Corvid Cloud support AI: where the LLM bill goes, and the six changes that cut it

$23,117 → $531 a month

97.7% lower at the list prices of the models we ran · quality: 76 of 81 checks pass after (78 of 81 before)

COMPANY	WORKLOAD	PRICES	PREPARED BY
Corvid Cloud	81 real model calls, scaled to 1,200,000 a month (assumption)	Public list prices read 11 Nov 2026	KH Labs

Executive summary

At 1,200,000 requests a month the support assistant costs $23,117. Six changes bring it to $531, a saving of $22,586 a month. Quality: 76 of 81 checks pass (78 before); every miss is listed. Three changes do most of the work: trim the prompt and retrieved context ($13,410), cap output and reasoning ($5,000) and route easy calls to a cheaper model ($3,488). Each call is priced at the list price of the model that served it.

LEVER	SAVES / MONTH	OF THE BILL	BILL AFTER
1. Trim the prompt and retrieved context	$13,410	58.0%	$9,707
2. Cap output and reasoning	$5,000	21.6%	$4,707
3. Route easy calls to a cheaper model	$3,488	15.1%	$1,219
4. Prompt caching	$639	2.8%	$580
5. Dedupe repeated calls	$26	0.1%	$554
6. Batch the nightly job	$23	0.1%	$531
Total	**$22,586**	**97.7%**	**$531**

Savings in the order they were applied. Each is measured on what the previous levers left.

Cost per request falls from $0.019264 to $0.000443. At that rate the $1,900 fee is covered after about 100,950 requests.

Quality: 76 of 81 checks pass after, 78 of 81 before. The 5 remaining failures are listed in full; 2 of them also fail on the baseline.

Baseline shape. The baseline is a typical unoptimized stack: one big reasoning model on every call, a 3,073-token system prompt, 10 retrieved chunks, no output caps. A leaner stack starts from a lower bill; the free AI

check (a one-page report within 48 hours) sizes the saving to yours before you pay.

Baseline: what the bill is made of

On your bill the baseline is your last full invoice month, reconciled line by line against a usage export, and we report the variance. Here the baseline is the API `usage` field of each of the 81 baseline calls, priced at Claude Sonnet 5.5 list prices ($2.00 per million input tokens, $10.00 per million output). The table re-adds those calls and checks the sum against the analysis.

TASK	CALLS	INPUT TOKENS	OUTPUT TOKENS	COST, SAMPLE	PER MONTH
Support answers (from the help docs)	37	161,440	24,550	$0.5684	$8,420
Ticket classifier	24	76,978	15,532	$0.3093	$4,582
Ticket summaries	12	39,973	12,979	$0.2097	$3,107
Nightly account digest	8	32,800	40,739	$0.4730	$7,007
Total	**81**	**311,191**	**93,800**	**$1.5604**	**$23,117**

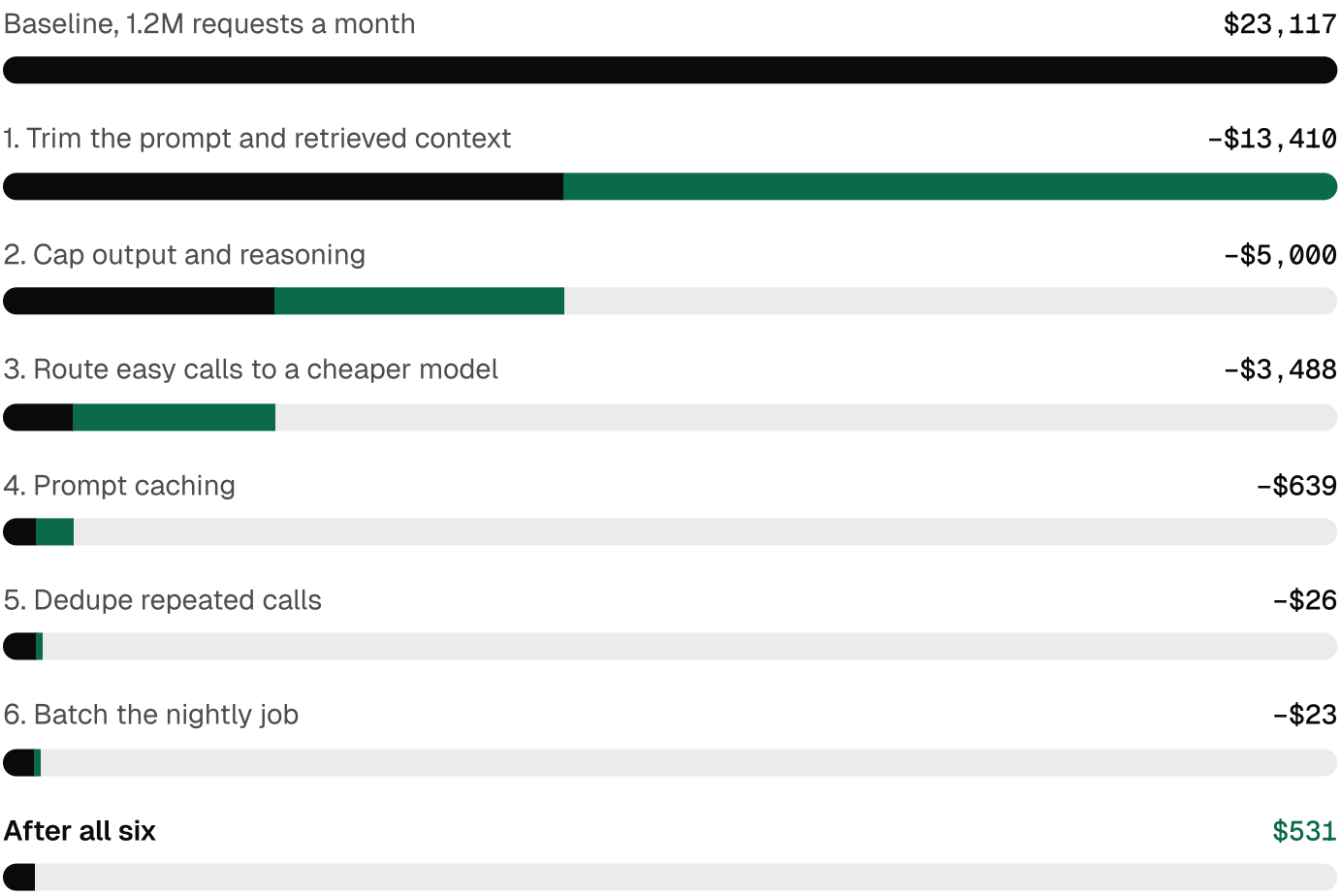
Baseline calls re-added from the run log

- `Re-added total $23,116.77 against the analysis figure $23,116.77: variance $0.00.`

Per month = sample cost × 1,200,000 / 81. We assume the 81-request mix repeats at 1.2M requests a month; about 10 percent of that volume is per-account nightly digests. The nightly digest is 8 of 81 calls but 30% of the cost: its average output was 5,092 tokens.

Savings by lever

Levers were applied one after another on the same 81 requests. Each section shows the money, one measured request before and after (cost per call at the list price of the model that served it), the code change that makes it, and how to undo it.



Monthly bill after each lever, at the list prices of the models we ran. Black is what is still paid; green is what that lever saved.

1 Trim the prompt and retrieved context

$13,410 / month
58.0% of the original bill

A shorter system prompt with every rule still in it, and the 3 best-matching knowledge-base passages instead of 10. Bill before this lever $23,117, after $9,707 a month.

Measured on request R074: “How do I reset 2FA for a teammate who lost their phone?”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	System prompt 3,073 tokens + 10 retrieved passages `Claude Sonnet 5.5`	4,371	481	$13.55
After	System prompt 1,792 tokens (same rules, no filler) + 3 re-ranked passages `Claude Sonnet 5.5`	2,266	341	$7.94

● **Change 1 · Trim the system prompt and re-rank context** `support/answer.py · sketch`

```
   def answer(question):
-      chunks = search(question, top_k=10)
-      system = SYSTEM_PROMPT        # 3,073 tokens
+      chunks = rerank(search(question, top_k=20))[:3]
+      system = POLICY_PACK + RULES  # 1,792 tokens, same rules
       return llm.chat(BIG, system, render(chunks, question))
```

In this run: the shorter prompt keeps every rule and drops the filler; retrieval re-ranks 20 candidate passages and keeps the best 3 instead of the first 10.

Risk. What to watch: a needed passage drops out of the shorter context. The top 3 passages held every passage an answer needed for 88.9% of questions (94.4% with the first 10); the quality check catches the rest. **Rollback.** Feature flag per prompt version; revert restores the old prompt and top-k in one deploy.

2 Cap output and reasoning

$5,000 / month
21.6% of the original bill

Shorter answer formats, a length limit per task, and less hidden reasoning on the big model. Bill before this lever $9,707, after $4,707 a month.

Measured on request R060: “Lost my phone, locked out of the owner account because of 2FA.”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	Asked to explain its reasoning, then give a label `Claude Sonnet 5.5`	486	424	$5.21
After	One label, max 12 tokens, low reasoning budget `Claude Sonnet 5.5`	454	5	$0.96

● **Change 2 · Cap output and reasoning** `llm/client.py · sketch`

```
+ MAX_TOKENS = {"qa": 320, "classify": 12, "summarize": 220, "report": 800}
  def chat(task, model, system, user):
-     return create(model, system, user)
+     return create(model, system, user,
+                   max_tokens=MAX_TOKENS[task],
+                   reasoning="low" if model == BIG else None,
+                   format=TERSE[task])  # e.g. one label
```

In this run: a length limit per task, a terse answer format, and the big model's reasoning set to low from this stage on.

Risk. Truncated answers. In this run one big-model digest hit its 800-token cap at the caps stage (that stage was not shipped). **Rollback.** Caps are per task in one dict; raise or remove a cap without other changes.

---

3 Route easy calls to a cheaper model

$3,488 / month
15.1% of the original bill

Classifier, summaries and nightly digests go to the small model; routine support questions too. Doubtful answers are re-run on the big model. Bill before this lever $4,707, after $1,219 a month.

Measured on request R077: “Ticket T-7105 | Customer: Aisha (Business)”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	Big model: Claude Sonnet 5.5	290	126	$1.84
After	Small model: Gemini 3.1 Flash-Lite	184	52	$0.12

Change 3 · Route easy calls to the small model, escalate doubtful ones `llm/router.py · sketch`

```
+ SENSITIVE = ["refund", "delete", "deletion", "gdpr", "privacy", "security", ...]
+ def route(task, question):
+     if task != "qa": return SMALL
+     if any(k in question.lower() for k in SENSITIVE):
+         return BIG
+     if needs_arithmetic(question): return BIG
+     return SMALL
+
+ def answer(question):
+     out = chat(route("qa", question), ...)
+     if doubtful(out, retrieved):  # unsure, no or bad cite
+         out = chat(BIG, ...)      # re-run on the big model
+     return out
```

In this run: classifier, summaries and digests go to the small model; questions with sums or sensitive words go to the big one. The doubt rule re-ran 4 answers on the big model.

Risk. The small model misses. The escalation rule re-ran 4 answers on the big model; 7 numeric or sensitive questions went straight to it. **Rollback.** Router returns BIG for everything when the flag is off.

---

4 Prompt caching

$639 / month
2.8% of the original bill

The part of every prompt that never changes (policies, label definitions) is billed at the provider's cheaper cached rate. Bill before this lever $1,219, after $580 a month.

Applied alone to the original bill it would save $6,564 a month; it reads small here because earlier levers already removed most of the tokens it acts on.

Measured on request R074: “How do I reset 2FA for a teammate who lost their phone?”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	Every token billed at the normal input rate `Gemini 3.1 Flash-Lite`	1,527	48	$0.45
After	1,172 of 1,527 prompt tokens are the stable prefix, billed at the cached rate `Gemini 3.1 Flash-Lite`	1,527	48	$0.19

● **Change 4 · Cache the stable prefix** `llm/client.py · sketch`

```
-       system=system,
+       system=cached(system),  # policies + label definitions
+       # the cached prefix must not change between calls
```

Priced on the measured prefix sizes; the 95% hit rate and the 1,024-token minimum are assumptions.

Risk. A lower hit rate than the 95% assumed (cache expiry, quiet hours); a prefix that changes per call caches nothing. **Rollback.** Turn caching off; prices fall back to normal input rates, answers unchanged.

---

5 Dedupe repeated calls

$26 / month
0.1% of the original bill

A repeated or reworded question is answered from a response cache, without a model call. Bill before this lever $580, after $554 a month.

Applied alone to the original bill it would save $1,718 a month; it reads small here because earlier levers already removed most of the tokens it acts on.

Measured on request R026: “What happens if my card payment fails?”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	A model call for a question we already answered `Gemini 3.1 Flash-Lite`	1,558	95	$0.26
After	normalized cache hit on R022: "What happens when my card payment fails"	0	0	$0.00

● **Change 5 · Answer repeated questions from a response cache** `support/answer.py · sketch`

```
   def answer(question):
+      hit = cache.get(question)  # exact, then normalized
+      if hit: return hit
       out = ...
+      cache.put(question, out)   # flushed when the KB changes
```

In this run: exact and normalized-text keys, replayed over the 81 requests.

Risk. A cached answer goes stale after a policy or help-doc change. **Rollback.** Clear the cache when the help docs change; entries also expire on a timer; or turn it off.

---

6 Batch the nightly job

$23 / month
0.1% of the original bill

The nightly digest is not urgent, so it runs as a batch job at half price. Bill before this lever $554, after $531 a month.

Applied alone to the original bill it would save $3,504 a month; it reads small here because earlier levers already removed most of the tokens it acts on.

Measured on request R009: “Nightly digest: Fjord Analytics”

	SETUP	INPUT TOK.	OUTPUT TOK.	PER 1,000 CALLS
Before	Synchronous call, nobody is waiting for it at 02:00 `Gemini 3.1 Flash-Lite`	913	98	$0.38
After	Same call through the provider's batch service (50 percent off, results within 24 h) `Gemini 3.1 Flash-Lite`	913	98	$0.19

- **Change 6 · Run the nightly digest through the provider's batch service**

`jobs/nightly_digest.py · sketch`

```
- for account in accounts:
-     digest = llm.chat(task="report", ...)
+ jobs = [digest_request(a) for a in accounts]
+ batch = submit_batch(jobs)  # half price, done within 24 h
+ # results collected by the 06:00 job
```

Priced at half on the digest calls, the providers' published batch rate.

Risk. Digests arrive later (within 24 h, not at once). **Rollback.** Switch the job back to synchronous calls.

Each code change is a provider-neutral sketch in the shape it would land in your repository; the real change uses your provider’s SDK. Caching, dedupe and batch were priced on measured tokens rather than run as separate model calls.

Price comparison: the same tokens on each provider’s list

The result in this report prices each call at the list price of the model that served it: $23,117 to $531 a month, the “As run” row. For comparison only, the same measured token counts are also priced as each provider’s big and small tier, our mapping to its commercial models. Claude Sonnet 5.5, GPT-6.1 Sol, Gemini 4 Carbon all list at the same $2.00 in and $10.00 out per million tokens, so the before bill is the same on every list; the after bill moves with each small model’s price.

PRICE LIST	BIG / SMALL TIER PRICED	BEFORE	AFTER	SAVED
As run (the result)	Claude Sonnet 5.5 / Gemini 3.1 Flash-Lite	$23,117	$531	97.7%
Anthropic	Claude Sonnet 5.5 / Claude Haiku 5.5	$23,117	$433	98.1%
OpenAI	GPT-6.1 Sol / GPT-6 Luna	$23,117	$425	98.2%
Google	Gemini 4 Carbon / Gemini 3.1 Flash-Lite	$23,117	$524	97.7%

Comparison only: the same tokens re-priced on each provider’s list. The result is the As run row.

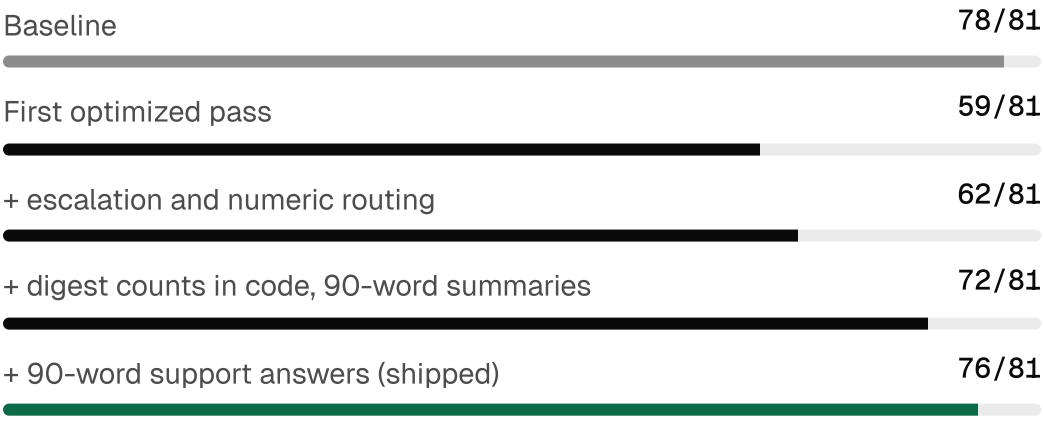
Google’s Gemini 4 Carbon list prices rise on 1 Dec 2026; the table uses the prices valid on 11 Nov 2026.

Quality check

Nothing ships that fails the quality check. The classifier is scored by exact label match; everything else by a grading model (Gemini 4 Argon, a second AI model) with a written rubric (correct, complete, policy-safe), the full knowledge base, must-have facts and a must-not list.

TASK	BASELINE	AFTER	CHANGE
Support answers (from the help docs)	36/37	35/37	-1
Ticket classifier	23/24	23/24	0
Ticket summaries	12/12	10/12	-2
Nightly account digest	7/8	8/8	+1
All checks	**78/81 (96.3%)**	**76/81 (93.8%)**	**-2**

How the pass rate moved while tuning



How we measured. The first optimized pass (59/81) was scored before any fix; the three fixes came from reading its failures. On your traffic we split before tuning: 80% to tune on, 20% held back as a test set and scored once, at the end.

Every check that still fails

ID	TASK	REQUEST	BASELINE	GRADER'S NOTE ON THE FINAL ANSWER
R019	Ticket summaries	Ticket T-7108 | Customer: Hiro (Business) Hiro: We need our data in the EU region for a new German client. Can you move …	pass	Missed specifying EU region and creating a new EU workspace.
R031	Support answers (from the help docs)	My invoice has the wrong company name. Can you reissue last month's?	pass	Did not offer to check with the Billing team about reissuing last month's invoice.
R045	Support answers (from the help docs)	How do I set up a webhook and verify it's really from you?	fail	Omitted Settings > Integrations > Webhooks and falsely claimed setup location wasn't in the policy.
R049	Ticket summaries	Ticket T-7109 | Customer: Dana (Team) Dana: Charged $1,740 on the 1st but we only have 60 seats at $29. Should be $1,740…	pass	Missing the 'no action' part of the status.
R056	Ticket classifier	Is there a way to switch the Kanban board to dark mode?	fail	exact label mismatch

Risks and rollback

Each lever ships behind its own flag, so any one can be turned off without touching the others. The quality check runs on every code change and nightly for the 30 days after.

LEVER	WHAT CAN GO WRONG	ROLLBACK
Trim the prompt and retrieved context	What to watch: a needed passage drops out of the shorter context. The top 3 passages held every passage an answer needed for 88.9% of questions (94.4% with the first 10); the quality check catches the rest.	Feature flag per prompt version; revert restores the old prompt and top-k in one deploy.
Cap output and reasoning	Truncated answers. In this run one big-model digest hit its 800-token cap at the caps stage (that stage was not shipped).	Caps are per task in one dict; raise or remove a cap without other changes.
Route easy calls to a cheaper model	The small model misses. The escalation rule re-ran 4 answers on the big model; 7 numeric or sensitive questions went straight to it.	Router returns BIG for everything when the flag is off.
Prompt caching	A lower hit rate than the 95% assumed (cache expiry, quiet hours); a prefix that changes per call caches nothing.	Turn caching off; prices fall back to normal input rates, answers unchanged.
Dedupe repeated calls	A cached answer goes stale after a policy or help-doc change.	Clear the cache when the help docs change; entries also expire on a timer; or turn it off.
Batch the nightly job	Digests arrive later (within 24 h, not at once).	Switch the job back to synchronous calls.

30-day verification plan

Savings are measured per request, so a busier or quieter month does not change the answer. The baseline cost per request is frozen before any code change merges.

`Day 0`	Freeze the baseline: last full invoice month, reconciled with the usage export, divided by its request count. Sample: $0.019264 per request.
`Days 1-3`	Merge the code changes behind flags. Turn them on for 10% of traffic; compare cost per request and quality pass rate against the other 90%.
`Days 4-7`	Full traffic once the reserved test set scores within the agreed margin of the baseline. Any lever that fails is switched off and listed in the report.
`Days 8-30`	Daily cost per request from the usage export, nightly quality check on the reserved test set, an alert if either drifts.
`Day 30`	Reconcile the invoice. Verified saving = (baseline cost per request − new cost per request) × requests actually served. Sample: ($0.019264 − $0.000443) × 1,200,000 = $22,586.

The guarantee

$1,900

FIXED FEE · ABOUT ONE WEEK

If verified 30-day savings are below the $1,900 fee, we refund the difference. Verified saving = (old cost per request − new cost per request) × requests served. You keep the code changes, the quality test set and the report either way. This fits teams spending at least $5,000 a month on model APIs; below that the fee can exceed what there is to save.

- ✓ Shipped code changes, one per lever, each behind a flag
- ✓ A before and after report reconciled to your invoice
- ✓ A quality test set built from your own traffic, with a reserved test split
- ✓ A ranked fix list with dollars per fix

Get this report on your own bill

Send your AI bill export. The free AI check, a one-page report within 48 hours, tells you which of the six levers apply and a dollar range, before you pay anything. Or check your bill free in your browser with the LLM Bill Checker.
Get a free AI check: khlabs.dev/#check · hello@khlabs.dev

Founding pilots: 50% off until 15 Dec 2026. Cut Your AI Bill ~~$1,900~~ $950 for our first 3 clients. Or write to hello@khlabs.dev. Download this report as a PDF.

Method

- **Workload.** 81 requests: 37 support questions over a 32-chunk knowledge base (8 are repeats or rephrasings), 24 ticket classifications with correct labels, 12 ticket summaries, 8 nightly digests.
- **Models.** Big Claude Sonnet 5.5 (run as claude-sonnet-5-5-high), small Gemini 3.1 Flash-Lite (run as gemini-3.1-flash-lite), grader Gemini 4 Argon (run as gemini-4-argon-high). Tokens from each call's usage field; each call priced at its model's list price.
- **Caching** assumes 95% of a stable prefix of at least 1,024 tokens is served from cache; cache expiry and storage fees are reconciled against your invoice when we run it on your bill. **Dedupe** found 3 exact and 5 normalized repeats in 81. **Batch** is priced at half.
- **Scoring.** Every stage is a real run at temperature 0, scored by exact label match and by one grading model with a written rubric. On your traffic we add repeated runs per stage and the held-back test split.
- **Data.** Raw data: results.json. The baseline above is re-added from the run log and must match the analysis before this report is built.