OPENCLAW

Stop Burning Money

Real strategies used in production to slash your AI bill from $200/mo to $15/mo.

Version 1.0 · February 2026 · openclaw-guide.com

□ Table of Contents

Version 1.0 | February 2026

Real strategies used in production to slash your AI bill without sacrificing capability.

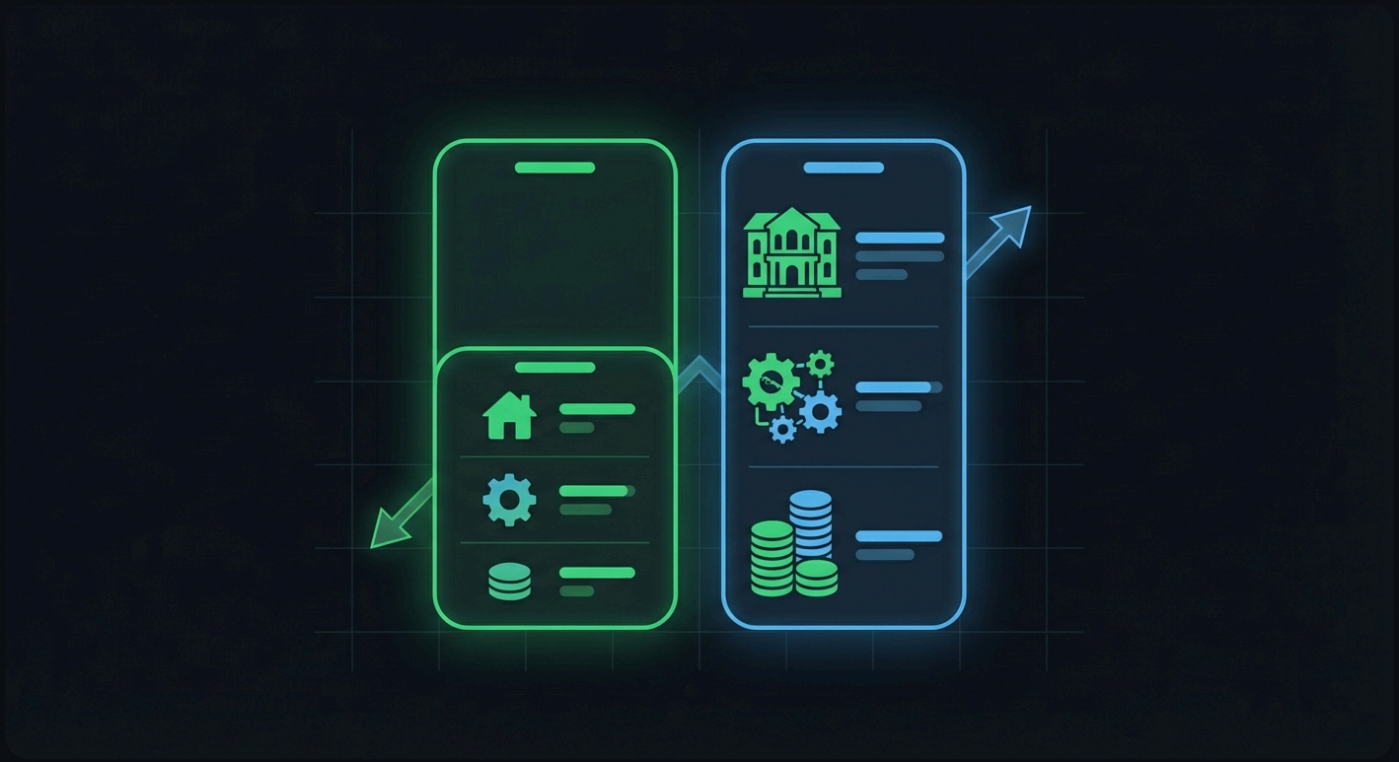
The Problem

Most OpenClaw users start with:

- Claude Opus for everything
- No model routing
- Heartbeats running Opus every 5 minutes
- Unoptimized prompts

Result: $150-300/month in API costs.

After this guide: $15-30/month.



Strategy 1: Model Routing (Biggest Savings)

The 80/20 Rule: 80% of tasks don't need Opus.

The Routing Hierarchy

Task Type	Use This Model	Cost Ratio
Heartbeats/Crons	Haiku 3.5	1x (cheapest)
Quick questions	Haiku 3.5	1x
Daily briefings	Haiku 3.5	1x
General chat	Sonnet 4.5	3x
Code review	Sonnet 4.5	3x
Complex planning	Opus 4	15x
Deep research	Opus 4	15x

Implementation

JSON

```
// In ~/.openclaw/config.json
{
  "agents": {
    "defaults": {
      "model": "claude-sonnet-4-5"
    }
  },
  "crons": {
    "defaultModel": "claude-haiku-3-5"
  }
}
```

Savings: Heartbeat running every 30 minutes:

- Opus: $0.50/check × 48/day = $24/day = $720/month
- Haiku: $0.01/check × 48/day = $0.48/day = $14.40/month

That's a 50x savings on heartbeats alone.

Strategy 2: Cache Your Memory

Problem: Every session re-reads memory files.

> □ **Solution:** Cache with TTL.

Before (Every Session)

TERMINAL

```
Session 1: Read 50KB memory → 50,000 tokens
Session 2: Read 50KB memory → 50,000 tokens
Session 3: Read 50KB memory → 50,000 tokens
Total: 150,000 tokens
```

After (1-Hour Cache)

TERMINAL

```
Session 1: Read 50KB → 50,000 tokens (cache set)
Session 2: Cache hit → 0 tokens
Session 3: Cache hit → 0 tokens
Hour passes
Session 4: Read 50KB → 50,000 tokens (cache refresh)
Total: 100,000 tokens (33% reduction)
```

Implementation

JSON

```
{
  "memory": {
    "cacheTtlMs": 3600000
  }
}
```

Strategy 3: Prompt Compression

Verbose prompt (2,500 tokens):

```
TERMINAL
I need you to help me analyze the following data. Please look at the
information provided and give me a comprehensive breakdown of what
you see. I'm particularly interested in trends, patterns, and any
anomalies that might be present. Take your time and be thorough...
```

Compressed prompt (400 tokens):

```
TERMINAL
Analyze this data. Report: trends, patterns, anomalies. Be concise.
```

Savings: 84% fewer input tokens.

Quick Compression Rules

1. Remove filler words ("I need you to", "please", "if possible")
2. Use bullet points instead of paragraphs
3. Combine related instructions
4. Drop redundant context

Strategy 4: Smart Heartbeat Intervals

Common mistake: Checking every 5 minutes "just in case."

Reality: Most systems don't need sub-minute responsiveness.

Recommended Intervals

Task Type	Interval	Why
Critical alerts	5 min	Fail fast
Daily briefings	60 min	Timely but not urgent
Health checks	30 min	Balance coverage/cost
Weekly reports	24 hours	Daily is overkill
Content generation	3 hours	Quality over frequency

Cost Comparison

Interval	Checks/Day	Monthly Cost (Haiku)
5 min	288	$8.64
15 min	96	$2.88
30 min	48	$1.44
60 min	24	$0.72

 Recommendation: 30 minutes for most use cases.

Strategy 5: Use Local Models for Simple Tasks

Free tier: Ollama + local models.

When to Use Local

- Summarization
- Simple classification
- Format conversion
- Quick lookups
- Draft generation (then polish with Claude)

Setup

BASH

```
# Install Ollama
curl -fsSL https://ollama.com/install.sh | sh

# Pull a model
ollama pull llama3.2:3b

# Configure OpenClaw
export OLLAMA_HOST=http://localhost:11434
```

Model Recommendations

Size	Model	Use Case
3B	Llama 3.2	Quick tasks, summaries
7B	Qwen 2.5 Coder	Code completion
14B	DeepSeek Coder V2	Complex code work

Strategy 6: Batch Your Operations

Instead of 10 separate requests:

```
TERMINAL
Request 1: "Add task A"
Request 2: "Add task B"
Request 3: "Add task C"
...
```

Do one batched request:

```
TERMINAL
Add these tasks:
- Task A: [description]
- Task B: [description]
- Task C: [description]
```

Savings: 70-90% fewer API calls = fewer tokens on system prompts.

Strategy 7: Session Management

Problem: Each new session loads system prompt + memory + context.

> □ **Solution:** Keep sessions alive longer.

Before

```
TERMINAL
10 sessions/day × 10,000 tokens overhead = 100,000 tokens/day
```

After

```
TERMINAL
2 sessions/day × 10,000 tokens overhead = 20,000 tokens/day
```

Savings: 80% on overhead.

The $15/Month Stack

Here's a real configuration used in production:

JSON

```
{
  "agents": {
    "defaults": {
      "model": "claude-sonnet-4-5"
    }
  },
  "crons": {
    "defaultModel": "claude-haiku-3-5"
  },
  "memory": {
    "cacheTtlMs": 3600000
  },
  "heartbeat": {
    "intervalMs": 1800000,
    "model": "claude-haiku-3-5"
  }
}
```

Monthly breakdown:

- Heartbeat (30 min): $1.44
- Daily briefings (Haiku): $0.90
- Main chat (Sonnet, 500 msgs): $7.50
- Complex tasks (Opus, 50 msgs): $3.75
- Local model (Ollama): $0.00

Total: ~$14/month

Quick Savings Calculator

Your Current Spend	With Optimization	Savings
$50/mo	$15/mo	70%
$100/mo	$20/mo	80%
$200/mo	$30/mo	85%
$500/mo	$50/mo	90%

Cost Optimization Checklist

- ☐ Default model set to Sonnet (not Opus)
- ☐ Heartbeat model set to Haiku
- ☐ Heartbeat interval at least 30 minutes
- ☐ Memory cache enabled (1 hour TTL)
- ☐ Prompts compressed
- ☐ Ollama installed for local tasks
- ☐ Batching operations where possible
- ☐ Sessions kept alive longer

Next Steps

Want the complete system?

 The Autonomous Agent Playbook ($49)

- ▸ Full cost tracking dashboard
- ▸ Automated model routing
- ▸ Budget alerts setup
- ▸ Real production configs

 Free PDF #3: Security Checklist

- ▸ API key management
- ▸ Channel authentication
- ▸ What to NEVER expose

Get all guides at: openclaw-guide.com

Built with ❤ by the OpenClaw community

Questions? Join us at discord.gg/clawd