THE CONTENT ENGINE

SUBAGENT 3

Desire Content

Automatically scrape competitor websites — pricing pages, feature lists, case studies, and testimonials — then generate comparison pages, ROI calculators, one-pagers, and bottom-of-funnel blog posts that make the business case before your buyer asks for it.

A step-by-step technical implementation guide.

Part 3 of 4 • The Content Engine Series

What This Subagent Does

The Desire subagent builds the business case for your product before a prospect ever asks for it. It scrapes your competitors' public-facing content — pricing, features, positioning, testimonials — and generates the comparison assets, ROI frameworks, and evaluation content that your buyer would otherwise have to build themselves.

Most buyers are 60–70% through their evaluation before they talk to sales. This subagent ensures that when they do that evaluation, your content is doing the framing. You control the criteria, the comparison narrative, and the ROI math.

The Workflow at a Glance

1. Firecrawl scrapes competitor pricing pages, feature lists, case studies, and testimonials
2. Claude structures the raw data into standardized competitor profiles
3. Claude generates comparison pages, ROI calculators, one-pagers, and BoFu blog posts
4. n8n routes outputs to Google Drive and notifies your sales and marketing teams
5. You review, approve, and publish to your website and sales enablement library

Why This Matters

Comparison content is the highest-converting content type in B2B SaaS. Prospects searching "[Your Product] vs [Competitor]" are at the bottom of the funnel with active purchase intent. If you don't own that page, your competitor does — or a third-party review site controls the narrative entirely.

What You Need Before Starting

Required Accounts & API Keys

Tool	Purpose	Notes
n8n	Workflow orchestration	Same instance as Subagents 1 and 2.
Claude API	Data structuring + content generation	Anthropic API key. Opus recommended for nuanced comparison writing.
Firecrawl	Competitor website scraping	Scrapes pricing, features, case studies. firecrawl.dev
Google Sheets	Competitor data warehouse	Stores structured competitor profiles for ongoing reference.

Google Drive	Output storage	Where finished comparison pages, one-pagers, and calculators are saved.
Slack (optional)	Team notifications	Alerts sales when new competitive assets are ready.

Prerequisite: Document Your Own Product Data

Before you can generate comparison content, you need your own product data structured and ready. Claude needs both sides of the comparison. Prepare the following:

1. **Feature list:** Every feature, organized by category. Include what’s included in each pricing tier.
2. **Pricing:** All plans, pricing, and what’s included. Note any usage-based components, setup fees, or contract requirements.
3. **Value metrics:** The 3–5 measurable outcomes your product delivers. These become the inputs for your ROI calculator. Examples: hours saved per week, reduction in error rate, increase in conversion rate, decrease in time-to-value.
4. **Customer proof points:** Specific results from real customers. Ideal format: "[Customer type] achieved [specific metric] in [timeframe]." You need 3–5 of these minimum.
5. **Differentiators:** The 2–3 things you do that competitors genuinely cannot match. Not features — capabilities or approaches that are structurally different.

Store This as a Reference Document

Create a Google Doc or JSON file with all of this data. You’ll reference it in every prompt. When your product changes (new features, pricing updates), update this document and re-run the subagent to regenerate all comparison assets automatically.

Step 1: Identify and Prioritize Competitors

You don’t need comparison pages for every competitor. Start with the 3–5 competitors that come up most often in your sales conversations — the ones your prospects are actively evaluating alongside you.

1A. How to Pick Your Competitors

1. **Ask your sales team:** "Which competitors come up most in deals?" and "Which competitors do we lose to most often?" These are your top priority.
2. **Check your CRM:** Filter closed-lost deals by the “competitor” field (if you track this). Sort by frequency.
3. **Search Google:** Search "[your product] vs" and see what autocomplete suggests. These are the comparisons your prospects are already making.
4. **Check G2/Capterra:** Look at your category page. The products with the most reviews in your segment are likely the ones your buyers are evaluating.

1B. Create a Competitor Registry

For each competitor, create a row in a Google Sheet with the following information. This becomes the input for the scraping step.

Column	Example	Notes
Competitor Name	Acme Corp	Display name for content
Domain	acme.com	Used for Firecrawl scraping
Pricing URL	/pricing	Path to pricing page
Features URL	/features or /product	Path to feature list
Case Studies URL	/customers or /case-studies	Path to customer stories
About/Positioning URL	/about or homepage	Where they describe who they serve
Priority	1 (highest) to 5	Based on frequency in sales conversations
Last Scraped	Date	Tracks when data was last refreshed

Step 2: Scrape Competitor Data

This step uses Firecrawl to extract content from each competitor's public web pages. You'll scrape four page types per competitor: pricing, features, case studies, and positioning.

2A. Firecrawl Scraping Workflow

In n8n, this runs as a loop: for each competitor in your registry, scrape all four page types and store the results.

n8n Workflow Nodes

1. **Schedule Trigger** — Set to run monthly. Competitor pages don't change daily, so monthly scraping keeps your data current without wasting API calls.
2. **Google Sheets Node (Read)** — Pull all rows from your Competitor Registry sheet.
3. **SplitInBatches Node** — Process one competitor at a time.
4. **HTTP Request Nodes (Firecrawl)** — One request per page type, per competitor.

Firecrawl API Call — Pricing Page

```
POST https://api.firecrawl.dev/v1/scrape

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

Body:
{
  "url": "https://{{$json.domain}}{{$json.pricing_url}}",
  "formats": ["markdown"],
  "onlyMainContent": true
}
```

Repeat this pattern for each page type, substituting the URL path. The response contains the page content in markdown format, which Claude can parse directly.

Firecrawl API Call — Features Page

```
Body:
{
  "url": "https://{{$json.domain}}{{$json.features_url}}",
  "formats": ["markdown"],
  "onlyMainContent": true
}
```

Firecrawl API Call — Case Studies Page

```
Body:
{
  "url": "https://{{$json.domain}}{{$json.case_studies_url}}",
  "formats": ["markdown"],
  "onlyMainContent": true
}
```

Firecrawl API Call — Positioning Page

```
Body:
{
  "url": "https://{{$json.domain}}{{$json.about_url}}",
  "formats": ["markdown"],
  "onlyMainContent": true
}
```

Handling Scraping Failures

Some pages will fail to scrape — JavaScript-heavy SPAs, pages behind authentication, or sites with aggressive bot protection. Add error handling in n8n: use an IF node after each Firecrawl call to check if the response contains content. If it's empty, log the failure and continue to the next page type. Don't let one failed scrape block the entire competitor.

2B. Alternative: Firecrawl Crawl Endpoint

If a competitor's information is spread across many pages (e.g., individual feature pages, multiple case studies), use Firecrawl's crawl endpoint instead of scrape. This follows links and returns content from multiple pages.

```
POST https://api.firecrawl.dev/v1/crawl

Body:
{
  "url": "https://{{$json.domain}}",
  "limit": 20,
  "includePaths": ["/pricing*", "/features*", "/customers*", "/product*"],
  "excludePaths": ["/blog*", "/careers*", "/legal*"],
  "formats": ["markdown"]
}
```

The crawl endpoint is asynchronous — it returns a job ID, and you poll for results. Use an n8n Wait node (30–60 seconds) followed by a GET request to check job status. The crawl may take 1–5 minutes depending on the number of pages.

Step 3: Structure Competitor Data with Claude

Raw scraped content is messy — full of marketing copy, navigation elements, and inconsistent formatting. This step sends the raw data to Claude to extract and structure it into a standardized competitor profile.

3A. The Data Structuring Prompt

```
System Prompt:
You are a competitive intelligence analyst. Your job is to extract
structured product data from raw website content.

User Message:
Here is scraped content from {{$json.competitor_name}}’s website.

PRICING PAGE:
{{$json.pricing_content}}

FEATURES PAGE:
{{$json.features_content}}

CASE STUDIES PAGE:
{{$json.case_studies_content}}

POSITIONING PAGE:
{{$json.positioning_content}}

Extract and structure the following into JSON:

{
  "competitor_name": "",
  "tagline": "their one-line positioning",
  "target_audience": "who they say they serve",
  "pricing": {
    "model": "per-seat / usage-based / flat rate / etc.",
    "plans": [
      {
        "name": "plan name",
        "price": "$X/mo or $X/yr",
        "features_included": ["feature 1", "feature 2"],
        "limitations": ["limit 1", "limit 2"]
      }
    ],
    "has_free_tier": true/false,
    "has_enterprise_plan": true/false,
    "notes": "any caveats, hidden costs, or unusual pricing structures"
  },
  "features": {
    "categories": [
```

```
      {
        "category": "Feature Category Name",
        "features": ["feature 1", "feature 2"]
      }
    ]
  },
  "customer_proof": [
    {
      "customer_type": "industry/size/role",
      "result": "specific outcome mentioned",
      "quote": "exact quote if available"
    }
  ],
  "strengths": ["what they appear to do well"],
  "weaknesses": ["gaps, missing features, or complaints implied"],
  "positioning_summary": "2-3 sentences on how they position themselves"
}

Be factual. Only include what you can verify from the scraped content.
If information is unavailable, use null. Return valid JSON only.
```

3B. Store Structured Profiles

After Claude returns the structured JSON, store each competitor profile in a dedicated Google Sheet (one tab per competitor) or as a JSON file in Google Drive. This becomes your competitive intelligence database that the content generation step draws from.

```
// n8n Code Node — Parse and prepare for storage
const response = $input.first().json;
const text = response.content[0].text;
const clean = text.replace(/```json|```/g, '').trim();
const profile = JSON.parse(clean);

// Flatten for Google Sheets or keep as JSON for Drive
return [{
  json: {
    competitor_name: profile.competitor_name,
    profile_json: JSON.stringify(profile),
    last_updated: new Date().toISOString(),
    tagline: profile.tagline,
    target_audience: profile.target_audience,
    plan_count: profile.pricing?.plans?.length || 0,
    feature_count: profile.features?.categories?.reduce(
      (sum, c) => sum + c.features.length, 0) || 0
  }
}];
```

Keep This Data Fresh

Set the monthly schedule trigger to re-scrape all competitors and regenerate profiles. The structured profiles feed directly into content generation — when a competitor changes pricing or adds features, your comparison content updates automatically on the next run.

Step 4: Generate Comparison Pages

This is the highest-value output of the subagent. Comparison pages capture bottom-of-funnel search traffic and control the evaluation narrative. You'll generate one comparison page per competitor.

4A. The Comparison Page Prompt

```
System Prompt:
You are a conversion copywriter for a B2B SaaS company.
You write comparison pages that are fair, factual, and persuasive.
You never trash competitors — you win on the dimensions that
matter most to your ICP.

User Message:
Generate a comparison page: Us vs. {{$json.competitor_name}}

OUR PRODUCT DATA:
{{$json.our_product_data}}

COMPETITOR PROFILE:
{{$json.competitor_profile}}

OUR ICP:
[YOUR IDEAL CUSTOMER PROFILE]

ICP'S TOP EVALUATION CRITERIA (ranked by importance):
1. [CRITERION 1 — e.g., ease of implementation]
2. [CRITERION 2 — e.g., depth of reporting]
3. [CRITERION 3 — e.g., pricing transparency]
4. [CRITERION 4 — e.g., customer support quality]
5. [CRITERION 5 — e.g., integration ecosystem]

Write a comparison page with these exact H2 sections:

H2: [Us] vs [Competitor]: An Honest Comparison
  Opening paragraph that acknowledges both products serve the
  market well, then frames the comparison around what matters
  most for [ICP]. 2-3 sentences.

H2: Feature Comparison
  A markdown table comparing features across the top 8-10
  capabilities your ICP cares about. Use checkmarks (✓),
  X marks (✗), and "Partial" where appropriate.
  Below the table, write 2-3 sentences highlighting where
  the meaningful differences are.

H2: Pricing Comparison
```

```
  Side-by-side pricing overview. Be transparent about both.
  Highlight total cost of ownership, not just sticker price.
  Note hidden costs (setup fees, overages, add-ons) where they exist.

H2: Where [Competitor] Wins
  Be honest — list 1-2 areas where the competitor is genuinely
  better or where they're a better fit. This builds trust with
  the reader and makes your other claims more credible.

H2: Where We Win
  List 3-4 areas where your product is better, tied directly
  to ICP evaluation criteria. Each point should reference a
  specific customer result or capability, not just a feature.

H2: Who Should Choose [Competitor]
  Describe the buyer profile that's a better fit for them.
  This disqualifies bad-fit prospects and builds credibility.

H2: Who Should Choose [Us]
  Describe the buyer profile that's a perfect fit for you.
  Be specific about company size, use case, and priorities.

H2: What Our Customers Say
  Include 2-3 customer proof points from your data.
  Format as brief case snippets with specific results.

H2: Ready to See [Us] in Action?
  2-sentence CTA to book a demo. Confident, not pushy.

Output JSON:
{
  "page_title": "[Us] vs [Competitor]: Honest Comparison for [Year]",
  "meta_description": "under 155 chars, comparison-focused",
  "slug": "/compare/us-vs-competitor",
  "body": "full page copy in markdown with H2 sections",
  "word_count": 1200
}
```

On Fairness

The "Where [Competitor] Wins" section is not optional. Readers can tell when a comparison page is one-sided marketing. A genuinely fair comparison converts better because it earns trust. If your product wins on the dimensions that matter most to your ICP, you don't need to pretend you win on everything.

Step 5: Generate ROI Calculators

ROI calculators do the math that justifies the purchase. They translate your value metrics into dollars and hours saved, giving your champion the business case they need for internal buy-in.

5A. The ROI Calculator Spec Prompt

Claude generates the specification for a calculator, not the calculator itself. You’ll build the interactive version from the spec. This keeps the prompt focused on the logic while letting you choose the implementation (spreadsheet, web page, or embedded tool).

```
System Prompt:
You are a business analyst who builds ROI calculators for B2B SaaS
products. You focus on measurable outcomes, conservative estimates,
and clear assumptions.

User Message:
Build an ROI calculator specification for our product.

OUR VALUE METRICS:
{{$json.value_metrics}}

OUR PRICING:
{{$json.our_pricing}}

CUSTOMER PROOF POINTS:
{{$json.customer_results}}

ICP: [YOUR ICP]

Create an ROI calculator spec with:

1. INPUT FIELDS (what the prospect fills in):
   - List each input with: label, description, default value,
     and data type (number, currency, percentage)
   - Use 5-8 inputs maximum (more = less completion)
   - Include current cost/time inputs that establish the baseline

2. CALCULATION LOGIC:
   - For each value metric, show the formula in plain language
   - Use conservative multipliers (cite the customer data they’re based on)
   - Calculate: annual time saved, annual cost saved, annual revenue impact

3. OUTPUT DISPLAY:
   - Total annual ROI (as dollar amount and percentage)
   - Payback period (months)
   - Breakdown by value metric
```

```
   - One-line "headline result" (e.g., "You'd save $47,000/yr")

4. ASSUMPTIONS FOOTNOTE:
   - List every assumption and its source
   - Show which customer data point each estimate is based on

Output JSON:
{
  "calculator_title": "",
  "inputs": [
    {
      "label": "",
      "description": "",
      "default_value": 0,
      "type": "number|currency|percentage",
      "unit": "hours|dollars|people|etc."
    }
  ],
  "formulas": [
    {
      "metric": "",
      "formula_plain": "plain English formula",
      "formula_code": "JavaScript expression",
      "source": "which customer data this is based on"
    }
  ],
  "outputs": [
    {
      "label": "",
      "formula_code": "JavaScript expression",
      "format": "currency|percentage|months|number"
    }
  ],
  "assumptions": ["assumption 1", "assumption 2"]
}
```

5B. Building the Calculator

Once you have the spec, build the interactive version using one of these approaches:

Approach	Best For	Implementation
Google Sheet Template	Sales team internal use	Create a template sheet with input cells and formula cells. Sales clones it per prospect and fills in their numbers.
React/HTML Component	Website embedding	Use the JavaScript formulas from the spec to build an interactive calculator. Embed on your pricing or ROI page.

Notion/Coda Calculator	Quick internal builds	Coda and Notion support formula fields. Build a shareable calculator page from the spec.
Spreadsheet + PDF	Leave-behind for prospects	Sales fills in the inputs, exports as PDF, and sends to the prospect's champion for internal circulation.

Pro Tip: Generate the React Component with Claude

You can take the calculator spec JSON and ask Claude to generate a complete React component or single-file HTML page with the calculator UI, input fields, and live calculations. This gives you a working prototype you can embed on your website immediately.

Step 6: Generate One-Pagers and BoFu Blog Posts

Beyond comparison pages and calculators, this subagent produces two additional content types that sales teams need: one-pagers for prospect handoffs, and bottom-of-funnel blog posts that capture evaluation-stage search traffic.

6A. One-Pager Prompt

One-pagers are single-page PDF documents that a prospect's champion can forward internally. They should communicate your value proposition in under 60 seconds of reading.

```
User Message:
Create a one-pager for prospects evaluating us against {{$json.competitor_name}}.

OUR PRODUCT DATA: {{$json.our_product_data}}
COMPETITOR PROFILE: {{$json.competitor_profile}}
ICP: [YOUR ICP]

The one-pager must fit on a single page and include:

1. HEADLINE: One sentence that frames why we're the better choice
   for [ICP]. Bold, specific, no jargon.

2. THE 3 REASONS: Three bullet points, each with a bolded label
   and 1-2 sentences of supporting detail. Tied to ICP evaluation
   criteria. Each includes a proof point.

3. QUICK COMPARISON: A simple 5-row comparison table showing the
   dimensions that matter most. Use ✓, ✗, and brief text.

4. CUSTOMER RESULT: One specific case study result in 2 sentences.

5. CTA: One line with next step (book a demo, start a trial, etc.)

Output JSON:
{
  "headline": "",
  "reasons": [
    { "label": "", "detail": "", "proof": "" }
  ],
  "comparison_table": {
    "rows": [
      { "dimension": "", "us": "", "them": "" }
    ]
  },
  "customer_result": "",
  "cta": ""
```

```
}
```

6B. Bottom-of-Funnel Blog Post Prompt

BoFu blog posts target searches like "[Product A] vs [Product B]," "best [category] for [use case]," and "[Competitor] alternatives." These are high-intent keywords with direct purchase consideration.

```
User Message:
Write a bottom-of-funnel blog post targeting the keyword:
"{{$json.competitor_name}} alternatives"

OUR PRODUCT DATA: {{$json.our_product_data}}
COMPETITOR PROFILE: {{$json.competitor_profile}}
ICP: [YOUR ICP]

Write an 800-word blog post that:

1. Opens with empathy for the reader’s evaluation process
   (they’re researching because something isn’t working or they
   need to make a choice)
2. Briefly describes what [Competitor] does well (2-3 sentences)
3. Identifies 3-4 reasons someone might look for an alternative
   (based on gaps in their profile: pricing, features, support, etc.)
4. Introduces your product as one alternative — not the only one
5. For each gap, explains how you address it with specific detail
6. Includes a comparison table (5-6 key dimensions)
7. Ends with clear next steps for the reader

Tone: helpful evaluator, not aggressive competitor.
This is a blog post, not a sales page. Educate first, sell second.

Output JSON:
{
  "meta_title": "under 60 chars, includes target keyword",
  "meta_description": "under 155 chars",
  "target_keyword": "",
  "secondary_keywords": ["kw1", "kw2"],
  "body": "full 800-word post in markdown",
  "word_count": 800
}
```

Step 7: Route Outputs and Notify

All generated content routes to a structured storage system where marketing and sales can access it. Different content types go to different destinations.

7A. Output Routing

Content Type	Destination	Who Uses It
Comparison Pages	Google Drive → "Competitive/Comparison Pages" folder + Content Calendar	Marketing (publishes to website)
ROI Calculator Specs	Google Drive → "Sales Tools/ROI Calculators" folder	Marketing (builds interactive version) + Sales (uses spreadsheet version)
One-Pagers	Google Drive → "Sales Enablement/One-Pagers" folder	Sales (sends to prospects)
BoFu Blog Posts	Content Calendar (Google Sheets) with status "Draft"	Marketing (publishes to blog)
Competitor Profiles	Google Sheets → "Competitive Intel" tab	Sales + Marketing (reference)

7B. Google Sheets Content Calendar Columns

Add these columns to your existing content calendar (or create a new tab) for Desire-stage content:

Column	Type	Description
Date Created	Date	Auto-populated by n8n
Content Type	Dropdown	Comparison Page / ROI Calculator / One-Pager / BoFu Blog
Competitor	Text	Which competitor this content is about
Target Keyword	Text	Primary SEO keyword (for blog posts)
Draft	Long text or Link	The content itself or a Google Drive link
Status	Dropdown	Draft / In Review / Approved / Published
Last Updated	Date	When the content was last regenerated
Assigned To	Text	Who's reviewing this piece

7C. Slack Notification

```
Slack Message Template:

🎯 *New competitive content generated*

*Competitor:* {{$json.competitor_name}}
*Content types created:*
  • Comparison page ({{$json.comparison_word_count}} words)
  • ROI calculator spec
  • One-pager
  • BoFu blog post ("{{$json.blog_meta_title}}")

All drafts are in Google Drive. Review and publish.
```

Step 8: The Complete n8n Workflow

Workflow Architecture

```
Schedule Trigger (Monthly)
  │
  └── Google Sheets: Read Competitor Registry
        │
        └── SplitInBatches (one competitor at a time)
              │
              ├── HTTP Request: Firecrawl → Pricing page
              ├── HTTP Request: Firecrawl → Features page
              ├── HTTP Request: Firecrawl → Case studies page
              ├── HTTP Request: Firecrawl → Positioning page
              │
              └── Merge Node (combine all scraped data)
                    │
                    └── HTTP Request: Claude API (structure competitor profile)
                          │
                          ├── Google Sheets: Store structured profile
                          │
                          ├── HTTP Request: Claude API (comparison page)
                          │     └── Google Drive: Save comparison page
                          │
                          ├── HTTP Request: Claude API (ROI calculator spec)
                          │     └── Google Drive: Save calculator spec
                          │
                          ├── HTTP Request: Claude API (one-pager)
                          │     └── Google Drive: Save one-pager
                          │
                          ├── HTTP Request: Claude API (BoFu blog post)
                          │     └── Google Sheets: Append to content calendar
                          │
                          └── Slack: Notify team
```

Node Count and Estimated Build Time

Metric	Value

Total n8n nodes	18–24 (depending on competitor count and output types)
Build time (first time)	3–5 hours (plus 1–2 hours documenting your own product data)
Build time (experienced)	1–2 hours
Monthly run time	15–30 minutes per competitor (API dependent)
Output per competitor	4 content pieces (comparison page, calculator, one-pager, blog post)
Output per run (5 competitors)	20 content pieces + 5 structured profiles

Step 9: Human-in-the-Loop Review

Competitive content carries the highest accuracy risk of any subagent. Incorrect claims about a competitor's pricing or features can damage your credibility and potentially create legal issues. Every piece requires careful review.

1. **Accuracy check (critical).** Visit each competitor's actual website and verify every claim in the comparison page. Check pricing numbers, feature availability, and plan limitations. If Firecrawl scraped stale data, update it manually.
2. **Fairness check.** Read the comparison as if you were a prospect. Does it feel balanced, or does it read like an attack piece? The "Where They Win" section should be genuinely honest. If it feels like a throwaway concession, rewrite it.
3. **Differentiation check.** Are your "where we win" points actually differentiated, or are they things most products in the category can claim? "Easy to use" is not a differentiator. "Deploys in 15 minutes with no IT involvement" is.
4. **ROI check.** Verify the calculator's assumptions. Are the multipliers conservative enough to be credible? A prospect who runs the calculator and gets an unbelievable number will dismiss it. Better to under-promise.
5. **Legal check.** Don't make claims you can't substantiate. Avoid language like "they can't do X" — use "based on their published features, X is not listed." Never reference internal competitive intel that isn't publicly available.
6. **CTA check.** The demo CTA should feel like a natural next step, not a hard sell. The prospect has just read a thorough evaluation — they're ready if the content did its job.

> **Time Investment**
>
> Plan for 20–30 minutes per comparison page (the accuracy check takes the most time), 10–15 minutes per one-pager, 15–20 minutes per blog post, and 15–20 minutes per ROI calculator spec. For a full run of 5 competitors, that's roughly 4–6 hours of review work. This is a monthly investment, not weekly.

Troubleshooting Common Issues

Problem	Solution
Firecrawl returns empty or partial data	Some sites use heavy JavaScript rendering. Try the crawl endpoint instead of scrape, or add a "waitFor" parameter (Firecrawl supports waiting for JS to render). If a page is truly blocked, manually copy the content and paste it into your competitor registry sheet.
Claude's comparison is too one-sided	Strengthen the "Where They Win" section in your prompt. Add: "A skeptical reader should feel this comparison is fair. If every point favors us, the content loses credibility."
Pricing data is behind a "Contact Sales" wall	Note this honestly in the comparison: "[Competitor] requires a sales conversation for pricing. Based on publicly available data and customer reports..." Don't fabricate pricing.
ROI calculator numbers seem too high	Reduce multipliers by 25–50%. Add a "conservative estimate" toggle that uses lower assumptions. It's better to show a credible $30k savings than an unbelievable $200k.
One-pager content is too long	Add a strict constraint: "Total word count under 300 words. Every sentence must earn its place. If it doesn't help the prospect's champion sell internally, cut it."
Competitor recently changed their product	This is why the monthly schedule matters. If you learn of a change mid-cycle, manually update the competitor profile JSON and re-run only the content generation steps (skip scraping).
Blog posts don't rank for comparison keywords	Comparison keywords are competitive. Make sure the meta title includes the exact search phrase ("[Product] vs [Competitor]"). Add internal links from other content. Consider running both a comparison page (on a /compare/ URL) and a blog post (targeting "alternatives") for maximum coverage.

What's Next

This subagent automates the bottom-of-funnel content that typically takes weeks of manual competitive research. Combined with Subagents 1 (Awareness) and 2 (Interest), you now have the top, middle, and bottom of your content funnel generating content automatically.

In Subagent 4 (Action Content), you'll build the final layer: extracting real objections from sales call transcripts and generating objection-handling emails, battle cards, and "why us" pages that neutralize the specific concerns killing your deals.

Up next: Subagent 4 — Action Content (Sales Objections → Deal-Closing Assets)