on roadmap, no false expe

---

### Skill 3: `response-drafting`

**File:** `/skills/response-drafting/SKILL.md`

#### Trigger Conditions

This skill automatically loads when:
- Drafting customer-facing communications (emails, tickets, chat)
- Responding to escalations or sensitive situations
- Delivering good news, bad news, or status updates
- Adapting tone for different stakeholders or situations
- Following up after silence or delays

**Trigger phrases:**
- "Draft a response to..."
- "How should I tell the customer...?"
- "What's the best way to say...?"
- "Help me respond to this escalation"
- "I need to deliver bad news about..."

#### Knowledge & Capabilities Provided

**1. Core Communication Principles**

Seven foundational rules for all customer communication:
1. **Lead with empathy** — acknowledge their situation first
2. **Be direct** — bottom-line-up-front, customers are busy
3. **Be honest** — never overpromise, mislead, or hide bad news in jargon
4. **Be specific** — concrete details, timelines, names (not vague language)
5. **Own it** — "we" not "the system" when taking responsibility
6. **Close the loop** — every response has clear next steps
7. **Match their energy** — frustrated → empathetic; excited → enthusiastic

**2. Universal Response Structure**

Four-part template for most communications:
1. **Acknowledgment** (1-2 sentences) — show you understand their situation
2. **Core Message** (1-3 paragraphs) — deliver the information, answer, or update
```

```
3. **Next Steps** (1-3 bullets) — what you'll do, what they should do, when they'll he
4. **Closing** (1 sentence) — warm but professional sign-off

**3. Tone Spectrum by Situation**

| Situation | Tone | Characteristics |
├───────────┼──────┼─────────────────┤
| Good news / wins | Celebratory | Enthusiastic, warm, congratulatory, forward-looking
| Routine update | Professional | Clear, concise, informative, friendly |
| Technical response | Precise | Accurate, detailed, structured, patient |
| Delayed delivery | Accountable | Honest, apologetic, action-oriented, specific |
| Bad news | Candid | Direct, empathetic, solution-oriented, respectful |
| Issue / outage | Urgent | Immediate, transparent, actionable, reassuring |
| Escalation | Executive | Composed, ownership-taking, plan-presenting, confident |
| Billing / account | Precise | Clear, factual, empathetic, resolution-focused |

**4. Length Guidelines by Channel**

- **Chat/IM:** 1-4 sentences, immediate point
- **Support ticket:** 1-3 short paragraphs, structured and scannable
- **Email:** 3-5 paragraphs max, respect their inbox
- **Escalation response:** As long as needed but well-structured with headers
- **Executive communication:** 2-3 paragraphs max, data-driven

**5. Response Templates for Common Scenarios**

Pre-written templates for:
- Acknowledging a bug report
- Responding to billing/account issues
- Declining a feature request
- Outage/incident communication
- Following up after silence

Each template includes tone guidance, structure, and what to verify before sending.

**6. Personalization by Relationship Stage**

- **New customer (0-3 months):** More formal, extra context, proactively offer help
- **Established customer (3+ months):** Warm, collaborative, can reference history
- **Frustrated/escalated:** Extra empathy, urgency, concrete action plans, shorter fee

**7. Follow-up and Escalation Guidance**

When to follow up, how often, and when to escalate to manager, product, engineering, o

#### How It Changes Claude's Behavior

**Without this skill:** Claude might draft generic, overly formal responses that don't

**With this skill:** Claude applies situation-specific tone guidelines, uses proven te
```

```
#### Inventory of Additional Resources

This skill contains only `SKILL.md` — no additional reference files. All templates and

#### Practical Scenario

**Situation:** Customer (established, 18-month relationship) reports a bug that's bloc

**Without response-drafting skill:** Claude might draft a formal, generic acknowledgme

**With response-drafting skill:** Claude:
1. **Identifies situation type:** Bug report, established customer, moderate urgency
2. **Selects tone:** Professional with warmth (established relationship), empathetic (
3. **Chooses template:** "Acknowledging a Bug Report"
4. **Personalizes:**
   - References customer by name
   - Acknowledges the specific impact on their workflow
   - Uses warm, collaborative tone befitting 18-month relationship
   - Skips introductory explanations they already know
5. **Structures response:**
   - Acknowledgment: "Thanks for flagging this — I can see how this would disrupt your
   - Core message: What the bug is, what's being done, workaround if available
   - Next steps: Timeline for investigation, when they'll get an update
   - Closing: "Let me know if you have questions or if this impacts you in other ways"
6. **Includes internal notes:** Verify the bug is actually reproducible, check if work

Result: Empathetic, appropriately toned response that fits the relationship, with clea

---

### Skill 4: `escalation`

**File:** `/skills/escalation/SKILL.md`

#### Trigger Conditions

This skill automatically loads when:
- Packaging an issue for engineering, product, or leadership
- Determining whether an issue warrants escalation
- Structuring reproduction steps for a bug report
- Assessing business impact of an issue
- Following up on escalated issues

**Trigger phrases:**
- "Should I escalate this?"
- "How do I escalate to engineering?"
- "Package this for the product team"
- "Write an escalation brief"
- "What's the business impact of this?"
```

```
#### Knowledge & Capabilities Provided

**1. When to Escalate vs. Handle in Support**

Clear decision criteria:

**Handle in Support:**
- Documented solution or known workaround exists
- Configuration or setup issue you can resolve
- Customer needs guidance or training, not a fix
- Known limitation with documented alternative

**Escalate When:**
- **Technical:** Confirmed bug needs code fix, infrastructure investigation, data corr
- **Complexity:** Beyond support's ability to diagnose, requires access support doesn'
- **Impact:** Multiple customers affected, production down, data at risk, security con
- **Business:** High-value customer at risk, SLA breach, customer requesting exec invo
- **Time:** Open beyond SLA, waiting unreasonably long, normal channels not progressin
- **Pattern:** Same issue from 3+ customers, recurring "fixed" issue, increasing sever

**2. Escalation Tiers**

Five escalation paths with clear triggers:

| From | To | When |
├──────┼─────┼──────┤
| L1 → L2 | Frontline → Senior Support | Deeper investigation needed, specialized know
| L2 → Engineering | Senior Support → Engineering | Confirmed bug, infrastructure issue
| L2 → Product | Senior Support → Product Mgmt | Feature gap, design decision, workflo
| Any → Security | Any Tier → Security Team | Data exposure, unauthorized access, vuln
| Any → Leadership | Any Tier → Execs | High-revenue churn risk, SLA breach on critica

**3. Structured Escalation Format**

Template ensuring all necessary context is included:
- One-line summary
- Severity (Critical / High / Medium)
- Target team
- Impact (breadth, depth, duration, revenue, time pressure)
- Issue description (3-5 sentences)
- What's been tried (actions → results)
- Reproduction steps (for bugs)
- Customer communication (last update, expectations, escalation risk)
- What's needed (specific ask, deadline)
- Supporting context (links, logs, threads)

**4. Business Impact Assessment**

Six dimensions to quantify:
```

```
- **Breadth:** How many customers/users? Growing?
- **Depth:** Blocked vs. inconvenienced?
- **Duration:** How long has this been going on?
- **Revenue:** ARR at risk? Pending deals affected?
- **Reputation:** Could this become public? Reference customer?
- **Contractual:** SLA breach? Contractual obligations?

**Severity shorthand:**
- **Critical:** Production down, data at risk, security breach, multiple high-value cu
- **High:** Major functionality broken, key customer blocked, SLA at risk
- **Medium:** Significant issue with workaround, important but not urgent

**5. Reproduction Steps Best Practices**

Guidelines for writing clear reproduction steps:
1. Start from clean state (account type, config, permissions)
2. Be specific ("Click Export in top-right of Dashboard" not "try to export")
3. Include exact values (specific inputs, dates, IDs)
4. Note environment (browser, OS, account type, feature flags)
5. Capture frequency (always? intermittent? specific conditions?)
6. Include evidence (screenshots, exact error messages, logs)
7. Note what you've ruled out ("tested Chrome and Firefox — same behavior")

**6. Follow-up Cadence**

Post-escalation follow-up timing:

| Severity | Internal Check-in | Customer Update |
├──────────┼───────────────────┼─────────────────┤
| Critical | Every 2 hours | Every 2-4 hours |
| High | Every 4 hours | Every 4-8 hours |
| Medium | Daily | Every 1-2 business days |

**7. De-escalation Criteria**

When to de-escalate:
- Root cause found and it's support-resolvable
- Workaround found that unblocks customer
- Issue resolves itself (but document root cause)
- New information changes severity

#### How It Changes Claude's Behavior

**Without this skill:** Claude might write vague escalations missing key context, or e

**With this skill:** Claude applies clear escalation criteria, structures briefs with

#### Inventory of Additional Resources

This skill contains only `SKILL.md` — no additional reference files. All escalation fr
```

```
#### Practical Scenario

**Situation:** A support rep has been working on an API issue for 2 days. One customer

**Without escalation skill:** Rep might wait longer (not realizing 3 customers = patte

**With escalation skill:** Claude recognizes:
1. **Escalation trigger met:** "Same issue from 3+ customers" = pattern escalation
2. **Appropriate tier:** L2 → Engineering (bug affecting multiple customers)
3. **Severity:** High (multiple customers, production-blocking)
4. **Required context:**
   - Impact: 3 customers named, ARR quantified
   - Timeline: Started 2 days ago
   - What's been tried: Detailed troubleshooting steps
   - Reproduction: Specific API endpoint, error rate, logs
   - Customer communication: What they've been told, deadlines
   - Business impact: Revenue at risk, escalation risk
5. **Follow-up plan:** Check with engineering every 4 hours, update customers every 4-

Result: Well-structured escalation that engineering can act on immediately, with appro

---

### Skill 5: `knowledge-management`

**File:** `/skills/knowledge-management/SKILL.md`

#### Trigger Conditions

This skill automatically loads when:
- Creating knowledge base articles from resolved tickets
- Updating existing KB content
- Organizing or categorizing KB articles
- Writing how-to guides, troubleshooting docs, or FAQs
- Maintaining KB hygiene and accuracy

**Trigger phrases:**
- "Write a KB article about..."
- "Document this for future reference"
- "How should I structure this guide?"
- "This should be self-service"
- "Update the article on..."

#### Knowledge & Capabilities Provided

**1. Article Structure and Formatting Standards**

Universal elements every article needs:
- **Title:** Clear, searchable, describes outcome/problem
- **Overview:** 1-2 sentences explaining coverage and audience
- **Body:** Structured content appropriate to article type
- **Related articles:** Links to companion content
- **Metadata:** Category, tags, audience, last updated date

Formatting rules:
- Headers (H2, H3) for scannability
- Numbered lists for sequential steps
- Bullet lists for non-sequential items
- Bold for UI elements and key terms
- Code blocks for commands, errors, configurations
- Tables for comparisons and reference data
- Callouts for warnings, tips, caveats
- Short paragraphs (2-4 sentences max)

**2. Writing for Searchability**

Title best practices:
- "How to configure SSO with Okta" (specific, includes tool name)
- "Fix: Dashboard shows blank page" (includes symptom)
- "API rate limits and quotas" (includes terms customers search for)
- "Error: 'Connection refused' when importing data" (exact error message)

Keyword optimization:
- Include exact error messages (customers copy-paste into search)
- Use customer language, not internal terminology
- Include common synonyms (delete/remove, dashboard/home page)
- Add alternate phrasings
- Tag with product areas matching customer mental models

Opening sentence formula:
- **How-to:** "This guide shows you how to [accomplish X]."
- **Troubleshooting:** "If you're seeing [symptom], this article explains how to fix i
- **FAQ:** "[Question in customer's words]? Here's the answer."
- **Known issue:** "Some users are experiencing [symptom]. Here's what we know and how

**3. Common Article Types**

Five article types with templates:

**How-to Articles:**
- Structure: Overview → Prerequisites → Steps → Verify It Worked → Common Issues → Rela
- Best practices: Start steps with verbs, include specific paths, show what users shou

**Troubleshooting Articles:**
- Structure: Symptoms → Cause → Solution (multiple options) → Prevention → Still Having
- Best practices: Lead with symptoms, provide multiple solutions (most likely first),

**FAQ Articles:**
- Structure: Question → Direct Answer → Details (if needed) → Related Questions
- Best practices: Answer in first sentence, keep concise, link related FAQs

**Known Issue Articles:**
- Structure: Status → Symptoms → Workaround → Fix Timeline → Updates
- Best practices: Keep status current, update when fixed, mark resolved but keep live

**Reference Articles:**
- Structure: Overview → Reference tables/lists → Examples → Notes
- Best practices: Optimized for quick lookup, comprehensive coverage

**4. Review and Maintenance Cadence**

| Activity | Frequency | Who |
├──────────┼───────────┼─────┤
| New article review | Before publishing | Peer review + SME |
| Accuracy audit | Quarterly | Support team (top-traffic articles) |
| Stale content check | Monthly | Flag articles not updated in 6+ months |
| Known issue updates | Weekly | Update all open known issues |
| Analytics review | Monthly | Check low helpfulness ratings, high bounce |
| Gap analysis | Quarterly | Identify top ticket topics without KB articles |

**Article lifecycle:** Draft → Published → Needs Update → Archived → Retired

**5. When to Update vs. Create New**

**Update existing:**
- Product changed, steps need refreshing
- Article mostly right but missing detail
- Feedback indicates confusion in specific section
- Better workaround/solution found

**Create new:**
- New feature/product area needs documentation
- Resolved ticket reveals gap (no article exists)
- Existing article covers too many topics (should split)
- Different audience needs same info explained differently

**6. Linking and Categorization Taxonomy**

Suggested category structure:
- Getting Started (account setup, first-time config, quick starts)
- Features & How-tos (by feature area)
- Integrations (by integration, plus API reference)
- Troubleshooting (common errors, performance, known issues)
- Billing & Account (plans, billing, account management)

Linking best practices:
- Troubleshooting → how-to ("For setup instructions, see...")
- How-to → troubleshooting ("If you encounter errors, see...")
- FAQ → detailed articles ("For full walkthrough, see...")
- Use relative links (survive restructuring better)
- Avoid circular links

#### How It Changes Claude's Behavior

**Without this skill:** Claude might write unstructured articles, miss searchability o

**With this skill:** Claude applies proven article templates, optimizes titles and key

#### Inventory of Additional Resources

This skill contains only `SKILL.md` — no additional reference files. All templates and

#### Practical Scenario

**Situation:** A support rep resolved a ticket about webhook configuration for the thi

**Without knowledge-management skill:** Rep might write a loose explanation in whateve

**With knowledge-management skill:** Claude:
1. **Identifies article type:** How-to (step-by-step instructions)
2. **Optimizes title:** "How to Configure Webhook Notifications" (includes exact searc
3. **Structures using how-to template:**
   - Overview with plain-language opening sentence
   - Prerequisites (what's needed before starting)
   - Numbered steps with specific UI paths
   - Verification section (how to confirm it worked)
   - Common Issues section (based on the 3 tickets)
   - Related articles (API auth, webhook troubleshooting)
4. **Optimizes for search:**
   - Tags: webhooks, notifications, API, integrations, events
   - Includes exact error messages customers might search for
   - Uses customer language ("webhook notifications" not "event callbacks")
5. **Adds metadata:**
   - Category: Integrations
   - Audience: Developers, Integration Admins
   - Source: Tickets #5100, #5201, #5234
   - Review date: June 2026 (quarterly)
6. **Suggests related updates:**
   - Link from "API Overview" article
   - Mention in "Integration Getting Started" guide

Result: Publish-ready article that's findable, scannable, and complete — reduces futur

---

## Part IV: Connectors & MCP Servers

The customer-support plugin pre-configures seven MCP (Model Context Protocol) servers
```

```
### Full Inventory of MCP Servers

**Source:** `.mcp.json` at plugin root

| Server Name | Type | URL | Service | Purpose |
├─────────────┼──────┼─────┼─────────┼─────────┤
| `slack` | HTTP | `https://mcp.slack.com/mcp` | Slack | Team chat, customer channels,
| `intercom` | HTTP | `https://mcp.intercom.com/mcp` | Intercom | Support platform (ti
| `hubspot` | HTTP | `https://mcp.hubspot.com/anthropic` | HubSpot | CRM (account deta
| `guru` | HTTP | `https://mcp.api.getguru.com/mcp` | Guru | Knowledge base (internal 
| `atlassian` | HTTP | `https://mcp.atlassian.com/v1/mcp` | Atlassian | Project tracke
| `notion` | HTTP | `https://mcp.notion.com/mcp` | Notion | Knowledge base (docs, wiki
| `ms365` | HTTP | `https://microsoft365.mcp.claude.com/mcp` | Microsoft 365 | Email (

### Required vs. Optional Connectors

**None of the connectors are strictly required** — the plugin functions without any of

| Connector | Impact on Plugin Value |
├───────────┼────────────────────────┤
| **Support platform** (Intercom, Zendesk, Freshdesk) | **High** — Provides ticket his
| **Knowledge base** (Guru, Notion, Confluence) | **High** — Enables `/research` to fi
| **CRM** (HubSpot, Salesforce) | **Medium-High** — Provides account context (ARR, pla
| **Project tracker** (Atlassian, Linear, Asana) | **Medium** — Lets `/escalate` check
| **Chat** (Slack, Teams) | **Medium** — Enables `/research` to search internal discus
| **Email** (MS365, Google Workspace) | **Low-Medium** — Supports `/research` for cust

**Recommended minimum setup:** Support platform + Knowledge base. This covers 80% of t

### Setup Instructions

#### General MCP Server Setup Flow

All seven pre-configured servers use **HTTP** type, which means they're cloud-hosted s

1. **Enable the server** in your Claude settings (Desktop app: Settings → MCP Servers)
2. **Authenticate** when prompted (usually OAuth — "Sign in with [Service]")
3. **Grant permissions** (read access to your workspace/account)
4. **Verify connection** (Claude will confirm when authentication succeeds)

#### Service-Specific Setup

**Slack** (`slack`)
- **Authentication:** OAuth (Sign in with Slack)
- **Permissions needed:** Read public channels, read private channels (if you want to 
- **Configuration:** After auth, select which workspaces Claude can access
- **What Claude can do:** Search messages, read channel history, post messages (if you
- **Common issue:** If you see "workspace not found," you may need admin approval for 

**Intercom** (`intercom`)
```

```
- **Authentication:** OAuth (Sign in with Intercom)
- **Permissions needed:** Read conversations, read contacts, read help center articles
- **Configuration:** After auth, select which workspace Claude can access
- **What Claude can do:** Search tickets, read conversation history, read help center 
- **Common issue:** Requires Admin or Support role in Intercom

**HubSpot** (`hubspot`)
- **Authentication:** OAuth (Sign in with HubSpot)
- **Permissions needed:** Read CRM data (contacts, companies, deals), read activity hi
- **Configuration:** After auth, select which HubSpot portal Claude can access
- **What Claude can do:** Search contacts and companies, read deal information, read a
- **Common issue:** Requires Sales or Service Hub user license

**Guru** (`guru`)
- **Authentication:** API token (generate in Guru settings)
- **Permissions needed:** Read cards, read collections
- **Configuration:**
  1. In Guru, go to Settings → API → Generate token
  2. In Claude, paste token when prompted
  3. Select which collections Claude can access
- **What Claude can do:** Search knowledge cards, read verified content, check verific
- **Common issue:** Token expires after 1 year — you'll need to regenerate

**Atlassian** (`atlassian`)
- **Authentication:** OAuth (Sign in with Atlassian)
- **Permissions needed:** Read Jira issues, read Confluence pages
- **Configuration:** After auth, select which sites (Jira instances, Confluence spaces
- **What Claude can do:** Search Jira issues, read issue details, search Confluence pa
- **Common issue:** Requires project access in Jira and space access in Confluence

**Notion** (`notion`)
- **Authentication:** OAuth (Sign in with Notion)
- **Permissions needed:** Read pages, read databases
- **Configuration:** After auth, select which pages/databases Claude can access
- **What Claude can do:** Search pages, read page content, query databases
- **Common issue:** Only surfaces pages/databases you've explicitly shared with the in

**Microsoft 365** (`ms365`)
- **Authentication:** OAuth (Sign in with Microsoft)
- **Permissions needed:** Read email (Outlook), read files (OneDrive/SharePoint)
- **Configuration:** After auth, confirm which M365 services Claude can access
- **What Claude can do:** Search emails, read email threads, search OneDrive/SharePoin
- **Common issue:** May require admin consent if your org restricts third-party app ac

### The `~~placeholder` System Explained

The `~~placeholder` system is what makes the customer-support plugin **tool-agnostic**

#### How It Works
```

~~~
In command and skill files, you'll see references like:
- `~~support platform`
- `~~knowledge base`
- `~~CRM`
- `~~chat`
- `~~project tracker`

These are **category placeholders** — generic references to a type of tool rather than

**Example from `/commands/triage.md`:**
```markdown
Before routing, check available sources:

- **~~support platform**: Search for similar open or recently resolved tickets
- **~~knowledge base**: Check for known issues or existing documentation
- **~~project tracker**: Check if there's an existing bug report or feature request
~~~

When Claude executes this command, it interprets:

- `~~support platform` → whichever support tool you've connected (Intercom, Zendesk, Freshdesk, etc.)
- `~~knowledge base` → whichever knowledge base you've connected (Guru, Notion, Confluence, etc.)
- `~~project tracker` → whichever tracker you've connected (Atlassian, Linear, Asana, etc.)

Why This Matters

Without placeholders: The plugin would hardcode "Search Intercom" and "Check Jira," making it unusable for orgs using Zendesk and Linear.

With placeholders: The plugin describes the workflow in terms of categories ("search the support platform," "check the project tracker"), and the actual tools are swapped in based on what each org connects.

This is what allows a single plugin to work across thousands of orgs with different tech stacks.

CONNECTORS.md Contents

The `CONNECTORS.md` file at the plugin root documents all placeholder categories and their mappings.

Full contents of `/CONNECTORS.md`:

CATEGORY	PLACEHOLDER	INCLUDED SERVERS	OTHER OPTIONS
Chat	`~~chat`	Slack	Microsoft Teams
Email	`~~email`	Microsoft 365	Google Workspace
Cloud storage	`~~cloud storage`	Microsoft 365	Google Drive, Dropbox
Support platform	`~~support platform`	Intercom	Zendesk, Freshdesk, HubSpot Service Hub
CRM	`~~CRM`	HubSpot	Salesforce, Pipedrive
Knowledge base	`~~knowledge base`	Guru, Notion	Confluence, Help Scout
Project tracker	`~~project tracker`	Atlassian (Jira/Confluence)	Linear, Asana

"Included servers" are pre-configured in `.mcp.json` — they work out of the box once you authenticate.

"Other options" are alternatives you can use but must configure manually (see next section).

How to Add New MCP Servers

If your organization uses a tool not pre-configured (e.g., Zendesk instead of Intercom), you can add it.

Step 1: Find the MCP Server

Check if the tool has an MCP server:

1. **Search the MCP registry:** [mcp.registry URL if known, or "search for '[tool name] MCP server'"]
2. **Check the vendor's docs:** Many SaaS companies publish MCP endpoints
3. **Ask in Claude settings:** The MCP server browser may list additional options

Step 2: Add to `.mcp.json`

Edit the plugin's `.mcp.json` file (or create a custom `.mcp.json` in your project directory):

```
{
  "mcpServers": {
    "zendesk": {
      "type": "http",
      "url": "https://mcp.zendesk.com/mcp"
    }
  }
}
```

Step 3: Authenticate

Restart Claude (if editing plugin directly) or reload the project. The new server will appear in MCP settings. Authenticate as described in the setup instructions above.

Step 4: Verify Connection

Test that the connection works:

```
/research [topic relevant to this tool]
```

Claude should search the newly connected tool and cite it in sources.

Step 5: Update Placeholder Mappings (Optional)

If you want to replace a pre-configured tool completely (e.g., use Zendesk instead of Intercom as your `~~support platform`), you may need to update the plugin's internal mappings. This is an advanced customization — see Part V: Customization & Extension for details.

Part V: Customization & Extension

The customer-support plugin is designed to be customized for your organization's specific tools, processes, terminology, and support workflows. There are three primary approaches to customization, each with different trade-offs.

Three Approaches Overview

APPROACH	EFFORT	MAINTAINABILITY	RISK	BEST FOR
A: Direct Modification	Low	Low (updates overwrite)	Medium	Quick experiments, personal tweaks
B: Fork the Plugin	Medium	Medium (manual merge)	Low	Team-wide customizations, significant changes
C: CLAUDE.md + Complementary Skills	Low	High (plugin untouched)	Low	Lightweight additions, company-specific knowledge

Approach A: Direct Plugin Modification

What it is: Edit the plugin's files in place — add content to existing skills, modify commands, change MCP configuration.

When to use:

- Quick personal experiments ("what if I added this to the triage taxonomy?")
- Testing an idea before committing to a full fork
- Small additions that don't fundamentally change the plugin

Risks:

- Plugin updates may overwrite your changes (if you reinstall or upgrade)
- No clean separation between original and customized content
- Hard to track what you changed vs. what came with the plugin

Example: Adding a New Issue Category

Your team frequently sees tickets about "mobile app crashes" but the current category taxonomy doesn't have a mobile-specific category. You want to add it.

Step 1: Locate the file

Open `/skills/ticket-triage/SKILL.md`

Step 2: Add the category

Find the "Category Taxonomy" section and add:

```
| Category | Description | Signal Words |
|----------|-------------|--------------|
| **Bug** | Product is behaving incorrectly or unexpectedly | Error, broken, crash, no
| **Mobile** | Mobile app-specific issues (iOS, Android) | Mobile, app crashes, iPhone
| **How-to** | Customer needs guidance on using the product | How do I, can I, where i
...
```

Step 3: Update routing rules

Scroll to the "Routing Rules" section and add:

```
| Route to | When |
|----------|------|
| **Tier 1 (frontline support)** | How-to questions, known issues with documented solu
| **Mobile Engineering** | Mobile app crashes, platform-specific bugs (iOS/Android), a
| **Tier 2 (senior support)** | Bugs requiring investigation, complex configuration, i
...
```

Step 4: Test

```
/triage Customer's iPhone app keeps crashing when they try to upload photos
```

Claude should now categorize this as "Mobile" and route to "Mobile Engineering."

Step 5: Document your change

Add a comment at the top of the file:

```
<!-- CUSTOMIZED: Added "Mobile" category for app-specific issues (Feb 2026) -->
```

This reminds you (and teammates) that the file has been modified when updates come out.

Example: Adding Company-Specific Response Template

You want to add a response template for a common situation at your company: customers asking about SOC 2 compliance.

Step 1: Open `/skills/response-drafting/SKILL.md`

Step 2: Add to the "Response Templates for Common Scenarios" section:

```
### Responding to Compliance/Security Questions (SOC 2, GDPR, etc.)

Hi [Name],

Thank you for asking about our [SOC 2 / GDPR / HIPAA] compliance.

[If SOC 2:]
We are SOC 2 Type II certified. Our most recent audit report is from
[date] and covers [scope]. I can share the report directly with you —
please confirm the best email address to send it to, and I'll get it
over within 24 hours.

[If GDPR:]
We are fully GDPR compliant. Here's a summary of our data practices:
- [Key point 1]
- [Key point 2]
- [Link to privacy policy]

If you need a Data Processing Agreement (DPA), I can send over our
standard DPA for your legal team's review.

[If other compliance:]
Let me connect you with our security team who can provide detailed
information about our [compliance standard] practices. I'll intro you
via email within 24 hours.

Please let me know if you have specific questions or requirements, and
I'm happy to coordinate with our security and legal teams.

Best,
[Your name]
```

Step 3: Test

```
/draft-response Customer (Acme Corp) is asking if we're SOC 2 compliant — they need th
```

Claude should now use this template and fill in the specifics.

Approach B: Fork the Plugin (Recommended for Teams)

What it is: Copy the entire plugin directory, rename it, and modify the copy. Install the fork alongside or instead of the original.

When to use:

- Team-wide customizations everyone should share
- Significant modifications that change plugin behavior
- You need to maintain changes across plugin updates
- You want clean separation between original and custom content

Advantages:

- Your changes are preserved in a separate plugin
- You can track changes in version control (git)
- Easy to share with teammates (package the fork as `.plugin` file)
- Can pull updates from original plugin and merge selectively

Maintenance burden: You must manually track and merge upstream changes when the original plugin updates.

Step-by-Step: Forking the Plugin

Step 1: Copy the plugin directory

```
cd ~/claude-plugins  # or wherever plugins are installed
cp -r customer-support customer-support-acme
```

Step 2: Update `plugin.json`

Edit `customer-support-acme/.claude-plugin/plugin.json`:

```
{
  "name": "customer-support-acme",
  "version": "1.0.0",
  "description": "Customer support plugin customized for Acme Corp workflows, tools, a
  "author": {
    "name": "Acme Corp Support Team"
  }
}
```

Step 3: Make your modifications

Now edit skills, commands, and MCP configuration freely. Examples below.

Step 4: Install the forked plugin

If you're working in a project directory:

```
mkdir -p .claude/plugins
cp -r customer-support-acme .claude/plugins/
```

Or package it for distribution:

```
cd customer-support-acme
zip -r customer-support-acme.plugin . -x "*.DS_Store"
```

Share `customer-support-acme.plugin` with teammates. They can install it via:

```
claude plugins add /path/to/customer-support-acme.plugin
```

Example: Adding Industry-Specific Knowledge (Healthcare Support)

Your company provides healthcare software, and support tickets often involve HIPAA, HL7 integrations, clinical workflows, and EHR systems. You want to add healthcare-specific context to the plugin.

Modification 1: Expand ticket categories

Edit `/skills/ticket-triage/SKILL.md` to add healthcare-specific categories:

```
| Category | Description | Signal Words |
├──────────┼─────────────┼──────────────┤
| **Integration (Healthcare)** | HL7, FHIR, EHR integrations | HL7, FHIR, EHR, Epic, C
| **Compliance** | HIPAA, PHI, audit logs, data privacy | HIPAA, PHI, BAA, audit log, 
| **Clinical Workflow** | Issues specific to clinical use cases | patient chart, clini
```

Modification 2: Add healthcare response templates

Edit `/skills/response-drafting/SKILL.md` to add:

```
### Responding to HIPAA/PHI Questions

Hi [Name],

Thank you for raising this. We take HIPAA compliance and PHI protection
very seriously.

[Describe how the platform handles PHI in this specific scenario]

Here are the relevant safeguards in place:
- All PHI is encrypted at rest (AES-256) and in transit (TLS 1.2+)
- Access logs are maintained for all PHI access (available for audit)
- Role-based access controls (RBAC) ensure only authorized users see PHI
- We have a signed Business Associate Agreement (BAA) with your organization

If you need additional documentation (our HIPAA compliance summary, BAA,
SOC 2 report, etc.), I'm happy to provide those. Let me know what would
be most helpful.

Best,
[Your name]
```

Modification 3: Add healthcare-specific escalation tier

Edit `/skills/escalation/SKILL.md` to add:

```
| From | To | When |
├───────┼──────┼───────┤
| Any → Compliance/Security | Any Tier → Compliance Team | HIPAA breach, PHI exposure,
| L2 → Integration Engineering | Senior Support → Integration Team | HL7/FHIR issues,
```

Modification 4: Update MCP servers for healthcare tools

Edit `.mcp.json` to add:

```
{
  "mcpServers": {
    "slack": { "type": "http", "url": "https://mcp.slack.com/mcp" },
    "intercom": { "type": "http", "url": "https://mcp.intercom.com/mcp" },
    "hubspot": { "type": "http", "url": "https://mcp.hubspot.com/anthropic" },
    "jira-healthcare": {
      "type": "http",
      "url": "https://mcp.atlassian.com/v1/mcp",
      "project": "HEALTHCARE"
    },
    "confluence-compliance": {
      "type": "http",
      "url": "https://mcp.atlassian.com/v1/mcp",
      "space": "COMPLIANCE"
    }
  }
}
```

Step 5: Document your fork

Update the `README.md` in your forked plugin:

```
# Customer Support Plugin — Acme Healthcare Edition

This is a fork of the official Anthropic customer-support plugin (v1.0.0),
customized for Acme Corp's healthcare support workflows.

## Customizations

- Added healthcare-specific ticket categories (Integration-Healthcare, Compliance, Cli
- Added HIPAA/PHI response templates
- Added Compliance/Security and Integration Engineering escalation tiers
- Connected to Acme's HEALTHCARE Jira project and COMPLIANCE Confluence space

## Maintenance

- **Base plugin version:** customer-support v1.0.0
- **Last upstream merge:** February 14, 2026
- **Customization owner:** Acme Support Team (support-team@acme.com)

When the upstream plugin updates, review changes and merge selectively to preserve cus
```

Example: Adding Company-Specific Processes (Acme Corp)

You want to encode Acme Corp's specific support processes, escalation chains, and team structure into the plugin.

Modification 1: Add Acme's escalation chain

Edit `/skills/escalation/SKILL.md`:

```
## Acme Corp Escalation Chain

### L1 → L2 (Support Escalation)
**From:** Frontline Support (Alice, Bob, Carol)
**To:** Senior Support (David, Emma)
**When:** Issue requires deeper investigation or specialized product knowledge
**Slack channel:** #support-escalations
**SLA:** L2 acknowledges within 2 hours

### L2 → Engineering
**From:** Senior Support
**To:** Engineering team leads:
  - **Backend:** Frank (Slack: @frank)
  - **Frontend:** Grace (Slack: @grace)
  - **Mobile:** Henry (Slack: @henry)
**When:** Confirmed bug, infrastructure issue, needs code change
**Slack channel:** #engineering-escalations
**SLA:** Engineering acknowledges within 4 hours (P1/P2), next business day (P3)

### L2 → Product
**From:** Senior Support
**To:** Product Manager: Iris (Slack: @iris)
**When:** Feature gap, design decision, workflow doesn't match customer expectations
**Slack channel:** #product-feedback
**SLA:** Product acknowledges within 1 business day

### Any → Security/Compliance
**From:** Any support tier
**To:** Security Lead: Jack (Slack: @jack), Compliance Officer: Karen (Slack: @karen)
**When:** Potential data exposure, HIPAA concern, vulnerability report
**Slack channel:** #security-alerts (urgent), #compliance (non-urgent)
**SLA:** Security acknowledges within 1 hour for critical, 4 hours for high

### Any → Leadership
**From:** Any tier (usually L2 or support manager)
**To:** VP Support: Laura (Slack: @laura), CEO: Mike (for existential issues)
**When:** High-revenue churn risk, legal/PR risk, SLA breach on enterprise account
**Email:** laura@acme.com (VP), mike@acme.com (CEO — use sparingly)
**SLA:** Leadership acknowledges within business hours
```

Modification 2: Add Acme's priority definitions

Edit `/skills/ticket-triage/SKILL.md`:

```
## Acme Corp Priority Framework

### P1 — Critical
**Criteria:**
- Production completely down for any Enterprise customer
- PHI exposure or HIPAA breach
- Data loss or corruption affecting multiple customers
- Security incident in progress

**Acme SLA:** 1-hour response, continuous work, updates every hour to customer

**Who gets paged:** On-call engineer (via PagerDuty), Support Manager, VP Engineering

### P2 — High
**Criteria:**
- Major feature broken for Enterprise customer
- Production issue affecting 3+ customers (any tier)
- Enterprise customer blocked with no workaround
- Integration failure blocking clinical workflow

**Acme SLA:** 2-hour response, active investigation same day, updates every 4 hours

**Who gets notified:** Senior Support, relevant engineering team lead (via Slack)

### P3 — Medium
**Criteria:**
- Feature broken but workaround available
- Single Pro-tier customer affected
- Non-blocking issue for Enterprise customer

**Acme SLA:** 4-hour response (business hours), resolution within 3 business days

**Who handles:** Frontline or Senior Support, escalate to Engineering if needed

### P4 — Low
**Criteria:**
- Cosmetic issues
- Feature requests
- General how-to questions
- Free-tier customer issues

**Acme SLA:** 1 business day response, resolution at normal pace

**Who handles:** Frontline Support
```

Modification 3: Add Acme's auto-response templates

Edit `/skills/ticket-triage/SKILL.md` to replace generic templates with Acme-specific ones:

```
### Bug — Initial Response (Acme Corp)

Hi [Name],

Thank you for reporting this. I can see how [specific impact] would
disrupt your [clinical workflow / team's work / patients' care].

I've logged this as a [priority] issue and our [team] is investigating.
[If workaround exists: "In the meantime, you can [workaround]."]

You'll receive an update from me within [SLA timeframe]. If this is
blocking urgent patient care, please call our support hotline at
1-800-ACME-911 and reference ticket #[ticket ID].

Best,
[Your name]
Acme Support Team
```

Approach C: Complementary Skills via CLAUDE.md (Lightest Touch)

What it is: Keep the original plugin completely untouched. Add routing rules in your project's `CLAUDE.md` file and create additional skills in your project's `.claude/skills/` directory that layer on top of the plugin.

When to use:

- Adding project-specific or company-specific knowledge without modifying the plugin
- Layering industry knowledge on top of a general plugin
- You want to receive plugin updates seamlessly (no merge conflicts)
- Quick additions that only apply to specific projects

Advantages:

- Zero modification to the original plugin (receives updates cleanly)
- Changes are project-local (don't affect other projects)
- Easy to version control in your project repo

Limitations:

- Routing rules in `CLAUDE.md` may not trigger as reliably as built-in skill logic
- No structural link between your additions and the plugin (they coexist by convention)
- If the plugin changes significantly, your routing rules may need updating

Example: Adding Company Voice and Process Knowledge

Your company (Acme Corp) has a specific brand voice, terminology, and internal approval process that should apply to all customer communications.

Step 1: Create a complementary skill

In your project directory, create `.claude/skills/acme-support-standards/SKILL.md`:

```
---
name: acme-support-standards
description: >
  Acme Corp support team standards, brand voice, terminology, and
  approval processes. Use when drafting customer responses, triaging
  tickets for Acme customers, or following Acme's internal support
  workflows.
---

# Acme Corp Support Standards

## Brand Voice for Customer Communications

### Voice Attributes
- **Empathetic and patient**: Healthcare is stressful. Acknowledge the
  pressure clinicians are under.
- **Precise and clinical**: Use correct medical terminology when speaking
  to clinical users. Avoid oversimplifying.
- **Calm and reassuring**: Even in outages, convey competence and control.
- **Action-oriented**: Every response should have clear next steps.

### Terminology Rules

| Use This | Not This | Reason |
|----------|----------|--------|
| Acme Platform | our platform, the tool, our system | Always use brand name |
| clinician | user, customer (when referring to HCPs) | Respectful, professional |
| patient chart | patient record, EMR | Matches how our users speak |
| HL7 message | HL7 feed, data feed (when referring to HL7 specifically) | Precision m
| issue, incident | problem, bug (in customer-facing language) | Professional tone |

### Phrases to Avoid
- "It's easy to..." (what's easy for us may not be easy for them)
- "Just do X" (minimizes their situation)
- "The system is down" (use "Acme Platform is experiencing an issue")
- Any humor or casual language in outage communications (inappropriate for healthcare 

## Approval Workflow

### Customer-Facing Communications
```

```
| Type | Approval Required | Turnaround |
├──────┼───────────────────┼────────────┤
| Standard ticket response | None (frontline judgment) | Immediate |
| Escalation response (P1/P2) | Senior Support review before sending | 1 hour |
| Outage notification | Support Manager approval | 30 minutes |
| Feature request decline | Product Manager review | 1 business day |
| Billing adjustment | Billing Manager approval | 4 hours |
| Legal/compliance question | Compliance Officer review | 4 hours |

### Escalations to Engineering

| Severity | Approval Required | Notification |
├──────────┼───────────────────┼──────────────┤
| P1 | None (escalate immediately) | Notify Support Manager after escalating |
| P2 | Senior Support judgment | CC Support Manager on escalation |
| P3 | Senior Support approval | No notification needed |

## Internal Communication Standards

### Slack Etiquette
- **#support-team**: General support discussion, questions, knowledge sharing
- **#support-escalations**: L1 → L2 escalations (tag @senior-support)
- **#engineering-escalations**: L2 → Engineering (tag relevant team lead)
- **#security-alerts**: Any potential PHI exposure or security incident (tag @security

### Escalation Slack Message Format
```

🚨 [P1/P2/P3] [Category]: [One-line summary]

Customer: [Name/Account] Impact: [What's broken for them] Tried: [What we've done so far] Need: [What we need from this team] Deadline: [When customer expects resolution]

Ticket: [Link]

```
## Acme-Specific Customer Context

### Customer Tiers and ARR
When triaging, consider:
- **Enterprise** ($100K+ ARR): P2 minimum for any production issue, notify Account Man
- **Pro** ($10K-$100K ARR): Standard prioritization
- **Starter** (<$10K ARR): P3 for most issues unless blocking clinical workflow

### Key Accounts (Escalate Carefully)
- **Memorial General Hospital** (ARR: $500K) — Contact: Dr. Sarah Chen, CMIO
- **Regional Health Network** (ARR: $300K) — Contact: Mike Torres, VP IT
- **City Clinics Group** (ARR: $250K) — Contact: Linda Park, Operations Director

Any P1/P2 for these accounts: notify Account Manager immediately (Slack: @account-team
```

Step 2: Add routing rules in `CLAUDE.md`

Edit or create `CLAUDE.md` in your project root:

```
# Acme Corp Support Project

## Plugin Extensions

When using the customer-support plugin for Acme Corp tickets:
- Always apply the acme-support-standards skill for brand voice and terminology
- Follow Acme's approval workflow before sending responses
- Use Acme's priority definitions (P1 = Enterprise production down or PHI exposure)
- Check customer tier (Enterprise/Pro/Starter) and adjust priority accordingly
- For Enterprise accounts, notify Account Manager if P1/P2 escalation

## Response Drafting Rules

When drafting customer responses:
- Use "Acme Platform" not "our platform"
- Use "clinician" not "user" when referring to healthcare professionals
- Avoid casual language or humor in outage communications
- If escalation is P1 or P2, note that Senior Support review is required before sendin
- If customer is Memorial General, Regional Health, or City Clinics, flag for Account

## Escalation Rules

When escalating to Engineering:
- Use Acme's Slack escalation format (see acme-support-standards skill)
- Post in #engineering-escalations and tag relevant team lead (@frank for backend, @gr
- If P1, also notify Support Manager (@laura)
- If PHI-related, post in #security-alerts and tag @security-team
```

Step 3: Test the integration

```
/draft-response Customer (Memorial General Hospital) is asking why their HL7 ADT feed
```

Claude should:

1. Load the customer-support plugin (for `/draft-response` command)
2. Load the response-drafting skill (from the plugin)
3. Load the acme-support-standards skill (from your project)
4. Apply Acme's brand voice and terminology
5. Recognize Memorial General as a key account
6. Note in the internal draft notes: "Enterprise account ($500K ARR), notify Account Manager if this escalates to P1/P2"

Part VI: File Reference

Complete File Tree

```
customer-support/
├── .claude-plugin/
│   └── plugin.json                # Plugin metadata (name, version, description, 
│
├── commands/                      # Five slash commands
│   ├── draft-response.md          # /draft-response — draft customer communicatio
│   ├── escalate.md                # /escalate — package escalations for eng/produ
│   ├── kb-article.md              # /kb-article — write KB articles from resolved
│   ├── research.md                # /research — multi-source research with confid
│   └── triage.md                  # /triage — categorize, prioritize, route ticke
│
├── skills/                        # Five domain knowledge skills
│   ├── customer-research/
│   │   └── SKILL.md               # Multi-source research methodology, confidence
│   ├── escalation/
│   │   └── SKILL.md               # When/how to escalate, structured escalation f
│   ├── knowledge-management/
│   │   └── SKILL.md               # KB article types, searchability, maintenance
│   ├── response-drafting/
│   │   └── SKILL.md               # Communication best practices, tone guidelines
│   └── ticket-triage/
│       └── SKILL.md               # Category taxonomy, P1-P4 priority, routing ru
│
├── .mcp.json                      # MCP server configurations (7 pre-configured s
├── CONNECTORS.md                  # Placeholder system documentation and tool opt
├── README.md                      # Plugin overview and quick start
└── LICENSE                        # License information (if included)
```

Cross-Reference Map

Which commands use which skills:

COMMAND	PRIMARY SKILL(S)	SECONDARY SKILLS
`/triage`	ticket-triage	customer-research (for duplicate detection and context gathering)
`/research`	customer-research	—

COMMAND	PRIMARY SKILL(S)	SECONDARY SKILLS
`/draft-response`	response-drafting	customer-research (for gathering customer context)
`/escalate`	escalation	ticket-triage (for priority assessment), customer-research (for context)
`/kb-article`	knowledge-management	—

Which commands use which MCP servers:

COMMAND	COMMONLY USED MCP SERVERS
`/triage`	Support platform (Intercom), Knowledge base (Guru/Notion), Project tracker (Atlassian), CRM (HubSpot)
`/research`	Knowledge base (Guru/Notion), Support platform (Intercom), Chat (Slack), Email (MS365), CRM (HubSpot), Cloud storage (MS365)
`/draft-response`	CRM (HubSpot), Support platform (Intercom), Email (MS365), Chat (Slack)
`/escalate`	Support platform (Intercom), CRM (HubSpot), Project tracker (Atlassian), Chat (Slack)
`/kb-article`	Knowledge base (Guru/Notion), Support platform (Intercom)

Skill interdependencies:

- `ticket-triage` → references `customer-research` for duplicate detection methodology
- `escalation` → references `ticket-triage` for priority definitions
- `response-drafting` → references `escalation` for escalation communication guidelines
- `knowledge-management` → references `customer-research` for source prioritization when writing articles

Part VII: Troubleshooting & FAQ

Common Issues and Solutions

Issue 1: Plugin Not Loading or Commands Not Recognized

Symptoms:

- `/triage` shows "command not found"
- Plugin doesn't appear in Claude's plugin list
- Commands are grayed out or unavailable

Possible Causes & Solutions:

Cause: Plugin not installed

- **Solution:** Run `claude plugins add customer-support` and restart Claude

Cause: Plugin installed but not activated for this project

- **Solution:** Check Claude settings → Plugins → ensure "customer-support" is enabled

Cause: Plugin directory structure is malformed

- **Solution:** Verify `.claude-plugin/plugin.json` exists and is valid JSON. Check that `commands/` directory contains `.md` files.

Cause: Plugin name mismatch

- **Solution:** The `name` field in `plugin.json` must match the directory name (e.g., if directory is `customer-support/`, `name` must be `"customer-support"`)

Issue 2: MCP Server Not Connecting

Symptoms:

- `/research` doesn't search connected tools
- "Could not connect to [service]" error
- Commands work but don't pull data from external sources

Possible Causes & Solutions:

Cause: MCP server not authenticated

- **Solution:** Go to Claude Settings → MCP Servers → find the server (e.g., "intercom") → click "Authenticate" → complete OAuth flow

Cause: Insufficient permissions

- **Solution:** Re-authenticate and ensure you grant all requested permissions (read access to conversations, contacts, etc.)

Cause: Workspace/account mismatch

- **Solution:** Verify you selected the correct workspace during authentication. If you have multiple Intercom workspaces, make sure you chose the right one.

Cause: MCP server URL incorrect or outdated

- **Solution:** Check `.mcp.json` for correct URLs. Verify the service's MCP endpoint hasn't changed (check vendor documentation).

Cause: Token expired (for API token-based auth like Guru)

- **Solution:** Generate a new API token in the service's settings and re-authenticate in Claude

Issue 3: Commands Return Generic Answers Without Context

Symptoms:

- `/triage` doesn't check for duplicate tickets
- `/research` only uses web search, doesn't check internal docs
- `/draft-response` doesn't pull customer history
- Commands work but feel "disconnected" from your data

Diagnosis: MCP servers are not connected or not working properly.

Solutions:

1. **Verify MCP connections:**

   ```
   Ask Claude: "What MCP servers are currently connected?"
   ```

   Claude will list active connections. If your support platform, KB, or CRM isn't listed, it's not connected.

2. **Test individual connections:**

   ```
   /research [topic you know exists in your knowledge base]
   ```

Check if Claude cites your KB in the sources. If not, the KB connection isn't working.

3. **Re-authenticate:**
   - Go to Settings → MCP Servers
   - Find the problematic server
   - Click "Disconnect" then "Authenticate" again
4. **Check permissions:**
   - Some services (like Notion) require you to explicitly share pages/databases with the integration
   - Some services (like Slack) require admin approval for the app

Issue 4: Skills Not Activating

Symptoms:

- Commands run but don't apply the expected knowledge (e.g., `/triage` doesn't use P1-P4 framework)
- Responses feel generic
- No mention of categories, priorities, or frameworks from skills

Possible Causes & Solutions:

Cause: Skill trigger phrases not matching

- **Solution:** Skills activate based on conversation context. Make sure your question/command includes trigger phrases. For example, `/triage` should automatically trigger `ticket-triage`, but if it doesn't, try: `/triage <issue> — use the P1-P4 priority framework`

Cause: Skill file has syntax errors

- **Solution:** Check that `SKILL.md` has valid YAML frontmatter (between `---` markers) and that `name` and `description` fields are present

Cause: Skill description doesn't include trigger phrases

- **Solution:** The `description` field in the skill's frontmatter must include specific phrases that match what users would say. If you modified a skill, verify the description still covers the use cases.

Issue 5: Custom Modifications Not Taking Effect

Symptoms:

- You edited a skill or command file but changes don't appear

- Plugin still uses old version of content

Solutions:

1. **Reload the plugin:**
    - Restart Claude entirely (quit and reopen)
    - Or reload the project: close the project, reopen it
2. **Check file location:**
    - If you forked the plugin, make sure the fork is in the right location (`.claude/plugins/` in your project or in the global plugins directory)
    - Make sure you're editing the fork, not the original
3. **Verify no syntax errors:**
    - YAML frontmatter must be valid (no unmatched quotes, correct indentation)
    - Markdown must be well-formed (headers, code blocks closed properly)
4. **Clear Claude's cache (advanced):**
    - Some systems cache plugin content. If restarting doesn't help, clear Claude's cache (method varies by OS — check Claude documentation)

Plugin Loading Verification

To verify the plugin is loaded correctly:

Step 1: Check plugin list

```
Ask Claude: "What plugins are currently active?"
```

Claude should list "customer-support" (or your forked name).

Step 2: Check command availability

```
Type: /tri
```

Autocomplete should suggest `/triage` with the description "Triage and prioritize a support ticket or customer issue"

Step 3: Test a command

```
/triage Test ticket: customer can't log in
```

Claude should produce a structured triage output with category, priority, routing, etc.

Step 4: Verify skills are loaded

```
Ask Claude: "What skills does the customer-support plugin provide?"
```

Claude should list: ticket-triage, customer-research, response-drafting, escalation, knowledge-management

Step 5: Verify MCP connections

```
Ask Claude: "What MCP servers are connected for customer support workflows?"
```

Claude should list Slack, Intercom, HubSpot, Guru, Atlassian, Notion, MS365 (or whichever subset you've authenticated).

MCP Debugging

If MCP servers aren't working:

Step 1: Check connection status

In Claude Settings → MCP Servers, each server should show "Connected" or "Authenticated". If it shows "Not Connected" or "Authentication Failed," click it to re-authenticate.

Step 2: Test with direct query

Instead of using a command, ask Claude directly:

```
Search Slack for messages about [topic] in the past week
```

If Claude can't find anything or says "I don't have access to Slack," the MCP connection isn't working.

Step 3: Check service-side permissions

- **Slack:** Go to Slack workspace settings → Apps → find "Claude" → verify it has permission to read channels

- **Intercom:** Go to Intercom settings → Apps & Integrations → find "Claude" → verify it has read permissions
- **HubSpot:** Go to HubSpot settings → Integrations → Private Apps → find "Claude" → verify scopes include CRM read access
- **Notion:** Go to Notion settings → Connections → find "Claude" → verify pages/databases are shared with the integration

Step 4: Re-authenticate

If permissions look correct but it still doesn't work, disconnect and reconnect:

1. Settings → MCP Servers → [Server name] → Disconnect
2. Wait 10 seconds
3. Settings → MCP Servers → [Server name] → Authenticate
4. Complete OAuth flow, granting all requested permissions

Step 5: Check MCP server URLs

If you're using a self-hosted or custom MCP server, verify the URL in `.mcp.json` is correct and accessible. Test the URL in a browser (some will return a 404 but others will return server info).

FAQ

Q: Can I use this plugin without connecting any MCP servers?

A: Yes. The plugin works standalone — commands will prompt you to provide context manually (paste customer messages, ticket history, etc.) instead of pulling it automatically. However, you'll lose the major efficiency gains.

Q: Which MCP servers are most important to connect?

A: Support platform (Intercom, Zendesk, etc.) and Knowledge base (Guru, Notion, Confluence) provide 80% of the value. CRM adds customer context. The rest are nice-to-have.

Q: Can I connect multiple tools in the same category (e.g., both Guru and Notion as knowledge bases)?

A: Yes. Claude will search both when using `~~knowledge base` placeholders. This works well if you have different types of knowledge in different systems (e.g., Guru for verified support knowledge, Notion for internal docs).

Q: How do I update the plugin when a new version is released?

A: Run `claude plugins update customer-support`. If you've forked the plugin, you'll need to manually merge changes from the upstream version into your fork.

Q: Can I create my own commands?

A: Yes. Add `.md` files to the `commands/` directory (or your fork's `commands/` directory). Follow the structure of existing commands: YAML frontmatter with `description` and `argument-hint`, then Markdown body with instructions for Claude.

Q: Can I modify the P1-P4 priority definitions to match my company's SLAs?

A: Yes. Edit `/skills/ticket-triage/SKILL.md` and update the "Priority Framework" section with your SLA timings and criteria. If you want to preserve the original for reference, fork the plugin first.

Q: How do I share my customized plugin with my team?

A: Fork the plugin, make your changes, then package it:

```
cd your-forked-plugin
zip -r your-plugin-name.plugin . -x "*.DS_Store"
```

Share the `.plugin` file. Teammates can install it with `claude plugins add /path/to/your-plugin-name.plugin`

Q: Can I use this plugin in Claude Code (CLI) or is it Cowork-only?

A: It works in both Claude Code and Cowork. Some features (like MCP server browsing in settings) are more convenient in the Cowork desktop app, but all functionality works in Claude Code.

Q: What if my support platform isn't listed (e.g., we use Zendesk)?

A: Check if Zendesk has an MCP server (search "Zendesk MCP" or check the MCP registry). If it does, add it to `.mcp.json` following the pattern of existing servers. If not, you can still use the plugin — provide ticket context manually or build a custom MCP server for Zendesk (see MCP documentation).

Q: Can I add new skills without modifying the plugin?

A: Yes, use Approach C (Complementary Skills via CLAUDE.md). Create skills in `.claude/skills/` in your project directory. Add routing rules in `CLAUDE.md` to tell Claude when to use your custom skills alongside the plugin's skills.

Q: How do I know which version of the plugin I have installed?

A: Check `.claude-plugin/plugin.json` — the `version` field shows the installed version. Current official version: 1.0.0.

Q: The plugin is great for reactive support (responding to tickets). Can it help with proactive support?

A: Yes, with some creativity:

- Use `/research` to identify common customer questions and create preemptive KB articles
- Use `/kb-article` to document known issues before they become frequent tickets
- Use `/escalate` to package patterns (e.g., "5 customers asked about X this month") as feature requests to product

Q: Can I integrate this with our ticket routing system to auto-triage tickets?

A: The plugin itself doesn't auto-triage, but you could:

1. Use the plugin to triage sample tickets and document triage rules
2. Export those rules to your ticketing system's automation
3. Or build a custom integration using Claude's API to auto-triage tickets via the plugin

This would be a custom implementation beyond the plugin's out-of-box scope.

Appendix: Quick Reference Card

Five Commands at a Glance

COMMAND	WHEN TO USE	QUICK SYNTAX
`/triage`	Categorize and prioritize incoming tickets	`/triage <customer issue>`
`/research`	Answer customer questions with multi-source research	`/research <question>`
`/draft-response`	Write professional customer communications	`/draft-response <situation>`
`/escalate`	Package escalations for engineering/product	`/escalate <issue> [customer]`
`/kb-article`	Create KB articles from resolved tickets	`/kb-article <resolved issue>`

Five Skills at a Glance

SKILL	WHAT IT PROVIDES	ACTIVATES WHEN
ticket-triage	P1-P4 priority, category taxonomy, routing rules	Categorizing tickets, assessing priority
customer-research	Multi-source search, confidence scoring, source prioritization	Researching questions, gathering context
response-drafting	Tone guidelines, templates, communication best practices	Drafting customer responses
escalation	Structured escalation format, impact assessment, repro steps	Packaging escalations, assessing severity
knowledge-management	Article types, searchability, KB hygiene	Writing KB articles, maintaining docs

Priority Framework Quick Reference

PRIORITY	CRITERIA	RESPONSE SLA	EXAMPLE
P1	Production down, data loss, security breach, all users affected	1 hour, continuous work	API completely unavailable for all customers
P2	Major feature broken, many users affected, no workaround	4 hours, same-day investigation	Dashboard not loading for Enterprise customer
P3	Feature partially broken, workaround available, small team affected	1 business day, 3-day resolution	Export missing one column, can export and add manually
P4	Minor inconvenience, cosmetic, general question, feature request	2 business days, normal pace	Button text is unclear

Category Taxonomy Quick Reference

CATEGORY	WHEN TO USE
Bug	Product behaving incorrectly (error, crash, not working as expected)
How-to	Customer needs guidance or training
Feature request	Customer wants capability that doesn't exist

CATEGORY	WHEN TO USE
Billing	Payment, subscription, invoice issues
Account	Access, permissions, user management
Integration	Third-party connections, API, webhooks
Security	Data exposure, compliance, vulnerabilities
Data	Data quality, migration, import/export
Performance	Speed, reliability, timeouts, availability

End of Customer Support Plugin Handbook

For questions, feedback, or contributions, contact the plugin maintainers or visit the plugin repository.

RationalEyes.ai

contact@rationaleyes.ai

Intelligent Automation for Knowledge Work

Customer Support Plugin Handbook — Version 1.0.0