THE CONTENT ENGINE

SUBAGENT 4

Action Content

Automatically extract objections from sales call transcripts, categorize them by theme, and generate objection-handling battle cards, follow-up email sequences, and “why us” website pages that neutralize the specific concerns killing your deals.

A step-by-step technical implementation guide.

Part 4 of 4 • The Content Engine Series

What This Subagent Does

The Action subagent closes the loop between sales conversations and marketing content. It extracts the real objections your prospects raise on calls, identifies which ones are killing deals, and generates the content that neutralizes those objections — both in real-time on calls and in follow-up communications.

Every other subagent in this series generates content for prospects who haven't talked to sales yet. This one is different: it arms your sales team with specific, tested responses to the exact concerns that come up in live deals. It's the difference between a rep scrambling for an answer and a rep delivering a confident, proof-backed reframe.

The Workflow at a Glance

1. Call recording platform API extracts transcripts from recent closed-won and closed-lost deals
2. Claude identifies and categorizes every objection by theme
3. Claude ranks objections by frequency and impact on deal outcomes
4. Claude generates: live reframes, follow-up emails, email sequences, battle cards, and website copy
5. n8n routes outputs to sales enablement library and notifies the team

Why Closed-Lost Data Is Gold

Most teams only train on what works. But the highest-leverage insight is in what doesn't work: the objections that appeared in closed-lost deals but not in closed-won deals. These are the deal-killers — the concerns your team currently has no good answer for. This subagent surfaces them automatically.

What You Need Before Starting

Required Accounts & API Keys

Tool	Purpose	Notes
n8n	Workflow orchestration	Same instance as Subagents 1–3.
Claude API	Objection extraction + content generation	Anthropic API key. Opus recommended for nuanced sales content.
Call Recording Platform	Transcript source	Gong, Chorus, Fireflies, Otter, or any platform with API access to transcripts.

Google Sheets	Objection database + battle card storage	Central reference for all extracted objections and generated content.
Google Drive	Asset storage	Where battle cards, email templates, and website copy are saved.
Slack (optional)	Team notifications	Alerts sales when new objection-handling assets are ready.

Supported Call Recording Platforms

This guide uses Gong as the primary example because it has the most mature API for transcript extraction. However, the same workflow works with any platform that exposes transcripts via API. Here's a comparison:

Platform	API Access	Transcript Format	Setup Difficulty
Gong	Full REST API. Requires Gong Enterprise or API add-on. Endpoints for calls, transcripts, trackers, and deal data.	Structured JSON with speaker labels, timestamps, and topic tags.	Medium — requires admin approval for API key.
Chorus (ZoomInfo)	REST API available on Enterprise plans. Access to call recordings and transcripts.	JSON with speaker separation and topic detection.	Medium — requires ZoomInfo admin access.
Fireflies.ai	GraphQL API. Available on Pro plan and above. Endpoints for transcripts, action items, and summaries.	Structured with speaker labels, AI summaries, and key topics.	Easy — self-serve API key generation.
Otter.ai	Limited API. Transcript export available on Business plan.	Plain text or JSON with speaker labels.	Easy — but less structured data than alternatives.
Manual Upload	No API needed. Export transcripts as text files and upload to a Google Drive folder that n8n monitors.	Plain text. Less structured but works.	Easiest — no technical setup required.

No Call Recording Platform?

If you don't use Gong, Chorus, or Fireflies, you can still run this subagent. Option 1: Have your reps manually log the top objections they hear each week in a Google Sheet. Option 2: Use Zoom or

> Google Meet’s built-in transcript feature, export the transcripts, and drop them in a Google Drive folder. Option 3: Have reps record a 2-minute voice memo after each call summarizing the objections, transcribe with Whisper or Otter, and feed that in. The extraction quality depends on transcript quality, but even rough notes produce useful outputs.

Prerequisite: Document Your Proof Points

Claude can’t generate credible objection responses without proof. Before building this subagent, compile your arsenal of evidence:

1. **Customer case studies:** Specific results with named customers (or anonymized industry examples). Format: "[Customer type] achieved [specific metric] in [timeframe]." You need 5–10 of these covering different use cases and company sizes.
2. **Data points:** Internal benchmarks, survey results, or aggregate customer data. Examples: "Average implementation time: 14 days," "95% customer retention rate," "3.2x average ROI in year one."
3. **Customer quotes:** Direct quotes from happy customers, ideally addressing specific concerns. The best quotes preemptively answer objections: "We were worried about the switching cost, but the migration took less than a week."
4. **Competitive proof:** Win stories against specific competitors. Anonymized is fine: "A 200-person fintech company switched from [Competitor] and saw 40% faster onboarding."

Store all of this in a Google Doc or JSON file. Claude will reference these proof points when generating objection responses, emails, and website copy.

Step 1: Extract Transcripts from Your Call Platform

This step pulls raw call transcripts from your recording platform, filtered by deal outcome and date range. You want both closed-won and closed-lost transcripts — comparing objections across outcomes reveals which objections are deal-killers versus which ones are speed bumps.

1A. Gong API Setup

Gong's API requires an access key generated by your Gong admin. Once you have credentials, the relevant endpoints are:

Endpoint	Purpose	Method
/v2/calls	List calls with filters (date range, outcome, participants)	GET
/v2/calls/{id}/transcript	Full transcript for a specific call	GET
/v2/calls/extensive	Bulk call data including trackers, topics, and action items	POST
/v2/stats/activity/scorecards	Call scoring data (if you use Gong scorecards)	POST

n8n Workflow: Fetch Call List

```
// Step 1: Get list of recent calls
POST https://api.gong.io/v2/calls/extensive

Headers:
  Authorization: Basic {{$credentials.gongApiBase64}}
  Content-Type: application/json

Body:
{
  "filter": {
    "fromDateTime": "2025-01-01T00:00:00Z",
    "toDateTime": "2025-02-01T00:00:00Z"
  },
  "contentSelector": {
    "exposedFields": {
      "content": {
        "trackers": true,
        "topics": true,
        "pointsOfInterest": true
      },
      "collaboration": {
        "publicComments": true
      },
      "parties": true
    }
  }
}

// Response includes call IDs, participants, topics, and tracker hits
```

n8n Workflow: Fetch Individual Transcripts

```
// Step 2: For each call ID, fetch the full transcript
GET https://api.gong.io/v2/calls/{{$json.callId}}/transcript

Headers:
  Authorization: Basic {{$credentials.gongApiBase64}}

// Response:
{
  "callTranscripts": [
    {
      "callId": "123456",
      "transcript": [
        {
          "speakerId": "buyer-id",
          "topic": "Pricing Discussion",
          "sentences": [
            {
              "start": 1234.5,
              "end": 1240.2,
              "text": "That's more than we budgeted for this quarter..."
            }
          ]
        }
      ]
    }
  ]
}
```

1B. Alternative: Fireflies.ai API

Fireflies uses a GraphQL API, which is simpler for transcript extraction:

```
POST https://api.fireflies.ai/graphql

Headers:
  Authorization: Bearer {{$credentials.firefliesApi}}
  Content-Type: application/json

Body:
{
  "query": "{ transcripts(limit: 50) {
    id title date duration
    sentences { speaker_name text }
    action_items { text }
    summary { overview }
  }}"
}
```

1C. Alternative: Manual Transcript Upload

If you don’t have API access, use this lightweight approach:

1. **Create a Google Drive folder** called "Call Transcripts" with two subfolders: "Closed-Won" and "Closed-Lost."
2. **After each call** (or weekly), export transcripts from your call platform and drop the text files into the appropriate subfolder.
3. **In n8n**, use the Google Drive Trigger node to watch for new files in both folders. When new transcripts appear, the workflow kicks off automatically.

This approach requires more manual effort but works with any call recording platform, including simple Zoom or Google Meet transcripts.

Step 2: Extract and Categorize Objections with Claude

This is the most important step in the subagent. Claude analyzes raw transcripts, identifies every distinct objection, and categorizes them by theme, frequency, and impact on deal outcomes.

2A. The Objection Extraction Prompt

This prompt processes transcripts in batches (5–10 calls per batch) to give Claude enough context to identify patterns.

```
System Prompt:
You are a sales intelligence analyst. Your job is to extract,
categorize, and rank prospect objections from sales call transcripts.
An objection is any concern, hesitation, pushback, or question that
signals resistance to moving forward with a purchase.

User Message:
Here are transcripts from recent sales calls.

CLOSED-WON CALLS:
---
Call 1 (Deal: {{$json.deal1_name}}):
{{$json.transcript1}}
---
Call 2 (Deal: {{$json.deal2_name}}):
{{$json.transcript2}}
---
[...additional transcripts...]

CLOSED-LOST CALLS:
---
Call A (Deal: {{$json.dealA_name}}, Lost reason: {{$json.lostReasonA}}):
{{$json.transcriptA}}
---
[...additional transcripts...]

Analyze these transcripts and extract every distinct objection.
For each objection:

1. EXACT QUOTE: The prospect’s actual words (or closest paraphrase)
2. THEME: Categorize into one of these themes:
   - Price (too expensive, budget constraints, ROI uncertainty)
   - Timing (not now, other priorities, end of quarter)
   - Trust (too new, unproven, references needed)
   - Switching Cost (migration effort, training, disruption)
   - Internal Buy-in (need approval, multiple stakeholders)
   - Competitor (already using X, evaluating Y)
   - Product Fit (missing feature, wrong use case)
   - Contract (terms, commitment length, flexibility)
3. FREQUENCY: How many calls did this objection appear in?
4. WON/LOST SPLIT: Did this objection appear more in won or lost deals?
5. DEAL-KILLER SCORE (1-5): How likely is this objection to kill a deal?
   5 = appears mostly in lost deals and rarely in won deals
   3 = appears equally in both
   1 = appears in won deals too (meaning it’s manageable)
6. CURRENT HANDLING: How did reps handle this objection in the
   transcripts? Quote their response if available.

Output JSON:
{
  "total_calls_analyzed": 10,
  "won_calls": 5,
  "lost_calls": 5,
  "objections": [
    {
      "id": 1,
      "objection": "clean phrasing of the objection",
      "exact_quote": "prospect’s actual words",
      "theme": "Price",
      "frequency": 7,
      "appeared_in_won": 3,
      "appeared_in_lost": 4,
      "deal_killer_score": 4,
      "current_handling": "how reps currently respond",
      "current_handling_effective": true/false
    }
  ],
  "theme_summary": {
    "Price": { "count": 3, "avg_deal_killer_score": 3.7 },
    "Timing": { "count": 2, "avg_deal_killer_score": 2.5 }
  }
}

Sort objections by deal_killer_score descending.
Return valid JSON only.
```

2B. Handling Long Transcripts

A single sales call transcript can be 5,000–15,000 words. Sending 10 full transcripts in one prompt will exceed context limits. Use one of these approaches:

1. **Pre-summarize with Claude (recommended):** Before the extraction step, run each transcript through a summarization prompt that extracts only the objection-relevant segments. This reduces each transcript to 500–1,000 words.

2. **Use Gong's topics and trackers:** If you're on Gong, use the "topics" and "trackers" data from the /v2/calls/extensive endpoint. Gong pre-identifies discussion topics and can tag objection-related moments, giving you focused excerpts instead of full transcripts.
3. **Process one call at a time:** Run the extraction prompt on individual transcripts, then run a second aggregation prompt that combines all extracted objections across calls. More API calls, but avoids context limits.

Pre-Summarization Prompt

```
System Prompt:
You are a sales call analyst. Extract only the sections of this
transcript where the prospect raises concerns, objections, hesitations,
or asks challenging questions. Include 2-3 sentences of context before
and after each objection for the rep's response.

User Message:
Here is a full sales call transcript:
{{$json.full_transcript}}

Extract every objection-relevant segment. For each segment, output:
{
  "segments": [
    {
      "timestamp": "approximate time in call",
      "prospect_said": "their objection or concern",
      "rep_responded": "how the rep handled it",
      "context": "what was being discussed"
    }
  ]
}

If no objections were raised, return { "segments": [] }.
```

Step 3: Generate Objection-Handling Content

For each objection extracted in Step 2, Claude generates five content types. These form a complete objection-handling system that covers live calls, post-call follow-up, multi-touch sequences, internal reference, and website preemption.

3A. The Master Objection-Handling Prompt

This prompt generates all five content types per objection in a single API call. It references your proof points document for evidence-backed responses.

```
System Prompt:
You are a sales enablement strategist. You write objection-handling
content that is confident without being defensive, evidence-backed
without being preachy, and empathetic without being soft.

PROOF POINTS AVAILABLE:
{{$json.proof_points}}

PRODUCT OVERVIEW:
[YOUR PRODUCT DESCRIPTION]

ICP:
[YOUR IDEAL CUSTOMER PROFILE]

User Message:
Here are the top objections from our sales calls, ranked by
deal-killer score:

{{$json.ranked_objections}}

For EACH objection, generate all five of these content types:

TYPE 1 — LIVE REFRAME (for use on calls)
  3 sentences. The rep says this in real time.
  Structure: Acknowledge the concern → Reframe the thinking →
  Bridge to proof.
  Must sound natural in conversation, not scripted.
  Do NOT start with "I understand" or "That’s a great question."

TYPE 2 — FOLLOW-UP EMAIL (under 150 words)
  Sent within 24 hours after the objection was raised on a call.
  Structure: Reference the conversation → Address the concern
  with a specific proof point → Provide a resource or next step.
  Subject line included.

TYPE 3 — OBJECTION-HANDLING EMAIL SEQUENCE (3 emails)
```

```
  For when the objection stalls a deal. Spaced 3-5 days apart.
  Email 1: Readdress the concern with new proof
  Email 2: Share a case study or customer story relevant to the concern
  Email 3: Create urgency or reframe the cost of inaction
  Each email under 120 words. Subject lines included.

TYPE 4 — BATTLE CARD ENTRY
  A structured reference card for the sales team.
  Includes: Objection (as the buyer says it), What they really mean,
  Best response, Proof point to cite, Common follow-up question,
  and Red flag (when this objection signals a bad-fit deal).

TYPE 5 — WEBSITE COPY
  A "Why Us" or "Why Now" section (100 words) for the website that
  preemptively handles this objection before a prospect ever raises it.
  Should work as a standalone section on a landing page.
  Includes a section heading (H2).

Output JSON:
{
  "objection_responses": [
    {
      "objection_id": 1,
      "objection": "the objection",
      "theme": "Price",
      "deal_killer_score": 4,
      "live_reframe": "3 sentences",
      "follow_up_email": {
        "subject": "",
        "body": ""
      },
      "email_sequence": [
        { "email_number": 1, "subject": "", "body": "", "send_delay": "Day 1" },
        { "email_number": 2, "subject": "", "body": "", "send_delay": "Day 4" },
        { "email_number": 3, "subject": "", "body": "", "send_delay": "Day 8" }
      ],
      "battle_card": {
        "objection_as_buyer_says_it": "",
        "what_they_really_mean": "",
        "best_response": "",
        "proof_point": "",
        "common_follow_up": "",
        "red_flag": ""
      },
      "website_copy": {
        "heading": "",
        "body": ""
      }
    }
  ]
}
```

```
Tone across all types: confident, not defensive. Empathetic, not soft.
Every response must include specific proof, not generic reassurance.
```

On Tone

The single biggest failure mode for AI-generated objection handling is defensiveness. Phrases like “we actually do have...” or “that’s not really the case...” signal insecurity. Strong objection handling acknowledges the concern as valid, reframes the evaluation criteria, and lets the evidence do the convincing. Add this instruction if Claude’s output sounds defensive: “Delete any sentence that starts with ‘actually,’ ‘but,’ or ‘however.’ Rewrite from a position of strength.”

3B. Generating “Why Now” Pages

Beyond individual objection responses, the Action subagent generates two complete website pages that address the most common objections preemptively.

Why Us Page Prompt

```
User Message:
Based on the objections extracted from our sales calls, generate
a complete "Why Us" page for our website.

TOP OBJECTIONS (ranked by frequency):
{{$json.ranked_objections}}

PROOF POINTS: {{$json.proof_points}}
PRODUCT: [YOUR PRODUCT DESCRIPTION]
ICP: [YOUR ICP]

Write a "Why [Company]" page that:
1. Opens with a 2-sentence value proposition that addresses the
   #1 evaluation concern for our ICP
2. Has 4-6 H2 sections, each addressing one major objection theme
   (price, trust, switching cost, etc.) but framed positively —
   not as objection responses but as value statements
3. Each section includes a customer proof point or data point
4. Ends with a CTA to book a demo or start a trial

Tone: authoritative, not salesy. This reads like a confident company
that knows its strengths, not one that’s anticipating criticism.

Output JSON:
{
  "page_title": "",
  "meta_description": "",
  "body": "full page in markdown with H2 sections"
}
```

Why Now Page Prompt

```
User Message:
Generate a "Why Now" page that addresses timing objections.

TIMING-RELATED OBJECTIONS FROM CALLS:
{{$json.timing_objections}}

Write a "Why Now" page that:
1. Opens by acknowledging that timing is always a question
2. Makes 3-4 arguments for acting now vs. waiting:
   - Cost of delay (quantified if possible)
   - Competitive risk of inaction
   - Implementation timeline and how it fits their calendar
   - Current market conditions that make this urgent
3. Each argument backed by a proof point or data
4. Ends with a low-commitment CTA (assessment, demo, trial)

Output JSON:
{
  "page_title": "",
  "meta_description": "",
  "body": "full page in markdown"
}
```

Step 4: Format and Distribute Assets

The raw JSON output from Claude needs to be formatted and distributed to the right teams in the right format. This step routes each content type to its destination.

4A. Output Routing Map

Content Type	Format	Destination	Who Uses It
Live Reframes	Google Sheet (searchable by objection theme)	"Sales Enablement/Objection Reframes" sheet	AEs during live calls
Follow-up Emails	Google Sheet (copy-pasteable templates)	"Sales Enablement/Email Templates" sheet	AEs post-call
Email Sequences	Google Sheet or CRM sequences	"Sales Enablement/Sequences" sheet or direct to CRM	AEs for stalled deals
Battle Cards	Google Doc or PDF (one doc per objection theme)	Google Drive → "Sales Enablement/Battle Cards" folder	AEs as reference
Website Copy (per-objection)	Content Calendar	"Content Calendar" sheet with type “Website — Why Us Section”	Marketing for website updates
Why Us Page	Google Doc	Google Drive → "Website Copy" folder	Marketing for website publish
Why Now Page	Google Doc	Google Drive → "Website Copy" folder	Marketing for website publish

4B. Battle Card Google Sheet Schema

Create a Google Sheet called "Objection Battle Cards" with one row per objection:

Column	Type	Source
Objection ID	Number	From Claude’s extraction (Step 2)
Objection	Text	Clean phrasing of the concern
Theme	Dropdown	Price / Timing / Trust / Switching / Buy-in / Competitor / Fit / Contract

Deal-Killer Score	Number	1–5 from extraction
Frequency	Number	How many calls it appeared in
What They Really Mean	Text	From battle card output
Live Reframe	Long text	3-sentence response for calls
Best Response (Detailed)	Long text	From battle card output
Proof Point	Text	Customer data / case study to cite
Follow-up Email Subject	Text	Ready to copy-paste
Follow-up Email Body	Long text	Ready to copy-paste
Red Flag	Text	When this signals a bad-fit deal
Last Updated	Date	Auto-populated

4C. Slack Notification

```
Slack Message Template:

🎯 *New objection-handling assets generated*

*Calls analyzed:* {{$json.total_calls_analyzed}} ({{$json.won_calls}} won,
{{$json.lost_calls}} lost)
*Objections extracted:* {{$json.objection_count}}
*Top deal-killer:* "{{$json.top_objection}}" (score: {{$json.top_score}}/5)

*Assets created:*
  • {{$json.objection_count}} live reframes
  • {{$json.objection_count}} follow-up email templates
  • {{$json.objection_count}} 3-email sequences
  • {{$json.objection_count}} battle card entries
  • 1 updated "Why Us" page
  • 1 updated "Why Now" page

Everything is in the Sales Enablement folder. Review and deploy.
```

Step 5: The Complete n8n Workflow

Workflow Architecture

```
Schedule Trigger (Bi-weekly or Monthly)
  │
  ├── HTTP Request: Call Platform API → Closed-Won transcripts
  ├── HTTP Request: Call Platform API → Closed-Lost transcripts
  │   (Or: Google Drive Trigger watching for new transcript files)
  │
  └── SplitInBatches (5-10 transcripts per batch)
        │
        └── HTTP Request: Claude API (pre-summarize transcripts)
              │
              └── Merge Node (combine all summaries)
                    │
                    └── HTTP Request: Claude API (extract + categorize objections)
                          │
                          └── Code Node: Parse JSON, rank by deal-killer score
                                │
                                ├── HTTP Request: Claude API (objection-handling content)
                                │     │
                                │     ├── Code Node: Split into content types
                                │     ├── Google Sheets: Write battle card entries
                                │     ├── Google Sheets: Write email templates
                                │     └── Google Sheets: Write reframes
                                │
                                ├── HTTP Request: Claude API (Why Us page)
                                │     └── Google Drive: Save as Doc
                                │
                                ├── HTTP Request: Claude API (Why Now page)
                                │     └── Google Drive: Save as Doc
                                │
                                └── Slack: Notify team
```

Node Count and Estimated Build Time

Metric	Value

Total n8n nodes	16–22 (depending on call platform and output destinations)
Build time (first time)	3–5 hours (plus 1–2 hours compiling proof points)
Build time (experienced)	1–2 hours
Run time per cycle	10–25 minutes (dependent on transcript volume and API speed)
Output per run	5–15 objection entries with 5 content types each, plus 2 website pages
Total assets per run	30–80 individual content pieces

Step 6: Human-in-the-Loop Review

Action content directly interfaces with live sales conversations and prospect communications. The stakes for accuracy and tone are higher here than any other subagent. Every piece needs careful review.

1. **Reframe naturalness check.** Read each live reframe out loud as if you’re on a call. Does it sound like something a human would actually say? If it sounds scripted, rewrite it. The best reframes feel spontaneous even though they’re prepared.
2. **Proof accuracy check.** Verify every customer stat, case study reference, and data point cited in emails and battle cards. Claude may match the wrong proof point to an objection, or subtly misstate a metric.
3. **Tone calibration.** Objection responses should be confident, not aggressive. Empathetic, not condescending. Read each email through the prospect’s eyes: would this make you feel understood, or lectured?
4. **Red flag validation.** Review the "red flag" field in each battle card. Does this accurately identify when the objection signals a bad-fit prospect? Sales teams waste time chasing deals they should disqualify early.
5. **Sequence logic check.** For the 3-email sequences, make sure each email builds on the previous one without repeating the same argument. The sequence should escalate: readdress → provide social proof → create urgency.
6. **Website copy alignment.** The “Why Us” and “Why Now” pages should match your website’s existing voice and style. These pages often need the most editing to integrate with your brand’s tone.

Time Investment

Plan for 30–45 minutes reviewing battle cards and reframes (read them aloud), 20–30 minutes reviewing email templates and sequences, and 15–20 minutes per website page. Total per run: 2–4 hours. This runs bi-weekly or monthly, so the time investment is manageable.

Feedback Loop: Making This Smarter Over Time

After your sales team uses the generated content for 2–4 weeks, gather feedback: Which reframes worked? Which emails got responses? Which objections kept coming up despite the new content? Feed this back into the prompts as examples of what works and what doesn’t. Over time, the quality of generated content improves because Claude has real performance data to learn from.

Troubleshooting Common Issues

Problem	Solution
Transcripts are too long for Claude's context	Use the pre-summarization prompt (Step 2B) to extract only objection-relevant segments. This reduces each transcript by 80–90%. Alternatively, process one transcript at a time and aggregate objections in a second pass.
Claude extracts too many minor objections	Add a threshold to the prompt: "Only include objections that appear in 2+ calls or that directly stalled deal progress. Ignore casual questions or curiosity-driven inquiries."
Reframes sound scripted	Add this instruction: "Write like a human having a conversation, not a sales training manual. Use contractions. Start with different words each time. Vary sentence length." Include 2–3 examples of reframes your best reps actually use.
Email content is too generic	Your proof points document is too thin. Add more specific customer results with numbers, timelines, and use cases. The more specific the proof, the more specific (and persuasive) the email.
Gong API rate limits	Gong's API has rate limits (varies by plan). Add Wait nodes (5–10 seconds) between transcript requests. Process in smaller batches if needed. Gong also offers webhooks — consider using those instead of polling.
Battle cards aren't being used by sales	The format matters. If reps won't open a Google Sheet during a call, convert battle cards to Slack-searchable format: create a Slack channel where each objection is a pinned message with the reframe. Reps can search during calls.
Same objections repeat across cycles	This is expected and good — it means the data is consistent. Use the repeat data to validate that your objection list is accurate. Update the content only when the handling changes or new proof points become available.
Lost deal reasons don't match extracted objections	CRM lost-deal reasons are often sanitized ("timing" when it was really "price"). Trust the transcript data over the CRM field. Consider adding a post-call objection tag that reps fill in immediately.

Series Complete: The Full Content Engine

With all four subagents built, you now have a complete automated content engine that covers every stage of the buyer journey:

Subagent	Stage	What It Does	Schedule
1: Awareness	Top of funnel	Scrapes buyer questions, generates blog posts, carousels, and video scripts	Weekly
2: Interest	Mid-funnel	Detects trending topics, generates POV content and hot takes	Daily
3: Desire	Bottom of funnel	Scrapes competitors, generates comparison pages, ROI calculators, and one-pagers	Monthly
4: Action	Deal stage	Extracts call objections, generates battle cards, emails, and website copy	Bi-weekly

Each subagent runs independently on its own schedule. Together, they produce 50–100+ content pieces per month across every format your marketing and sales teams need — with human review as the quality gate between generation and publication.

The system gets smarter over time. As you edit Claude’s output, note patterns in what you change. Feed those patterns back into the prompts as examples and constraints. After 2–3 cycles, the raw output quality improves significantly because the prompts are calibrated to your voice, your market, and your buyers’ actual concerns.

The 80/20 of Implementation

If you’re building this from scratch, start with Subagent 1 (Awareness). It’s the simplest to build, produces the highest volume of content, and teaches you the n8n + Claude workflow patterns you’ll reuse across all four subagents. Once Subagent 1 is running, add the others one at a time, in order. Full engine build time: 2–4 weeks of part-time work.

End of The Content Engine Series • 4 of 4