

THE HERMES AGENT

PLAYBOOK

Free companion to the Hermes Agent Full Course

TABLE OF CONTENTS

Section 1: What is Hermes	0:01:33
Section 2: Cost & Model Options	0:04:21
Section 3: Three Levels of Hermes	0:06:44
Section 4: Install (Level Zero)	0:08:26
Section 5: Hermes Workspace (Level One)	0:15:28
Section 6: Knowledge Fortress	0:41:08
Section 7: Agent Taxonomy	0:43:39
Section 8: OpenClaw vs Hermes.....	0:44:37
Section 9: Migration Guide	0:51:44
Section 10: The Build Begins	0:54:20
Section 12: Design & Theming	0:57:00
Section 13: Webhook Automation	1:19:10
Section 14: AI-First Architecture	1:39:17
Section 15: Hermes Takes Over	1:45:12
Section 16: Meeting Intelligence	1:50:39
Section 17: Cron Jobs	2:00:37
Section 18: Self-Improving SDR	2:24:43
Section 19: Email System (Resend)	2:41:36
Section 20: Nurture Sequences	3:00:52
Section 21: Campaign Builder	3:11:27
Section 22: Activity Log	3:14:47
Section 23: Automated Campaigns (Apify)	3:32:19
Section 24: Campaign Approval	3:42:34
Section 25: Skill Builder	3:47:27
Everything Built – Final Recap	4:05:09

Appendix: Terminal Commands Reference

Appendix: Key Concepts Glossary

Appendix: Tech Stack Reference

Appendix: Full Timestamp Index

HOW TO USE THIS GUIDE

This guide contains every prompt, every terminal command, and every concept from the 4-hour Hermes Agent Full Course. It is designed to be your complete reference — whether you are watching the video or building on your own.

Two Ways to Use This Guide

1. Follow along with the video — Each section includes timestamps so you can jump to the exact moment in the course. Copy prompts directly from this guide instead of pausing and retyping.
2. Hand it to your AI agent — This entire guide can be pasted into Claude Code, Hermes, or any AI agent as context. The agent can then execute the builds, set up the integrations, and configure the cron jobs documented here.

Every prompt is formatted in a dark code block with a gold border so you can spot them instantly. Terminal commands use the \$ prefix. Key quotes are indented with a gold left border.

| *"An AI agent should get better the more you use it."*

Let us begin.

SECTION 1: WHAT IS HERMES

Overview

Hermes is an open-source AI agent built by Nous Research — the team that actually trains AI models for a living. Unlike ChatGPT, Claude, or even OpenClaw, Hermes is built on one core belief: an AI agent should get better the more you use it.

Every other agent resets between sessions. You ask it something today, it answers. You ask the same thing next week, it starts from zero. Hermes works differently. After completing a task, it records everything it did — every API call, every decision, every tool it used, and in what order. Nous Research calls this a trajectory.

The agent then examines the trajectory and asks itself: can I package this into something reusable? If yes, it creates a skill — a standard operating procedure the agent wrote for itself. Next time you ask something similar, it runs the skill: faster, cleaner, more consistent. If something changes, it refines the skill. It literally gets better.

Hermes vs OpenClaw

People confuse them constantly, but they are completely different tools. OpenClaw is a Swiss army knife — it connects to 50+ platforms and does everything. It is a general-purpose AI assistant. Hermes is a specialist — it does fewer things, but the things it does, it does better than anything else on the market.

Key Stats

Metric	Value
GitHub Stars	65,000+ (fastest growing agent on GitHub)
Contributors	200+
License	MIT — completely free
Model Support	Any model: Claude, GPT, Nous models, Ollama local
Core Feature	Self-improving via Trajectories to Skills pipeline
Memory	Three-layer persistent memory
Speed	~40% faster on repeated tasks (via skills)

Trajectories to Skills Pipeline

This is the core innovation that makes Hermes unique:

1. You give Hermes a complex task (something that takes 5+ tool calls)
2. Hermes records the entire process — every API call, every decision, every step
3. It asks itself: Can I package this into something reusable?
4. If yes, it writes a skill file — a markdown document with detailed instructions, pitfalls to avoid, lessons learned, and verification steps
5. Next time you ask something similar, it runs the skill instead of starting over
6. If conditions change, it refines the skill automatically

"This is procedural memory. The agent is literally writing its own playbooks. I did not see that anywhere."

BUILT: Understanding of Hermes core architecture: self-improving agent with trajectory-based learning

SECTION 2: COST & MODEL OPTIONS

The Anthropic Situation

Anthropic recently banned all third-party tools from using Claude subscription directly. This includes OpenClaw, Hermes, and everything that is not Claude Code. They provide a \$200 one-time credit called 'Extra Usage' — these are not API credits. This is what Mani uses in the course.

Important: Even with Extra Usage, Opus 4.6 was giving errors during filming. The course was moved to GPT 5.4 via Codex subscription for reliability.

Model Options (Cheapest to Most Expensive)

Option	Cost	Quality	Notes
Nous Models (OpenRouter)	Free (\$0)	Basic	Hermes 3 on OpenRouter. Not the smartest but works.
Ollama Local	Free (\$0)	Varies	Run on your own GPU. Free, private, slower.
ChatGPT Codex	\$20/month	Good	BEST VALUE. No per-token billing. Includes Codex.
Anthropic Extra Usage	\$200 credit	Best	Opus 4.6 quality. Credit runs out eventually.
Anthropic API	Per token	Best	Most expensive. Only if generating \$10K+/month.

"The real cost can start with zero and max 20 bucks a month is what I would recommend for something that helped me close six deals in three weeks."

Recommendation

For most viewers: Start with ChatGPT Plus (\$20/month) which includes Codex. It has some limits but no per-token billing. If you are already generating significant revenue, the Anthropic API with Opus gives the best quality.

BUILT: Model selection strategy: Free (Nous/Ollama) for testing, \$20/mo Codex for production, API for high-revenue operations

SECTION 3: THREE LEVELS OF HERMES

Level 0: Terminal + Telegram

The basic installation. You chat in the terminal or connect Telegram and text from your phone. This is where most tutorials stop and most people stay. You are flying blind — no visibility into what the agent is doing, no skill management, no session history.

For a lot of people, this is enough. You can text your AI from your phone, it runs on your machine, and it responds. But you can do much more.

Level 1: Hermes Workspace

An open-source web dashboard that gives you a cockpit. You can see every conversation, browse memory, manage skills, and watch the agent think in real time. It is like going from a command line to an actual cockpit.

Features: Dashboard, Chat, Files, Terminal, Jobs, Memory, Skills, Sessions — everything in one window. Full text search across all sessions.

Level 2: Custom Command Center

This is what the course builds. Your own custom web application using Claude Code as the builder. It connects Hermes, pulls in your meetings (Fathom), tracks outreach, manages clients — everything in one screen tailored to your business.

Nobody else has built this yet. This is where you level up from using an agent to running an agent-powered business.

Level	Tool	Visibility	Who It's For
0	Terminal + Telegram	Blind	Beginners, quick tasks
1	Hermes Workspace	Dashboard	Power users, skill management
2	Custom Build	Full Control	Agency owners, serious builders

BUILT: Understanding the three-tier architecture: Terminal, Workspace, Custom Command Center

SECTION 4: INSTALL — LEVEL ZERO

One-Command Install (macOS)

The installation takes barely 60 seconds. Visit the Hermes website and copy the install command. For Windows users, install WSL first, then run the command.

```
$ curl -fsSL
https://raw.githubusercontent.com/NousResearch/hermes-agent/main/install.sh
| bash
```

After installation, the setup wizard offers provider options:

1. Anthropic / Claude Models (API key or Extra Usage)
2. OpenAI / Codex (recommended for \$20/month)
3. OpenRouter (free Nous models)
4. Ollama (local models)
5. GitHub Copilot / HuggingFace

Telegram Bot Setup

To connect Telegram, you need two things: a bot token from BotFather and your target chat ID.

```
$ /newbot (send to @BotFather in Telegram)
```

Follow BotFather's prompts to name your bot and receive the token.

Open @userinfobot in Telegram — it will display your user ID. This is your target chat ID.

PROMPT:

```
Give Hermes your bot token and chat ID.
It will configure the Telegram gateway automatically.
If 'hi' doesn't get a response, Hermes will diagnose
and restart the gateway as a clean instance.
```

Troubleshooting

If the first setup attempt fails (500 error), keep trying. The transition from Anthropic subscription was not smooth and some users experienced errors. By the time you watch the video, it may be fixed. You can also use OpenAI Codex, OpenRouter, or Kimi as alternatives.

"For those of you who are into no-code, no-code. Or a lot of people this is enough."

BUILT: Hermes installed, Telegram connected, agent responding to messages from phone

SECTION 5: HERMES WORKSPACE — LEVEL 1

Clone and Run

```
$ git clone https://github.com/NousResearch/hermes-workspace.git
$ cd hermes-workspace
$ pnpm install
$ pnpm dev
```

If pnpm is not found, install it first:

```
$ npm install -g pnpm
```

The workspace runs on localhost. If the backend gateway is not running, you may need to set it up separately. You can ask your Claude Code agent or Hermes itself to handle the setup — they can run the required commands for you.

Backend Gateway Setup

The gateway requires Python. If you run into Python version issues, your builder agent (Claude Code or Codex) can fix them. The setup involves:

- 1. Set the PATH for Hermes agent
- 2. Clone the Hermes agent gateway
- 3. Create a Python virtual environment
- 4. Install dependencies
- 5. Start the gateway

PROMPT:

```
I am running my Hermes workspace and I'm running into these issues [paste error]. I want you to go fix it. The exact steps to resolve the Python/gateway errors.
```

Workspace Features

Feature	Description
Dashboard	Overview of agent status, recent activity, and system health
Chat	Conversational interface with full context

Feature	Description
Files	Browse agent's file system and configurations
Terminal	Watch commands execute in real time
Jobs	Background tasks and scheduled operations
Memory	Three-layer persistent memory browser
Skills	Browse, read, edit, install skills — the brain of the learning system
Sessions	All past conversations — click to continue any session
Search	Full-text search across sessions, skills, files — everything

Skills Marketplace

The workspace includes a marketplace with close to 2,000+ community skills — everything from GitHub issue workflows to Kubernetes management. But the real power is the skills Hermes creates itself based on your usage patterns.

Pre-built skills available: GitHub Issues, PowerPoint, Popular Web Designs, GitHub PR workflows, Kubernetes management, and 2,000+ more in the community hub.

"The real power is not the pre-built ones. It's the skills Hermes creates itself based on what you do and how you work."

BUILT: Hermes Workspace running with Dashboard, Chat, Files, Terminal, Jobs, Memory, Skills, Sessions, and Search

SECTION 6: KNOWLEDGE FORTRESS

Before building the Level 2 custom application, Mani introduces the Knowledge Fortress — a five-layer context system that powers professional builds. This is the difference between amateur and production-quality AI development.

The Five Layers

Layer	Contents	Purpose
1. Documentation	PRDs, database schemas, UI/UX specs	What are we building?
2. Security	Security best practices, framework rules	How do we keep it safe?
3. Technical	Prompting best practices, tech stack selection	How do we build it right?
4. Strategy	Personal preferences, project direction	Where are we going?
5. Knowledge	Research, historical interactions, prior work	What do we already know?

All five layers feed into your Claude project (the .claude.md files) and your builder agent. They can be markdown files, PDF files — the format does not matter. What matters is that the context is comprehensive.

The Professional Workflow

- 1. Idea: Define what you want to build (e.g., Agency Mission Control)
- 2. Context Generation: Chat with Claude Opus 4.6 to generate the documentation layer
- 3. Prototype: Use Lovable or Google Gemini 3.1 Pro for amazing UI/UX designs
- 4. Build Logic: Wire the backend, connect Supabase, create API integrations
- 5. Integration Guide: Make the app usable by both humans AND agents (edge functions)
- 6. Deploy: Ship it

"We are building applications that both humans and agents can use. How do humans use it? Beautiful UI. How do agents use it? Through API — building all the edge functions."

BUILT: Five-layer Knowledge Fortress framework for production AI development

SECTION 7: AGENT TAXONOMY

To maximize efficiency and control costs, Mani categorizes agents into three distinct roles. This taxonomy has been refined through 18+ months of production AI development since October 2024.

Three Agent Categories

Role	Agent	Function	Analogy
Orchestrator	OpenClaw	Coordinates tasks, delegates to other agents, handles edge cases	Manager
Builder	Claude Code / Codex	Writes code, constructs systems, builds the infrastructure	Developer
Executor	Hermes / Sub-agents	Completes specific tasks, learns and improves over time	Specialist Worker

The Orchestrator agent manages workflow and delegation. The Builder agent constructs the tools and systems. The Executor agent runs the day-to-day operations and improves with each execution. This separation is critical for cost management and quality control.

You can use Claude Code Agent SDK, OpenClaw, Hermes, or OpenClaw sub-agents as executor agents. The key is specialization — each agent role has different strengths and cost profiles.

"I distinguish agent roles into three categories. This is to maximize efficiency and control costs. I have been creating content using Vibe Coding Tools from October 2024."

BUILT: Agent taxonomy framework: Orchestrator (OpenClaw) + Builder (Claude Code) + Executor (Hermes)

SECTION 8: OPENCLAW VS HERMES — THE HONEST COMPARISON

This is one of the most-asked questions: Should I use OpenClaw or Hermes? Did Hermes kill OpenClaw? Should I switch? Mani uses both extensively and provides an honest, data-backed comparison.

Head-to-Head Comparison

Category	OpenClaw	Hermes Agent
Self-Improvement	Add-on (ClawHub skill)	Native core feature
Memory	Memory Wiki (cross-session)	Three-layer persistent memory
Speed (repeated tasks)	Standard	~40% faster (via skills)
Stability	Frequent breaking updates	More stable (not perfect)
Skills Ecosystem	13,700+ ClawHub skills	Built-in + self-generated skills
Integrations	50+ platforms	6 native channels + any API
Telegram	Built-in	Native built-in
GitHub Stars	346,000	65,000+ (fastest growing)
Community	170,000 Discord members	Built by Nous Research
Code Quality	Good	Better (per reports)
Best For	Orchestration, multi-agent	Execution, deep work, autonomy

What the Community Says

From Alex Finn's video (95,000+ views at time of filming) and other sources:

"After a day of playing with Hermes, I deleted OpenClaw."

"I switched from OpenClaw to Hermes and it's night and day in code quality."

"I tried Hermes for 15 minutes and gave up OpenClaw immediately."

But there is also the other side. OpenClaw's ecosystem is bigger, has way more integrations, 50+ platforms. The highest-rated skill on ClawHub is actually a self-improvement agent skill that mimics what Hermes does natively.

When to Use Which

Use Case	Best Choice	Why
Multi-platform orchestration	OpenClaw	50+ integrations, multi-agent coordination
Deep work and learning	Hermes	Self-improving skills, persistent memory
Quick automation	OpenClaw	Larger skill marketplace, faster setup
Agency-specific tasks	Hermes	Learns your business patterns over time
Cost-sensitive setup	Hermes	Lighter, fewer dependencies
Maximum ecosystem	OpenClaw	13,700+ community skills

The Power Move: Run Both

Mani runs both. OpenClaw handles orchestration and multi-platform integrations. Hermes handles execution, deep work, and autonomous improvement. They complement each other rather than compete.

OpenClaw has been catching up — they added Memory Wiki for cross-session memory. But in Hermes, self-improvement is the core architecture, not an add-on.

"They are not the same tool. They don't even compete in the same category in my opinion."

BUILT: Complete OpenClaw vs Hermes decision framework with data-backed comparison

SECTION 9: OPENCLAW TO HERMES MIGRATION GUIDE

For those switching from OpenClaw to Hermes, or wanting to run both side by side, here is the complete migration process.

Step 1: Export Your OpenClaw Data

Before migrating, export your existing OpenClaw workflows, skills, and configurations. This preserves your investment in the platform.

```
# Export OpenClaw configurations
cd ~/.openclaw
cp -r workflows/ ~/openclaw-backup/workflows/
cp -r skills/ ~/openclaw-backup/skills/
cp -r memory/ ~/openclaw-backup/memory/
cp config.json ~/openclaw-backup/
```

Step 2: Install Hermes

```
$ curl -fsSL
https://raw.githubusercontent.com/NousResearch/hermes-agent/main/install.sh
| bash
```

Run through the setup wizard and configure your preferred model provider.

Step 3: Transfer Context

Convert your OpenClaw memory and context into Hermes-compatible format. Hermes uses markdown-based memoryfiles.

PROMPT:

```
I am migrating from OpenClaw to Hermes. Here is my
OpenClaw configuration and key workflows. Please:
1. Save this context to your memory
2. Create skills for my most common workflows
3. Set up equivalent automation for my existing tasks

[Paste your OpenClaw config and workflow descriptions]
```

Step 4: Recreate Key Workflows as Skills

For each critical workflow, give Hermes the task and let it create a skill naturally. The trajectory-to-skill pipeline will capture the process.

Step 5: Running Both Simultaneously

You can run OpenClaw and Hermes sidebyside. They use different ports and different configuration directories. There is no conflict.

```
# OpenClaw runs on its default port
# Hermes runs on a separate port
# Both can connect to the same Telegram bot
# (use different bots for clarity)
```

Troubleshooting

Issue	Solution
Port conflict	Change Hermes gateway port in config
Telegram conflicts	Use separate bots for each agent
Memory not transferring	Manually paste key context into Hermes chat
Skills not generating	Ensure tasks are complex enough (5+ tool calls)
API key issues	Each agent needs its own API key configuration

BUILT: Complete migration path from OpenClaw to Hermes, with option to run both

SECTION 10: THE BUILD BEGINS — MISSION CONTROL

Architecture

The Level2build uses a dual-agent architecture: Claude Code as the builder agent and Hermes as the orchestrator/executor. The tech stack is React + Vite + Tailwind + Supabase.

The goal is to build Hermes Mission Control — a custom web application that connects to Fathom (meeting recordings), tracks outreach, manages clients, and gives the agent full visibility into your business operations.

ClawBuddy: The Inspiration

Before building, Mani shows ClawBuddy — his existing Mission Control system. This is what the course builds toward:

Meeting Intelligence: 1,700+ meetings with AI analysis, key takeaways, action items

Outreach Dashboard: Campaign tracking, email analytics, lead management

Agent Teams: Multi-agent orchestration with task assignment

Research Hub: Autonomous research capabilities

Email Agent: Automated email sending with engagement tracking

Phone Agent: Voice AI integration

Integration Guide: API documentation for agent access

The Integration Guide Concept

The most important architectural decision: every feature must be usable by both humans AND AI agents. Humans use the beautiful UI. Agents use edge functions and API endpoints. This is achieved through an Integration Guide — a comprehensive API reference that any agent can read and execute.

PROMPT:

```
I am building a Hermes Mission Control application.
I need to give you the API key for fetching my Fathom
meetings. Also I'm hoping the UI would look something
like this - futuristic. This has a green theme but I want
to go with the blue theme. Feel free to improve the visual
aesthetic of this application. Make it futuristic.
```

```
Also let me know how to add the API keys securely in the
chat. I don't want to share my API keys in the chat nor
```

```
you should display them in the chat. Tell me about the
secure way to share them – you can give me a .env.local
file for me to update.
```

Secure API Key Management

Claude Code creates a `.env.local` file that is hidden from the UI. You need an IDE (VS Code, etc.) to open and edit it. Add your keys:

```
# .env.local
VITE_SUPABASE_URL=your_supabase_url
VITE_SUPABASE_ANON_KEY=your_anon_key
FATHOM_API_KEY=your_fathom_api_key
RESEND_API_KEY=your_resend_api_key
```

"\$54,613. That's how much hit my Stripe account in the last 30 days. An AI agent did it."

BUILT: Mission Control project initialized with secure API key management and Fathom integration

SECTION 11: DESIGN & THEMING

Screenshot-Driven Design

Instead of describing designs in words, Mani uses screenshots as design references. Take a screenshot of a design you like, paste it into your builder agent, and say 'make it look like this.' The AI can see the image and replicate the aesthetic.

Design inspiration sources:

Dribbble — Search for 'dashboard design' or 'futuristic UI'

v0.dev — Vercel's AI design tool for quick prototypes

Lovable — Full design-to-code generation

Google Gemini 3.1 Pro — Excellent for UI/UX generation

The design.md Concept

Create a `design.md` file in your project root that captures your visual preferences. Claude Code references this file for all UI decisions:

```
# design.md
##Theme
-Color scheme: Deep navy blue primary
-Accent: Glassmorphic cards with glow animations
-Typography: Clean, modern, readable
-Layout: Sidebar navigation, card-based content

##Inspiration
-Futuristic dashboard aesthetic
-Blue theme (not green)
-Glass morphism for cards
-Glow animations on hover
```

Plan Mode for Complex UI

For major UI changes, use Plan Mode in Claude Code. Instead of directly building, the agent creates a detailed plan first, then executes. This prevents assumptions and produces significantly better results.

PROMPT:

```
I want some visual improvements to this application.
Right now the text is really tiny and the sidebar menu
is ugly and not readable. In some areas text is just too
light — we need to make it brighter, whiter.
```

Also I would need a dedicated page for these meetings. Based on what you have in the Fathom Meetings table, I want you to come up with a visually pleasing page design.

I want you to do two things in this pass:

1. Improving text size, making it visually pleasing
 2. Creating a page for each meeting with a page loader
- Design a visually pleasing page based on the data we have in Fathom meetings. Also add dedicated sections for Hermes to add notes, generate proposals, or generate custom action items on top of what Fathom already generated.

Cross-Agent Design Review

A powerful technique: use one AI model to review another's work. When Claude Code's visual output is not perfect (e.g., padding issues), copy the issues into Claude Chat or Codex and ask them to generate fix instructions. Then paste those instructions back into Claude Code.

"It almost felt like it was made with a junior dev. Even though Opus 4.6 is a good model, sometimes you cannot just ask it to improve — it stops at one point."

PROMPT:

The padding in corners is off. Logo has empty space before the menu starts. Some of the cards look like they are overlapping. There is no proper spacing.

Tell me explicitly which file to edit. Instead of 'add outer padding' say 'open App.tsx and wrap the top-level div with a padding container.'

BUILT: Glassmorphic dashboard UI with proper theming, dedicated meeting pages, and cross-agent design review workflow

SECTION 12: FATHOM WEBHOOK & MEETING IMPORT

Importing Existing Meetings

Fathom is a meetingnote-taker that joins your meetings automatically and takes detailed notes. It has an API for programmatic access to all your recordings, transcripts, and summaries.

PROMPT:

```
I replaced my Fathom API key. I want you to make the API calls and let me know how many meetings I have. Based on that, we will make a plan to import it over. I don't need everything - I need all the calls that happened in the past 45 days. Let's import them to our Supabase and create a new table with relevant details: date, meeting name, transcript, summary, action items, company name (if you can figure it out), emails, any kind of details, type of the call.
```

```
We are bringing over 45 calls. Also I'm giving you the UI/UX inspiration - follow the design.md guidelines and make this app really futuristic with a blue theme.
```

Setting Up the Webhook

After importing historical meetings, set up a webhook so future meetings are automatically synced. This creates a real-time pipeline from Fathom to your Mission Control.

PROMPT:

```
We are building a Fathom webhook. Create a webhooks page in Mission Control where I can register, manage, and test webhooks. Include:
-Webhook list page with status cards
-Add webhook modal
-Webhook detail page
-Function grid for webhook actions
-Supabase edge functions for receiving webhook data
```

```
Once you give me the webhook URL, I will update it in my Fathom application. I want you to come up with a detailed plan to implement this and a checklist as well.
```

Supabase CLI Authentication

To deploy edge functions, you need Supabase CLI authenticated:

```
$ supabase login
```

Open the browser, enter the verification code, and authorize. You can also use a token:

```
$ export SUPABASE_ACCESS_TOKEN=your_token_here
```

Configuring the Webhook

1. Goto Fathom Settings > Integrations > Webhooks
2. Click 'Add Webhook'
3. Paste the webhook URL from Mission Control
4. Enable: My Recordings, External Recordings, Transcripts
5. Enable all available event types
6. Save the webhook

PROMPT:

```
Just make it so that the data that we receive from Fathom
for this webhook is automatically saved to our meetings.
In future when we want to add additional functions we can,
but for now just save that as an automatic behavior.
```

This transforms the system into a real-time pipeline: new meetings are automatically imported, stored in Supabase, and available for Hermes to analyze.

"Think of it like a vending machine vs a chef. A vending machine: you press a button, you get a chip. A chef: it knows who you are, what your preferences are, and makes exactly what you want."

BUILT: Real-time Fathom webhook pipeline: meetings automatically imported to Supabase on completion

SECTION 13: AI-FIRST ARCHITECTURE

This is the most important architectural principle in the entire course: every feature must be API-accessible. If a human can do it through the UI, an AI agent must be able to do it through an API endpoint.

The Principle

RULE: Every feature in Mission Control must be usable by BOTH humans (via UI) AND AI agents (via API).

Human access: Beautiful glassmorphic React UI

Agent access: Supabase Edge Functions (REST API)

Update CLAUDE.md with this AI-first principle.

Future features (outreach, calls, etc.) follow this rule.

PROMPT:

You need to create another page called a settings page or an integration page. This integrations page will have all the API calls that an agent can do to use our application.

Everything a human can currently do — move certain things, fetch certain meetings, add certain meetings, create webhooks — AI should be able to use it as well.

We do that by creating edge functions that an agent can call. This is going to be a big build. As soon as I copy the integration guide and paste it to my AI agent, it should be able to pick up everything and start using the application. It can be Hermes, Claude Code, or OpenClaw.

19 Initial Endpoints

Claude Code created 19 API endpoints in the first pass. These cover all existing functionality:

Category	Endpoints	Actions
Meetings	4	List, Get, Update, Search
Webhooks	4	Create, List, Get, Delete
Functions	3	Create, List, Execute
Stats	2	Dashboard metrics, Meeting analytics

Category	Endpoints	Actions
Skills	3	List, Get, Create
System	3	Health check, Config, Logs

The Integration Guide

The output is a comprehensive Integration Guide page — a markdown/UI document that lists every endpoint, its parameters, and examples. When pasted into any AI agent, it gives the agent complete control over Mission Control.

```
# Integration Guide Structure
- Connection setup (API key, base URL)
- Authentication method
- Available endpoints with parameters
- Example requests and responses
- Error handling
- Rate limits
```

"This is a significant architectural change. We are making this app usable by both humans and AI — every future feature follows this rule."

BUILT: 19 API endpoints deployed. Integration Guide page created. Any AI agent can now control Mission Control.

SECTION 14: HERMES TAKES OVER

This is the pivotal moment in the course. The builder agent (Claude Code) has created the infrastructure. Now the executor agent (Hermes) receives the Integration Guide and begins autonomous operation.

Feeding Hermes the Integration Guide

PROMPT:

```
Wewillbeworking off of Hermes Mission Control
applicationwhere I have access to all my meetings.
Iwillgiveyou the integration guide.
Thekeyis:you take that integration guide and let me
knowyourthoughts. Save the integration guide first.
Thisguidewill be updated week over week because we
arealsogoing to create outreach and related solutions.
With the meetings, we will be creating skills for
proposals, how we run discovery calls, objectionswe
are addressing. We will create skills off of these
meetings and include those skills with new meetings.

Ifyou have any feedback about the application and what
features you would use and would like to see, suggest
more. But first, go through this guide.
```

Mani pasted 287 lines of the Integration Guide into the Hermes terminal. The agent immediately:

1. Saved the context to its memory
2. Analyzed all available API endpoints
3. Identified 112 meetings in the database
4. Provided feedback on the API structure
5. Suggested improvements and missing features
6. Started creating skills for meeting analysis

First Autonomous Actions

After receiving the Integration Guide, Hermes began operating autonomously. It pulled meeting data, analyzed transcripts, and provided insights:

API is live and working — confirmed 112 meetings

External meetings identified with unique companies

Total hours tracked: 75 hours of meetings

Proposed analysis order: Analysis first, then pull all external meetings

Suggested creating: Discovery Call Framework Skill, Objection Handling Skill, Proposal Template Skill

The Self-Improving Loop

Hermes does not just analyze — it improves. Unlike other agents where you manually update prompts, Hermes automatically flags changes and updates its skills:

After a discovery call: It examines the current proposal prompt and what was learned

If a client says 'monthly retainer is too expensive': It flags this and updates the skill

Next time: The skill automatically includes alternative pricing options

Over time: The proposal skill becomes a living document that reflects real client feedback

"Whenever a discovery call happens, it looks at the current proposal prompt, what we learned last time, and automatically updates the skill. I have not seen that with any AI agent so far."

BUILT: Hermes operating autonomously with Integration Guide. Self-improving skills for proposals and discovery calls.

SECTION 15: MEETING INTELLIGENCE

[1:50:39]

With Hermes connected to Mission Control, the next step is building intelligence on top of the raw meeting data. This transforms meeting recordings from passive notes into active business intelligence.

Discovery Call Framework

Mani shares two documents that are directly responsible for \$50,000+ in revenue over 30 days: the Proposal Engine prompt and the Discovery Call Framework. These are fed to Hermes so it can learn the patterns.

PROMPT:

```
I want to improve or actually rewrite your answer.
I will give you two files:
1. Our proposal engine - the prompt we currently use
2. Discovery call framework

These two documents are directly responsible for
making us $50,000+ since the past 30 days. The funny
thing is, they were also written by Hermes.
```

Meeting Insights

For each meeting, Hermes generates structured insights that are written back to Mission Control:

Insight	Description
Sentiment	Overall meeting tone (positive, neutral, negative)
Deal Probability	Likelihood of closing based on signals
Engagement Score	Level of prospect engagement during the call
Follow-up Priority	Urgency of next action
Decision Maker	Whether the decision maker was present
Pricing Discussion	Any pricing objections or preferences noted
Next Steps	Concrete action items identified
Proposal Ready	Whether enough info was gathered for a proposal

Feedback Mechanism

PROMPT:

We need to add a feedback mechanism at the insights level for each page so I can interact effectively. I want to add a comment and feedback section – a nice addition for me to give feedback or additional context about any meeting. Essentially we need a place to communicate – for me to add additional notes for each meeting and for you to work off of that. Give me a prompt that I can give my builder agent.

BUILT: Meeting Intelligence system with AI-generated insights, deal probability, engagement scores, and human feedback loop

SECTION 16: CRON JOBS — AUTOMATED INTELLIGENCE

Cron jobs transform Hermes from a reactive assistant into a proactive intelligence system. Three automated jobs run continuously without human intervention.

Cron Job 1: Meeting Analyzer

Runs every 2 hours. Pulls new meetings from Mission Control, analyzes transcripts, generates insights, and writes them back to the database.

```
# Meeting Analyzer Cron
Schedule: Every 2 hours
Action: Pull unanalyzed meetings from MCT API
Process: Analyze transcript, generate insights
Output: Write insights back to Mission Control
Skills Updated: Proposal engine, discovery framework
```

Cron Job 2: Hot Lead Detection

Monitors meetings for high-potential deals and sends immediate Telegram alerts when a hot lead is detected.

```
# Hot Lead Detection Cron
Schedule: After each meeting analysis
Triggers: High deal probability, decision maker present,
          budget confirmed, timeline under 30 days
Action: Send Telegram alert with deal summary
Includes: Deal size, probability, recommended next steps
```

Cron Job 3: Daily Briefing

Comprehensive daily summary sent via Telegram and email. Covers all meetings, pipeline status, client health, and action items.

```
# Daily Briefing Cron
Schedule: Daily
Content:
- Yesterday's meetings summary
- Today's action items
- Pipeline status
- Client health overview
- Upcoming events
- Skills updated (what improved overnight)
```

Results After First Run

After the first cron cycle ran, Hermes reported back via Telegram:

Analyzed 6 meetings total: 1 discovery call, 3 hotline/kickoff, 1 Premier DFC

Skills Updated: Proposal engine received 4 patches

Average deal size: Updated pricing model, added consulting hourly rate (\$100/hr)

Discovery framework: 3 patches applied

Added quarter payment option for existing clients

Added multi-phase agent build pricing (\$18.5K phase 1, \$150K+ year 1)

Added 3 new green flags to watch for in discovery calls

| *"After every sales call, it is already improving. This is amazing."*

Key Patterns Detected (First Week)

Channel partnerships: New growth lever identified

Community members who watch masterclass close in 24 hours

Multi-agent orchestration: New high-ticket anchor (\$150K and \$35K deals mentioned)

Proposal engine improvement: fastest closes come from community members already using tools

PROMPT:

```
Can you send how these three alerts look like in Telegram? Next up I will do a Resend integration where you have the ability to send email. Eventually we will change this to both Telegram alert and email alert. But for now send me how each of these would look like.
```

BUILT: Three autonomous cron jobs: Meeting Analyzer (2hr), Hot Lead Detection (real-time), Daily Briefing (daily)

SECTION 17: SELF-IMPROVING AI SDR

The SDR (Sales Development Representative) is the crown jewel of the system. It combines meeting intelligence with outreach automation to create a self-improving sales engine.

SDR Architecture

ICP Builder: Defines your Ideal Customer Profile from meeting data

Campaign Creator: Creates outreach campaigns (manual CSV or automated scraping)

AI Customization: Multiple levels of AI-generated personalization

Self-Improvement: Learns from email open rates, click rates, and responses

PROMPT:

```
Wewillcallthisaself-improvingAISDR.Theideaais:
1.ICPBuilder-buildidealcustomerprofilesfrommeeting data
2.CampaignCreator-createcampaignswithmanualCSV upload

    orautomatedleadscraping
3.AICustomizationLevels-howmuchAIpersonalization
4.Automatedcampaigns-Apifyscraperintegration

Thekey:weuseResendforemail(100emails/dayfree).
Wecreateawebhooktotrackopenrate,bouncerate, clickrate-
everything.Theagentwilllearnwhat's
workingandimprovecampaignseverysingleday.

Keep this on hold. Everything I told you is important
and we are going to call it step two. But first I want
you to give me a prompt for my builder agent that can
create a skills page.
```

The SDR Day/Night Cycle

Time	System	Action
Daytime (9-5)	Campaign Manager	Sends emails, respects sending windows
Daytime	Resend Webhooks	Tracks opens, clicks, bounces, deliveries
9 PM	SDR Intelligence	Analyzes all day's data, finds patterns

Time	System	Action
Night	Skill Patches	Updates outreach copywriting skill
Next Day	Campaign Manager	Uses improved copy from overnight analysis

"We are talking about an AI that can improve itself over and over every single day based on results. This is amazing."

BUILT: Self-improving SDR architecture: ICP Builder + Campaign Creator + Overnight Intelligence + Skill Patches

SECTION 18: EMAIL SYSTEM — RESEND INTEGRATION

Why Resend

Resend is the best email service for developers. It offers 100 emails per day for free, has excellent webhook tracking, and integrates cleanly with Supabase edge functions.

Setup Steps

1. Sign up at resend.com
2. Copy your API key
3. Add to `.env.local`: `RESEND_API_KEY=your_key`
4. Go to Domains > Add Domain
5. Use a subdomain (e.g., `updates.yourdomain.com`) — NOT your main domain
6. Add the DNS records (MX, TXT, CNAME) to your domain provider
7. Wait ~20 minutes for verification
8. Add Resend webhook in Mission Control for event tracking

```
# .env.local additions
RESEND_API_KEY=re_XXXXXXXXXXXXX

#Domain setup example
#Main domain: growthcreators.ai
#Email subdomain: updates.growthcreators.ai
#Sender: hermes@updates.growthcreators.ai
```

PROMPT:

```
I added a Resend API key in .env.local. Use the domain
updates.growthcreators.ai to set up our email sending.
The sender name should be Hermes Mission Control.
Also create a dedicated email function that agents can
use to send emails via Resend. And configure the Resend
webhook for tracking opens, clicks, bounces,
deliveries.
```

Email Safety Rules

Hermes was configured with a critical safety rule: never send emails without explicit user authorization. Even when Resend was configured, the agent confirmed before sending.

"You explicitly asked — I thought we are only setting limits on Gmail. Resend is all set up but you see how it actually made it a hard rule: no sending emails without authorization."

76 Endpoints Deployed

After the email system was integrated, the total endpoint count reached 76 — covering meetings, webhooks, functions, outreach, email, skills, memory, and system management.

BUILT: Resend email system deployed. 76 API endpoints total. Webhook tracking for opens, clicks, bounces, deliveries.

SECTION 19: NURTURE SEQUENCES

Nurture sequences are proactive email campaigns triggered when someone books a discovery call. The goal: maximize show-up rates by sending personalized, value-driven emails before the meeting.

Cal.com Integration

Cal.com is the scheduling tool. When someone books a discovery call, a webhook fires to Mission Control, which triggers the nurture sequence.

```
# Add Cal.com API key to Hermes environment
echo 'CAL_API_KEY=your_cal_api_key' >> ~/.hermes/.env

#Register Cal.com webhook in Mission Control
#Events: booking.created, booking.cancelled
```

PROMPT:

```
I want to add email nurture sequences. Now that you can
see my calendar, you know as soon as someone books a
discovery call.

When someone books a discovery call, I want to subscribe
them to an email sequence. This is a proactive nurture
sequence. If they have a week-long wait, we send alternate
days. Mandatory: 24 hours before and 1 hour before the call.
Two emails go out regardless. Cal.com also sends its own
notification, so don't duplicate that.
```

```
Important: Only trigger for 'Discovery call with Mani'
not for follow-up calls or other meeting types.
The goal of this nurture sequence is to make them SHOW UP
to the call. I will give you case studies and inputs.
```

Nurture Email Sequence

Email	Timing	Content
Welcome	Immediately on booking	Confirmation + what to expect + case study
Value #1	2 days before (if time permits)	Relevant case study for their industry

Email	Timing	Content
Value #2	Alternate days	More social proof and success stories
24hr Reminder	24 hours before	Preparation tips + agenda preview
1hr Reminder	1 hour before	Quick reminder with meeting link

Critical Rule: Discovery Calls Only

```
# ONLY trigger nurture for discovery calls
# Event name must contain: 'Discovery call with Mani'
# If you just see '30 minutes with Mani' – DO NOT trigger
# This prevents nurture sequences for internal or
# follow-up meetings
```

Cal.com Webhook Fix

A common issue: the Cal.com webhook was initially pointed to the Fathom endpoint, which expects a different payload format. A dedicated handler was needed:

PROMPT:

```
Cal.com webhook is pointing to the Fathom endpoint which
expects Fathom payloads. Cal.com sends a completely
different format. We need a dedicated handler.
Build a Cal.com-specific webhook endpoint.
```

BUILT: Nurture sequence system: auto-triggered on booking, personalized emails, show-up rate optimization

SECTION 20: CAMPAIGN BUILDER

Manual Campaign: CSV Upload

The campaign system supports two modes. Manual mode accepts a CSV file with lead data. The agent provides the expected format:

```
# CSV Format (minimum required fields)
email,name,company,industry,location
john@acmedental.com,John Smith,Acme Dental,Healthcare,Vancouver
sarah@lawfirm.ca,Sarah Jones,Jones Law,Legal,Toronto
```

PROMPT:

```
Go to Outreach > Campaigns > Create New Campaign
Upload a CSV file with your leads.
Format: email is required, name/company/industry optional.
The AI will personalize each email based on available data.
```

AI Personalization Levels

Campaigns support multiple levels of AI customization:

- Level 1: Template-based (same email, merge fields only)
- Level 2: Industry-personalized (customized per industry vertical)
- Level 3: Fully AI-written (unique email per lead based on research)

Campaign Management

The outreach dashboard shows all campaigns with status tracking:

- Ready for Review: Campaigns awaiting human approval
- Processing: AI is generating personalized emails
- Active: Currently sending (respects 9-5 sending window)
- Completed: All emails sent, tracking results

BUILT: Campaign builder with CSV upload, AI personalization, and status tracking dashboard

SECTION 21: AI ACTIVITY LOG

The AI Activity Log gives you complete visibility into what your agents are doing. Every check-in, every action, every upcoming task is logged and visible in Mission Control.

Memory Page Builder

PROMPT:

```
Build an agent memory dashboard page in Mission Control.
Show persistent memory, daily logs, and snapshots.
The agent syncs every 3 hours.
```

Daily logs should show:

- What was built/completed
- Decisions made
- Things learned
- Context from interactions
- Active projects
- Blocked items
- Skills updated

What the Activity Log Captures

Category	Details
Check-ins	Agent status reports every few hours
Actions Taken	API calls made, skills run, data analyzed
Skills Updated	Which skills were patched and why
Meetings Analyzed	Count and key findings
Campaigns	Status of active outreach campaigns
Errors	Any failures and how they were resolved
Memory Sync	Last sync time and what was preserved

Real Example

From the activity log after Day 1:

```
Day 1 Complete
- Built complete Mission Control with outreach,
  memory, and feedback systems
- Massive integration day
```

- Skills updated: 7 new, 3 patched
- Memory synced: 3 hours ago
- Next sync: 2 hours

BUILT: Complete agent activity tracking: check-ins, actions, skills, meetings, campaigns — all visible in dashboard

SECTION 22: AUTOMATED CAMPAIGN CREATION

Apify Scraper Integration

For automated lead generation, the system integrates with Apify — a web scraping platform. With a \$5 free credit, you can scrape Google Maps, LinkedIn, and other platforms for leads.

```
# Add Apify API key to .env.local
APIFY_API_KEY=your_apify_key
```

PROMPT:

```
I want to build an automated campaign creator / ICP
builder feature in the outreach section.
When creating an automated campaign, use Apify to search
for leads. For proof of concept, use their free scrapers.
```

```
The ICP builders should ask:
- Which vertical (e.g., law firms, accounting firms) -
- Geographic location (dropdown: US, Canada, cities) -
- Max results to scrape
Once scraped, create a campaign. Hermes agent will take
over from there. Build this feature and update the
integration guide with anything new.
```

Automated Campaign Flow

1. User enters ICP criteria (vertical, location, quantity)
2. System selects appropriate Apify scraper (Google Maps, etc.)
3. Scraper runs and returns lead data (30-60 seconds)
4. Leads are formatted and stored in the campaign
5. AI generates personalized outreach for each lead
6. Campaign enters 'Ready for Review' status
7. Human approves -> emails are sent during business hours

PROMPT:

```
Run a real Google Maps scrape for accounting firms in
Vancouver, BC. Next: Canada, US.
Max results: 20. Start scraping.
```

Webhook Event Tracking

When emails are sent via Resend, webhook events track engagement in real time:

45 events tracked in the demo: clicks, opens, deliveries

Each open, click, and bounce is logged with timestamp

Message content is stored for A/B analysis

Data feeds into the overnight SDR intelligence cron

BUILT: Automated lead scraping via Apify -> AI-personalized campaigns -> Resend delivery -> webhook tracking

SECTION 23: CAMPAIGN APPROVAL — HUMAN-IN-THE-LOOP

No campaign sends without human approval. This is the critical safety mechanism that prevents the AI from sending emails without oversight.

15-Minute Polling

```
#Campaign Approval Cron
Schedule: Every 15 minutes during business hours (9-5)
Action: Check for approved campaigns
If approved: Begin sending (respect rate limits)
If not approved: Skip and check again in 15 minutes
Daily limit: 50 emails per day
```

Approval Flow

1. Hermes creates a campaign with AI-generated emails
2. Campaign appears as 'Ready for Review' in Mission Control
3. Hermes sends a Telegram notification: 'Campaign ready for review'
4. Human reviews emails in the dashboard
5. Human clicks 'Approve' button
6. Next 15-minute poll picks up the approval
7. Emails begin sending during the 9-5 window
8. All events (opens, clicks, bounces) tracked via Resend webhook

PROMPT:

```
We need an approve button. You need to be polling every
15 minutes for approvals. When a campaign is approved,
start sending. Respect the 9-5 sending window.
Daily limit: 50 emails.
```

Sender Configuration

PROMPT:

```
Right now I'm updating the sender to 'Mani Kanasani'
and the email to updates.growthcreators.ai.
That way they receive email from me — not 'Hermes
Mission Control'. Make this change in the builder.
```

BUILT: Human-in-the-loop approval system: 15-minute polling, approve-to-send pipeline, rate-limited delivery

SECTION 24: SKILL BUILDER

The Skill Builder transforms skill creation from something that only happens during agent sessions into an on-demand capability. Drop a URL, paste a transcript, and Hermes extracts structured skills from any content.

Skill Builder Flow

- 1. Paste a URL, YouTube transcript, article, MCP documentation, or PDF
- 2. Pick the category (sales, technical, operations, etc.)
- 3. Add notes: 'Focus on X' or 'Ignore Y'
- 4. Click 'Submit for Processing'
- 5. AI extracts all teachableknowledge
- 6. Identifies pitfalls, bestpractices, step-by-step processes
- 7. Generates structuredskillproposals
- 8. Review each proposal:Approve, Edit, Reject, or Skip
- 9. Approved skills go liveinthe skill library

PROMPT:

```
Thenext feature I'm envisioning is improving or adding
another tab into our skills page. This is called Skill
Builder.

What this does: it takes a URL or a YouTube transcript
or an article, MCP, any kind of documentation. Once we
paste it, we extract a skill from it.
You will be the one who is processing. We will just
enter and submit for review. AI will process and update
the output.
What it can handle: URLs, transcripts, products, PDFs.
Different topics should generate multiple skills - each
gets their own approval.
```

Skill Categories

Category	Example Skills
Sales	Proposal templates, discovery frameworks, objection handling

Category	Example Skills
Technical	API integration patterns, deployment checklists
Operations	Meeting analysis, client onboarding, project management
Outreach	Email copywriting, ICP definition, nurture sequences
Research	Market analysis, competitor tracking, trend detection

BUILT: Skill Builder UI: URL/transcript input, AI extraction, proposal review, approve-to-library pipeline

EVERYTHING BUILT: FINAL RECAP

In 4 hours, the course built a complete AI-powered agency operating system. Here is everything that is now running:

Mission Control Dashboard

- Glassmorphic React + Vite + Tailwind web application
- Supabase backend with 76+ API endpoints
- Integration Guide for any AI agent to control the system
- Secure API key management via .env.local

Meeting Intelligence Engine

- Fathom webhook: real-time meeting import on call completion
- Transcript analysis with AI-generated insights
- Deal probability, engagement scores, sentiment analysis
- Human feedback mechanism for each meeting
- Self-improving proposal engine (4 patches in first cycle)
- Self-improving discovery framework (3 patches in first cycle)

Outreach & SDR System

- ICP Builder from meeting data and manual input
- Manual campaign creation via CSV upload
- Automated campaign creation via Apify scraping
- AI-personalized email generation (3 customization levels)
- Resend email delivery with full webhook tracking
- Human-in-the-loop approval system (15-minute polling)
- Overnight SDR intelligence (9 PM daily analysis)

Nurture Sequences

- Cal.com webhook integration for booking detection
- Automatic nurture emails for discovery call bookings
- 24-hour and 1-hour reminders (mandatory)
- Personalized case studies and value emails
- Discovery-call-only trigger (no false positives)

Agent Intelligence

- Three autonomous cron jobs running 24/7
- Skill library with self-improving skills
- Skill Builder for URL/transcript skill extraction
- Agent memory dashboard with daily logs
- Activity log with check-ins, actions, and learnings
- Google Workspace integration for calendar and email access

3 Cron Jobs Summary

Cron Job	Schedule	Purpose
Meeting Analyzer	Every 2 hours	Analyze transcripts, generate insights, patch skills
Hot Lead Detection	Real-time	Telegram alerts for high-probability deals
Daily Briefing	Daily	Complete business intelligence summary via email + Telegram
SDR Intelligence	9 PM nightly	Analyze campaign data, patch copywriting skills
Campaign Sender	Every 15 min (9-5)	Check for approved campaigns, send emails
Memory Sync	Every 3 hours	Backup agent memory and daily logs

Complete System Flow

```

NEWMEETING RECORDED
|->Fathom webhook fires
|->Meeting saved to Supabase
|->Meeting Analyzer cron picks it up
|->Transcript analyzed, insights generated
|->Skills patched (proposal, discovery framework)
|->Hot lead? -> Telegram alert
|->Daily briefing includes it

NEWBOOKING ON CAL.COM
|->Cal.com webhook fires
|->Nurture sequence triggered
|->Personalized emails sent before the call
|->Show-up rate optimized

OUTREACH CAMPAIGN
|-> ICP defined (manual or from meeting data)

```

```
|-> Leads scraped (Apify) or uploaded (CSV)
|-> AI writes personalized emails
|-> Human approves campaign
|-> Emails sent (9-5 window, 50/day limit)
|-> Opens/clicks/bounces tracked via Resend
|-> 9 PM: SDR Intelligence analyzes results
|-> Copywriting skill patched overnight
|-> Next day: better emails automatically
```

"This is just the beginning. Once you land, tell me more to tweak. This is not the end result — we are going to improve this significantly."

BUILT: COMPLETE SYSTEM: Meeting Intelligence + Self-Improving SDR + Nurture Sequences + 6 Cron Jobs + 76 Endpoints + Skill Builder

COMPLETE PROMPT REFERENCE

Every major prompt from the course, organized by category. Copy these directly into your AI agents.

Setup & Configuration Prompts

PROMPT:

```
I am building a Hermes Mission Control application.  
I need to give you the API key for fetching my Fathom  
meetings. The UI should be futuristic with a blue theme.  
Tell me the secure way to share API keys – give me a  
.env.local file for me to update.
```

PROMPT:

```
Configure Telegram with mybot token and chat ID.  
Set up the gateway as a clean single instance.  
Test the connection – send a test message.
```

PROMPT:

```
Check if you have Supabase CLI installed.  
Authenticate using: supabase login  
Or set the access token as environment variable.
```

Building Prompts

PROMPT:

```
Make the Fathom API calls. Import all calls from the  
past 45 days to Supabase. Create a new table with:  
date, meeting name, transcript, summary, action items,  
company name, emails, type of call. Follow design.md.
```

PROMPT:

```
Text is too tiny. Sidebar menu is ugly and not readable.  
Make text brighter/whiter. Create a dedicated page for  
each meeting with sections for Hermes to add notes,  
generate proposals, and generate custom action items.
```

PROMPT:

```
Padding in corners is off. Logo has empty space.  
Cards are overlapping. Tell me explicitly which file to  
edit – don't say 'add padding', say 'open App.tsx and
```

```
wrap the top-level div with a padding container.'
```

PROMPT:

```
Create a webhooks page with: webhook list with status
cards, add webhook modal, webhook detail page, function
grid, Supabase edge functions. Come up with a detailed
plan and checklist.
```

PROMPT:

```
Make it so data received from Fathom webhook is
automatically saved to meetings. For now just save
that as an automatic behavior.
```

PROMPT:

```
Create an integration page with all API calls an agent
can do. Everything a human can do, AI should be able
to do via edge functions. Create 19+ endpoints. As soon
as I copy the integration guide to an AI agent, it should
be able to use the entire application.
```

Hermes Agent Prompts

PROMPT:

```
We will be working off of Hermes Mission Control.
I will give you the integration guide. Save it first.
Create skills for proposals, discovery calls, objections.
Suggest features you would like to see.
[Paste 287 lines of Integration Guide]
```

PROMPT:

```
I will give you two files: our proposal engine and
discovery call framework. These are directly responsible
for $50,000+ in the past 30 days. Rewrite your answer
using these as context.
```

PROMPT:

```
Add a feedback mechanism at the insights level for
each meeting page. I need a place to communicate – add
notes for each meeting and for you to work off of
that.
Give me a prompt for my builder agent.
```

PROMPT:

```
Send how the three alerts (meeting analyzer, hot lead,
```

```
daily briefing) look in Telegram. Next we will do a
Resend integration for email alerts too.
```

PROMPT:

```
Give me a status update of everything we built so far
and current workflow. What do I see every day? What
can I expect? Send me the three test briefs through
Resend. My email: [email].
```

SDR & Outreach Prompts**PROMPT:**

```
We will call this a self-improving AI SDR. Components:
1. ICP Builder — from meeting data
2. Campaign Creator — manual CSV + automated scraping
3. AI Customization Levels
4. Resend for email (100/day free)
5. Webhook tracking for open/bounce/click rates
6. Learn from results, improve campaigns daily
```

PROMPT:

```
Give me a prompt for my builder agent that can create
a skills page. Whatever skills you have, populate that
page. Show me every day what you add and improve.
A log of how far skills came from day one.
```

PROMPT:

```
I added Resend API key in .env.local. Use the domain
[subdomain] to set up email sending. Sender name:
'Hermes Mission Control'. Create a dedicated email
function for agents. Configure Resend webhook.
```

PROMPT:

```
Add email nurture sequences. When someone books a
discovery call, subscribe them to an email sequence.
Proactive nurture: alternate days if week-long wait.
Mandatory: 24h and 1h before the call.
Only trigger for 'Discovery call with Mani'.
Goal: make them show up to the call.
```

PROMPT:

```
Cal.com webhook is pointing to Fathom endpoint which
expects Fathom payloads. Cal.com sends a different
format. We need a dedicated handler.
```

PROMPT:

```
Build an automated campaign creator / ICP builder
using Apify. Ask for vertical, geographic location,
max results. Scrape leads, create campaign.
Use free scrapers. Update integration guide.
```

PROMPT:

```
We need an approve button. Poll every 15 minutes
for approvals. When approved, start sending.
Respect 9-5 sending window. Daily limit: 50 emails.
```

PROMPT:

```
Add a Skill Builder tab. Takes URL, YouTube
transcript, article, MCP docs, or PDF. Extracts
structured skills. Submit for review. AI processes and
proposes skills. Each topic generates separate skills
with individual approval. Approve -> skill goes live
in library.
```

PROMPT:

```
Create a nightly cron at 9 PM. Analyze all campaign
data from the day. Find patterns in opens, clicks,
bounces. Patch the outreach copywriting skill.
Tomorrow's sends use improved copy automatically.
```

PROMPT:

```
Build agent memory dashboard. Show persistent memory,
daily logs, snapshots. Sync every 3 hours.
Daily logs: what built, decisions, things learned,
context, active projects, blocked items, skills updated.
```

PROMPT:

```
Make sure the integration guide is up to date with all
new info – the outreach SDR components that you built.
We have 77 endpoints total but the guide may not reflect
all of them. Update and give the updated guide to
Hermes.
```

PROMPT:

```
I am hoping to give you more visibility to what's
happening in my business. You need access to my Google
Calendar. How can we set that up with the least friction?
Can I give you an API key so you can set up permissions?
```

PROMPT:

```
Makesureyou're not sending me emails unless I  
specifically say so. Do not send emails without  
explicit authorization — regardless of the  
tool.
```

PROMPT:

```
Updatethesenderto'Mani Kanasani' and email to  
updates.growthcreators.ai. Recipients should see  
email from me, not from 'Hermes Mission Control'.
```

BUILT: 28 major prompts documented. Copy directly into your AI agents.

TERMINAL COMMANDS REFERENCE

Every terminal command used throughout the course:

Installation

```
#Install Hermes Agent (macOS/Linux)
curl -fsSL
https://raw.githubusercontent.com/NousResearch/hermes-agent/main/install.sh
|bash

#Install Hermes Workspace
gitclone https://github.com/NousResearch/hermes-workspace.git
cdhermes-workspace
npminstall -g pnpm
pnpminstall
pnpmdev
```

API Key Management

```
#Add APIkeyssecurely (never paste in chat)
echo 'RESEND_API_KEY=re_XXXXXX' >> .env.local
echo 'CAL_API_KEY=cal_XXXXXX' >>
~/.hermes/.env echo
'APIFY_API_KEY=apify_XXXXXX' >> .env.local
```

Supabase

```
#Login to Supabase CLI
supabase login
#Set access token
export SUPABASE_ACCESS_TOKEN=your_token
#Deploy edge functions
supabase functions deploy
```

Google Cloud Setup

```
# Google Cloud Console Steps:
1. Create a new project
2. Enable APIs: Gmail, Calendar, Drive, Sheets, Docs
3. Create credentials (OAuth 2.0 Client ID)
4. Application type: Desktop Application
5. Download the JSON credentials file
6. Provide to your agent for configuration
```

Telegram

```
# BotFather commands
/newbot      # Create new bot
/setname     # Change bot name
/setdescription # Set bot description

#Get your chat ID
#Message @userinfobot in Telegram
```

KEY CONCEPTS GLOSSARY

Concept	Definition
Trajectory	A complete recording of everything Hermes did during a task — every API call, decision, and tool used
Skill	A reusable SOP (Standard Operating Procedure) that Hermes writes for itself after completing a task
Knowledge Fortress	Five-layer context system: Documentation, Security, Technical, Strategy, Knowledge
Agent Taxonomy	Three agent roles: Orchestrator (OpenClaw), Builder (Claude Code), Executor (Hermes)
Integration Guide	API documentation that enables any AI agent to control Mission Control
AI-Fi	Making every feature API-accessible so both humans and agents can use it
Plan Mode	Claude Code feature that creates a detailed plan before building — better results for complex tasks
Edge Functions	Supabase serverless functions that serve as API endpoints for agent access
Cron Job	Scheduled task that runs automatically at set intervals (e.g., every 2 hours)
Nurture Sequence	Automated email series triggered by a booking event to maximize show-up rates
ICP	Ideal Customer Profile — defines the characteristics of your best potential clients
SDR	Sales Development Representative — the agent role that handles outreach and lead qualification
Human-in-the-Loop	System design where AI proposes actions but humans approve before execution
Self-Improving	The agent's ability to analyze results and patch its own skills automatically
Mission Control	Custom web dashboard connecting all agent operations in one screen
ClawBuddy	Mani's existing Mission Control system — the reference implementation

Concept	Definition
Memory Sync	Periodic backup of agent memory and daily logs (every 3 hours)
Sending Window	Business hours (9 AM - 5 PM) during which outreach emails are sent
Skill Patch	An incremental improvement to an existing skill based on new data or feedback
design.md	Configuration file that captures visual preferences for consistent UI generation

TECH STACK REFERENCE

Component	Technology	Purpose
Frontend	React + Vite + Tailwind CSS	Mission Control web application
Backend	Supabase (PostgreSQL + Edge Functions)	Database, API, and serverless functions
Builder Agent	Claude Code (Opus 4.6) / Codex (GPT 5.4)	Writes code and builds the system
Executor Agent	Hermes Agent (Nous Research)	Runs operations, learns, improves Multi-platform coordination
Orchestrator	OpenClaw (optional)	Auto-records meetings, provides transcripts
Meeting Notes	Fathom	100 free emails/day with webhook tracking
Email	Resend	Discovery call booking with webhook events
Scheduling	Cal.com	Google Maps scraper for automated ICP building
Lead Scraping	Apify	Agent alerts and communication
Messaging	Telegram	UI/UX prototyping and inspiration
Design	Lovable / Gemini 3.1 Pro / Dribbble / v0.dev	Email subdomain DNS configuration
DNS/Domain	GoDaddy / Namecheap / Hostinger	

TIMESTAMP INDEX

Quick reference for jumping to specific sections in the video:

Timestamp	Section	Topic
0:00:00	Intro	Stripe revenue reveal, what we're building
0:01:33	Section 1	What is Hermes — trajectories, skills, self-improvement
0:04:21	Section 2	Cost & model options — free to \$20/month
0:06:44	Section 3	Three levels: Terminal, Workspace, Custom Build
0:08:26	Section 4	Install — one command setup + Telegram
0:15:28	Section 5	Hermes Workspace — dashboard walkthrough
0:41:08	Section 6	Knowledge Fortress — 5-layer context system
0:43:39	Section 7	Agent Taxonomy — Orchestrator/Builder/Executor
0:44:37	Section 8	OpenClaw vs Hermes — honest comparison
0:51:44	Section 9	Migration Guide — switching or running both
0:54:20	Section 10	Build Begins — Mission Control architecture
0:57:00	Section 11	Design & Theming — screenshot-driven design
1:19:10	Section 12	Fathom Webhook — meeting import automation
1:39:17	Section 13	AI-First Architecture — 19 endpoints
1:45:12	Section 14	Hermes Takes Over — first autonomous operation
1:50:39	Section 15	Meeting Intelligence — insights & deal scoring

Timestamp	Section	Topic
2:00:37	Section 16	Cron Jobs — 3 automated intelligence systems
2:24:43	Section 17	Self-Improving SDR — outreach architecture
2:41:36	Section 18	Email System — Resend setup (76 endpoints)
3:00:52	Section 19	Nurture Sequences — Cal.com + show-up optimization
3:11:27	Section 20	Campaign Builder — CSV upload + AI personalization
3:14:47	Section 21	Activity Log — agent tracking dashboard
3:32:19	Section 22	Automated Campaigns — Apify scraper integration
3:42:34	Section 23	Campaign Approval — human-in-the-loop
3:47:27	Section 24	Skill Builder — URL to skill extraction
4:05:09	Final	Everything Built — complete system recap