INSTRUCTIVO COMPLETO — Stack Gastón (PC casa ↔ PC oficina)

> Generado: 2026-04-27 por Sophie (Claude). Vivo en `F:\gaston-workspace\INSTRUCTIVO_OFICINA.md` .

■ Acceso público (este doc + memorias) — desde cualquier dispositivo

Mirror	URL	Auth	Notas
Cloudflare Worker	https://docs.n8n-nocodedb.info/	público	TLS automático, siempre online (independiente PC casa)
GitHub repo privado	https://github.com/grancobud/gaston-stack-docs	login `gh`	versionado, push para actualizar
PC casa local	`F:\gaston-workspace\INSTRUCTIVO_OFICINA.*`	físico	fuente de verdad
LightRAG (Hermes)	workspace `gaston`	local	consultable vía agente
token-savior	memoria `instructivo-oficina-master-doc`	local	importance 10

Endpoints del Worker:

- `/` — portal con links
- `/instructivo.pdf` — el PDF (50 KB)
- `/instructivo.md` — el MD editable (38 KB)
- `/memorias.txt` — export de las 222 memorias del token-savior (156 KB)

Este documento tiene 3 cosas:

1. Snapshot completo del stack actual de la **PC casa** (qué hay, dónde, cómo se prende).
2. **Camino A**: cómo conectarte desde la **PC oficina a la PC casa** vía Tailscale (rápido — usás todo lo de casa remoto).
3. **Camino B**: cómo replicar todo el stack en la **PC oficina** desde cero (más trabajo — independencia total).

Recomendación: **arrancar con Camino A** (1 hora) y migrar a Camino B sólo si necesitás operar offline o cuando tu RTX 5090 esté en oficina.

0. TL;DR — Qué tenés que saber sí o sí

Cosa	Dónde está
Workspace principal	`F:\gaston-workspace\`
Memoria persistente	token-savior DB (SQLite) — accesible vía MCP

Cosa	Dónde está
Identidad/personalidad agente	"Sophie" — ver `IDENTITY.md` y `SOUL.md`
Stack containers	`F:\gaston-workspace\ai-stack\docker-compose.yml`
Polymarket-weather MCP (NUEVO)	`F:\gaston-workspace\polymarket-weather-mcp\`
Tailnet	`tail89f35a.ts.net`, owner `gastonrpersoglia@gmail.com`
PC casa Tailscale	`pc-casa.tail89f35a.ts.net` → `100.91.248.66`
OpenWebUI (chat)	http://localhost:3000 — auth admin
LightRAG (knowledge graph)	http://localhost:9622 (workspace `gaston`)
Hermes Agent	http://localhost:8642 (FastAPI shim OpenAI-compat)
MCPO Linux/Docker	http://localhost:8770 (Bearer `mcpo-gaston-secret`)
MCPO Windows host	http://localhost:8771 (mismo Bearer)

1. Snapshot del stack actual (PC casa)

1.1 Hardware (actual)

- CPU: Ryzen 7 X3D
- GPU: RTX 5090 32GB VRAM
- RAM: 64GB
- OS: Windows 10 Pro 10.0.19045
- Disco workspace: F:\

1.2 Containers Docker (todos en `127.0.0.1:*`, network `ai-net`)

Container	Image	Puerto host	Función
`openwebui`	`ghcr.io/open-webui/open-webui:0.9.2`	3000	UI principal de chat
`lightrag`	`ghcr.io/hkuds/lightrag:v1.4.15`	9622	Knowledge graph + RAG (workspace `gaston`)
`embedding-server`	`ghcr.io/huggingface/text-embeddings-inference:cpu-1.7`	8090	Embeddings local (sentence-transformers/all-MiniLM-L6-v2)
`hermes`	`gaston/hermes-shim:latest` (custom)	8642	Hermes Agent (FastAPI OpenAI-compat)
`mcpo`	`ghcr.io/open-webui/mcpo:main`	8770	Proxy OpenAPI para MCPs Linux/Docker
`time`	`mcp/time:latest`	-	MCP de fecha/hora

Container	Image	Puerto host	Función
`fetch`	`ghcr.io/stackloklabs/gofetch/server:1.0.3`	16388	MCP de HTTP fetch

Compose: `F:\gaston-workspace\ai-stack\docker-compose.yml` . Levantar: `cd F:\gaston-workspace\ai-stack && docker compose up -d` . Bajar: `docker compose down` (preserva volúmenes).

1.3 MCPO Windows host — proceso nativo en :8771

Por qué existe: hay MCPs Python que NO pueden correr dentro del container Linux (necesitan paths Windows, COM, exes locales). Por eso hay un MCPO host-side aparte.

- **Config**: `F:\gaston-workspace\ai-stack\mcpo-windows-config.json`
- **Launcher**: `F:\gaston-workspace\ai-stack\start-mcpo-windows.ps1`
- **Bearer key**: `mcpo-gaston-secret`
- **Logs**: `F:\gaston-workspace\ai-stack\mcpo-windows.log` y `.err`
- **MCPs registrados** (al 2026-04-27): tradingview-atila, tradingview-cdp, desktop-commander, excel, token-savior, telegram, ruflo, lightrag, MCP_DOCKER, hermes-admin, **polymarket-weather** (NUEVO).

Levantar: `powershell -ExecutionPolicy Bypass -File F:\gaston-workspace\ai-stack\start-mcpo-windows.ps1` . Reiniciar: matar el proceso que escucha en :8771 (`Get-NetTCPConnection -LocalPort 8771`) y relanzar el script.

1.4 MCPs custom (Python en host) — patrón

Nombre	Path	Tools
`cloudflare-mcp`	`C:\tools\cloudflare-mcp\`	87
`hermes-admin-mcp`	`C:\tools\hermes-admin-mcp\`	72
`mcpo-admin-mcp`	`C:\tools\mcpo-admin-mcp\`	29
`openwebui-admin-mcp`	`C:\tools\openwebui-admin-mcp\`	~70
`hostinger-mcp`	`C:\tools\hostinger-mcp\`	72
`tailscale-mcp`	`C:\tools\tailscale-mcp\`	66
`telegram-mcp`	`C:\tools\telegram-mcp\`	-
`polymarket-weather-mcp`	`F:\gaston-workspace\polymarket-weather-mcp\`	**100** (NUEVO)

Patrón: cada uno es un FastMCP `server.py` monolítico con `@mcp.tool()` . Se registran en `mcpo-windows-config.json` con `type: stdio` , `command: C:/Python314/python.exe` , `args: [path/to/server.py]` .

1.5 Polymarket-weather-mcp (proyecto nuevo)

Path: `F:\gaston-workspace\polymarket-weather-mcp\` **Tools**: 100 (86 raw JSON + 14 wrappers `_md` que devuelven Markdown legible para chat) **Categorías** (groups por prefix):

- `weather_*` (20) — NOAA METAR, GFS ensemble, ECMWF, anti-tampering CDG, ICAO mapping
- `pm_*` (18) — Polymarket Gamma + CLOB read
- `pmd_*` (6) — Data API: positions, leaderboard, P&L
- `chain_*` (6) — Polygonscan + known traders verificados
- `econ_*` (5) — FRED + BLS + consensus
- `negrisk_*` (4) — Negative risk arbitrage
- `strat_*` (16) — EV, Kelly, gap detection, scanners (gopfan2 / sharky / neobrother), signal engine
- `safety_*` (7) — bankroll guards, daily stop-loss, drawdown pause, blacklist
- `db_*` (8) — SQLite signals/trades/resolutions/stats
- `bayes_*` + `calib_*` (6) — Bayesian win rate, drift detection, sesgos por ciudad
- `config/health/cache` (3) — utilities

Modo: `paper` por default. Para activar live: setear `POLYMARKET_TRADING_MODE=live` en `.env` y trading real requiere `confirm=True` por llamada. **Bankroll**: $100 modo Aggressive (Half Kelly, 15% bankroll cap, $20 hardcap, daily stop -25%, drawdown pause -40%).

Wallet verificadas hardcoded en `core/chain/known_traders.py` (las únicas confiables):

- gopfan2: `0xf2f6af4f27ec2dcf4072095ab804016e14cd5817` ($1-2M, weather scan)
- Sharky6999: `0x751a2b86...` ($480-934K, HODL high-prob)
- Theo4: `0x56687bf...` ($22M, política)

■ ColdMath, 1pixel, meropi: datos de blogs/Medium SIN verificación on-chain — NO usar como benchmark.

1.6 Skills de Hermes / Claude Code

- Skills Claude Code: `C:\Users\Gaston\.claude\skills\` (incluye `polymarket-weather/SKILL.md` NUEVA)
- LightRAG indexa todas las skills (workspace `gaston`).
- Path doc Polymarket completa: `F:\Polymarket\polymarket_completo.md` (679+ líneas, incluye apéndice de investigación profunda).

1.7 Cloudflare Tunnel (acceso externo)

Dominio: `n8n-nocodedb.info` (zona en Cloudflare). Subdominios públicos del stack:

- `https://openwebui.n8n-nocodedb.info` → OpenWebUI
- `https://lightrag.n8n-nocodedb.info` → LightRAG
- `https://hermes.n8n-nocodedb.info` → Hermes API

Tunnel se administra vía MCP `cloudflare-admin` . Para servicios internos (MCPs admin, polymarket-weather) **NO usamos Cloudflare** — solo Tailscale (mejor seguridad).

1.8 Tailscale (red privada casa ↔ oficina)

Tailnet: `tail89f35a.ts.net` — account `gastonrpersoglia@gmail.com` . Devices al 2026-04-27:

- `pc-casa` (DESKTOP-D0PD9PT) Windows → `100.91.248.66`
- *PC oficina* (todavía no agregada)

Servicios accesibles desde otros nodos del tailnet (cuando agregás la oficina):

Desde otro nodo	URL
OpenWebUI	http://100.91.248.66:3000
LightRAG	http://100.91.248.66:9622
Hermes API	http://100.91.248.66:8642
MCPO Linux	http://100.91.248.66:8770
MCPO Windows	http://100.91.248.66:8771

■■ **Atención**: actualmente los puertos están bind a `127.0.0.1` en el compose. Para que sean alcanzables desde Tailscale necesitás cambiarlo a `0.0.0.0` O usar `tailscale serve` (recomendado, encripta y autentica). Ver sección 3.4.

2. Memorias guardadas (resumen — full export en token-savior)

Las que más vas a usar / no perder:

Sistema operativo + DNS

- `windows-dns-fix-applied` (importance 7): el 2026-04-27 cambié DNS Ethernet de Fibertel (`181.30.140.195/135`) a Cloudflare (`1.1.1.1, 1.0.0.1, 8.8.8.8`) porque Fibertel hijackea `*.polymarket.com` . Tailscale, WSL, Hyper-V no afectados.
- `windows-dns-original-backup` : backup del DNS original por si necesitás revertir.
- `polymarket-weather:isp-dns-hijacking` (importance 9): el problema base + 4 soluciones (DNS, WARP, Tailscale exit-node, hosts file).

Polymarket-weather-mcp

- `polymarket-weather:fee-changes` (importance 10, GUARDRAIL): 30-marzo-2026 weather markets ahora tienen fees dinámicos. 28-abril-2026 hay upgrade de stablecoin USDC.e → PolyUSD con ~1h downtime — re-aprobar contratos automáticamente.
- `polymarket-weather:risk-config-aggressive` (importance 9): config Aggressive para $100 (Half Kelly, 15% cap, $20 hardcap, daily stop -25%, drawdown -40%, drift threshold 50%, modo paper mín 7 días antes de live).
- `polymarket-weather:wallets-verified` (importance 8): 3 wallets confirmadas on-chain (gopfan2, Sharky6999, Theo4).
- `polymarket-weather:wallets-not-verified` (importance 7): ColdMath, 1pixel, meropi son blogs sin verificación, no usar como benchmark.
- `polymarket-weather:edge-erosion-status` : weather spreads <1¢ ahora. Oil/commodities sub-explotada post-Hormuz Q1 2026. NegRisk = 73% del profit en 8.6% oportunidades.

Arquitectura del stack

• `mcpo-architecture-dual-instance` (importance 9): 2 MCPOs — Docker en 8770 (config `mcpo-config.json`) y Windows host en 8771 (config `mcpo-windows-config.json`, launcher `start-mcpo-windows.ps1`). MCPs Python Windows van en el de 8771.

• `polymarket-mcp:architecture-sse-via-mcpo` (importance 9): patrón canónico para MCPs Python custom — stdio en mcpo-windows-config.json. Si tenés que hacer SSE, usar `TransportSecuritySettings(allowed_hosts=["*"])` para evitar HTTP 421 desde container.

• `openwebui-tool-server-add-via-db` (importance 8): cómo agregar tool servers a OpenWebUI editando `webui.db` directamente cuando el MCP admin no está disponible.

• `openwebui-function-calling-native` (importance 8): `models.default_params.function_calling` debe ser `native` (no `default`) para que los modelos procesen tool outputs en lenguaje natural.

• `ai-stack docker-compose optimizado 2026-04-27` + `AI Stack Docker Gaston — arquitectura completa 2026-04-24`: snapshots del compose y arquitectura.

Hermes

• `Hermes shim ES Hermes Agent real (no proxy)` (importance 9): el container Hermes corre el agente real (no es un proxy a OpenRouter). FastAPI shim en :8642.

• `6 admin MCPs custom + skills`: hermes-admin, mcpo-admin, openwebui-admin, cloudflare-admin, hostinger-admin, tailscale-admin son los MCPs custom de admin que escribimos.

Cuentas externas

• `Cloudflare setup completo (n8n-nocodedb.info)`: zona, tunnels, subdominios. Token guardado.

• `Hostinger account state`: VPS, dominios, snapshots.

• `Tailscale tailnet state`: device list, ACLs.

• `CannTrace Supabase YA implementado`: proyecto `sqdqvhjlmdweuuncwlfb`. URGENTE: 443MB/500MB usado.

3. CAMINO A: Conectarte desde oficina a PC casa (vía Tailscale)

Mejor opción si: querés operar de inmediato sin replicar todo, usando los modelos, GPU, MCPs y data de la PC casa.

3.1 En PC oficina — Instalar Tailscale (5 min)

1. Bajá Tailscale para Windows: https://tailscale.com/download/windows
2. Instalá. Login con la misma cuenta `gastonrpersoglia@gmail.com`.
3. Verificá que aparezca un nuevo device en https://login.tailscale.com/admin/machines (debería ser `pc-oficina` o similar).

Desde la PC oficina con Tailscale activo, podés hacer ping a `pc-casa.tail89f35a.ts.net` o `100.91.248.66`. Si responde, estás dentro del tailnet.

3.2 En PC casa — Exponer servicios al tailnet

Por seguridad, los puertos del compose hoy están bind a `127.0.0.1` (solo accesibles desde la propia PC). Hay que exponerlos al tailnet. Dos opciones:

Opción 3.2.a — `tailscale serve` **(RECOMENDADA)** Tailscale serve agrega TLS automático + autenticación + sin abrir nada al internet público.

```
# OpenWebUI
tailscale serve --bg --https=443 --set-path=/ http://localhost:3000

# LightRAG (puerto custom)
tailscale serve --bg --https=8443 --set-path=/ http://localhost:9622

# Hermes API
tailscale serve --bg --https=8642 --set-path=/ http://localhost:8642
```

Después acá:

- OpenWebUI desde oficina: `https://pc-casa.tail89f35a.ts.net/`
- LightRAG desde oficina: `https://pc-casa.tail89f35a.ts.net:8443/`
- Hermes desde oficina: `https://pc-casa.tail89f35a.ts.net:8642/`

Opción 3.2.b — Cambiar bind a Tailscale IP en docker-compose Editar `F:\gaston-workspace\ai-stack\docker-compose.yml` y cambiar `127.0.0.1:3000:8080` a `100.91.248.66:3000:8080` (idem para los demás). `docker compose up -d` .

Less recommended: pierde el TLS automático.

3.3 Acceso a MCPs desde oficina

Para que la PC oficina pueda invocar MCPs de Windows host (MCPO :8771):

```
# En PC casa
tailscale serve --bg --https=8771 --set-path=/ http://localhost:8771
```

Después en PC oficina, en cualquier app que consuma OpenAPI con bearer:

- URL: `https://pc-casa.tail89f35a.ts.net:8771/<mcp_name>/openapi.json`
- Auth: `Bearer mcpo-gaston-secret`

3.4 Configurar Claude Code en PC oficina apuntando a casa

Si querés que Claude Code corra en oficina pero use los MCPs de casa:

Editá `C:\Users\<vos>\.claude\settings.json` en oficina, agregá:

```
{
  "mcpServers": {
    "polymarket-weather": {
      "type": "sse",
      "url": "https://pc-casa.tail89f35a.ts.net:8771/polymarket-weather/sse",
      "headers": { "Authorization": "Bearer mcpo-gaston-secret" }
    }
    // ... agregar los que necesites
  }
}
```

3.5 Verificación final

Desde oficina:

```
# Ping al tailnet
ping pc-casa.tail89f35a.ts.net

# OpenWebUI (browser)
start https://pc-casa.tail89f35a.ts.net/

# Polymarket-weather MCP (curl)
curl -H "Authorization: Bearer mcpo-gaston-secret" `
     https://pc-casa.tail89f35a.ts.net:8771/polymarket-weather/openapi.json
```

4. CAMINO B: Replicar TODO el stack en oficina (independiente)

Hacelo cuando: querés operar offline en oficina, o cuando la 5090 esté allá, o cuando casa se apaga.

4.1 Pre-requisitos en PC oficina

Software	Versión	Cómo
Windows 10/11	-	-
Docker Desktop	latest	https://www.docker.com/products/docker-desktop/
Python	3.14 (mismo que casa)	https://www.python.org/downloads/
Git	latest	https://git-scm.com/
Node.js	LTS	https://nodejs.org/ (para `npx`)
uv (uvx)	latest	`pipx install uv` o https://docs.astral.sh/uv/
PowerShell 7	latest	`winget install Microsoft.PowerShell`
Tailscale	latest	https://tailscale.com/download
Claude Code	latest	`npm i -g @anthropic-ai/claude-code`

DNS: cambiar a Cloudflare 1.1.1.1 (mismo problema con Fibertel — guardar en bookmarks `F:\gaston-workspace\polymarket-weather-mcp\scripts\set_dns_cloudflare.ps1` que ya hace el cambio).

4.2 Copiar workspace desde casa

Opción más limpia: con git/rclone via Tailscale.

```
# Desde PC oficina (con Tailscale activo)
robocopy \\pc-casa.tail89f35a.ts.net\F$\gaston-workspace F:\gaston-workspace `
  /MIR /XD .git node_modules __pycache__ .venv data/logs `
  /R:3 /W:5
```

■■ Antes de robocopy, en PC casa habilitar share de F:\: `New-SmbShare -Name "F$" -Path "F:\" -FullAccess "<cuenta>"` . O usar rsync via Tailscale SSH.

Lo que sí hay que copiar:

- `F:\gaston-workspace\` completo (workspace + ai-stack + polymarket-weather-mcp + memory)
- `C:\tools\` (los 7 MCPs custom)
- `C:\Users\<vos>\.claude\` (skills + settings + token-savior DB)
- `F:\Polymarket\polymarket_completo.md`

Lo que NO copiar:

- `node_modules` , `.venv` , `.git` (regenerar en oficina)
- Logs viejos (`*.log` , `data/logs/`)
- Cache Python (`__pycache__`)

4.3 Levantar el stack en oficina

```
# 1) Stack Docker
cd F:\gaston-workspace\ai-stack
docker compose up -d

# 2) MCPO Windows host
powershell -ExecutionPolicy Bypass -File F:\gaston-workspace\ai-stack\start-mcpo-windows.ps1

# 3) Verificar
docker ps
curl -H "Authorization: Bearer mcpo-gaston-secret" http://localhost:8771/polymarket-weather/openap
```

4.4 Re-instalar dependencias Python globales

Los MCPs custom usan paquetes Python en user-site. Re-instalar:

```
# Lo que tiene PC casa (los más importantes)
C:\Python314\python.exe -m pip install --user `
  mcp[cli] py-clob-client web3 eth-account requests httpx websockets `
  python-dotenv pydantic cachetools aiosqlite numpy scipy pandas `
  python-metar tenacity pywin32 lightrag-hku
```

4.5 Configuración a verificar/copiar

Revisar y completar manualmente:

- `.env` files de cada proyecto (claves API que NO se versionan)
- `webui.db` de OpenWebUI: si copiaste el volumen, ya tenés la config — sino, agregar tool servers manualmente con el script `scripts\add_owui_tool_server.py`
- DB de token-savior: copiar de casa para no perder memorias.

4.6 Configuración de la PC casa para servir 24/7 (CRÍTICO)

Para que la PC oficina pueda conectarse en cualquier momento, la PC casa tiene que cumplir estas reglas. Aplica al **Camino A** (recomendado): la PC casa siempre prendida, lista para responder.

4.6.1 Energía — nunca dormirse, nunca hibernar

Ejecutar como **Administrador** en PowerShell:

```
# Sleep / hibernate / monitor sleep — TODO desactivado
powercfg -change -standby-timeout-ac 0
powercfg -change -standby-timeout-dc 0
powercfg -change -hibernate-timeout-ac 0
powercfg -change -hibernate-timeout-dc 0
powercfg -change -monitor-timeout-ac 0      # 0 = nunca apagar pantalla en AC
powercfg -change -monitor-timeout-dc 15     # 15 min en batería (si es laptop)
powercfg -h off                              # deshabilitar hibernación completa

# Plan de energía: Ultimate / Alto rendimiento
powercfg -setactive 8c5e7fda-e8bf-4a96-9a85-a6e23a8c635c  # High Performance

# Verificar
powercfg /list
powercfg /query SCHEME_CURRENT
```

Equivalente desde GUI: **Configuración** → **Sistema** → **Inicio/apagado y suspensión** → "Suspender: Nunca" y "Apagar pantalla: Nunca" (en AC).

■■ Atención: en algunos motherboards modernos hay un setting BIOS extra "ErP Ready" o "Modern Standby" que puede dormir igual aunque Windows diga lo contrario. Si la PC se duerme igual, entrar al BIOS y desactivar **ErP** y **S3 sleep**.

4.6.2 Auto-arranque post-reboot/cortes de luz

Lo que ya arranca solo después de reboot:

- ■ Containers Docker con `restart: unless-stopped` (todos los del compose).
- ■ Tailscale (servicio Windows, arranca al boot).
- ■ Docker Desktop NO arranca solo por default.
- ■ MCPO Windows host (proceso) NO arranca solo.

Hay que configurar 2 cosas:

(a) Docker Desktop al boot Abrir Docker Desktop → Settings → General → marcar **"Start Docker Desktop when you sign in to your computer"** y **"Open Docker Dashboard at startup"**.

(b) MCPO Windows host como tarea programada al boot

```
# Como Administrador
schtasks /Create `
  /TN "MCPO-Windows-Host" `
  /TR "powershell -ExecutionPolicy Bypass -WindowStyle Hidden -File F:\gaston-workspace\ai-stack\s
  /SC ONSTART `
  /RU SYSTEM `
  /F

# Verificar
schtasks /Query /TN "MCPO-Windows-Host"
```

El script `start-mcpo-windows.ps1` ya tiene chequeo de "si ya está corriendo no lo lanzo de nuevo", entonces es idempotente.

(c) Polymarket-weather SSE backup (opcional) Si en algún momento querés correr el MCP también como SSE server independiente (alternativa a stdio):

```
schtasks /Create `
  /TN "polymarket-weather-mcp-sse" `
  /TR "F:\gaston-workspace\polymarket-weather-mcp\scripts\start_sse_server.bat" `
  /SC ONSTART /RU SYSTEM /F
```

NO necesario en el setup actual — MCPO maneja el lifecycle vía stdio.

4.6.3 Auto-login Windows (sin password al boot)

Si Windows te pide password al arrancar, los servicios programados como `RU SYSTEM` corren OK pero los que dependen del **user session** (ej. Docker Desktop sin "Run when sign in") quedan colgados. Auto-login arregla eso.

■■ **Riesgo de seguridad**: cualquiera con acceso físico a la PC entra sin password. SOLO recomendable si:

- La PC está en un lugar físicamente seguro (tu casa).
- Bitlocker está activo o el disco está cifrado.
- Tenés Tailscale como única vía de acceso remoto (no RDP público).

Forma segura — Sysinternals Autologon:

1. Bajá Autologon.exe de https://learn.microsoft.com/en-us/sysinternals/downloads/autologon (oficial de Microsoft).
2. Ejecutá como Administrador. Ingresá usuario, dominio (`.` para local), y password.
3. Click **Enable**. La password queda cifrada en LSA Secrets (no en plaintext en registry).
4. Para desactivar: corré el .exe de nuevo y click **Disable**.

Alternativa GUI: Win+R → `netplwiz` → desmarcar "Los usuarios deben escribir su nombre y contraseña". Te pide la pass del user. Funciona pero deja la pass en registry plaintext (menos seguro).

4.6.4 Escritorio remoto (RDP) vía Tailscale

Para acceder a la GUI de la PC casa desde la oficina, RDP encapsulado en Tailscale es lo más limpio.

Activar RDP en PC casa (requiere Windows 10/11 Pro — el tuyo es Pro ✓):

```
# Como Administrador
# Habilitar RDP
Set-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server" -Name "fDenyTSConn

# Permitir en Firewall
Enable-NetFirewallRule -DisplayGroup "Remote Desktop"

# Verificar que está escuchando
Get-Service TermService          # debe estar Running
netstat -an | findstr ":3389"    # debe escuchar 0.0.0.0:3389
```

Equivalente GUI: **Configuración** → **Sistema** → **Escritorio remoto** → toggle ON.

Conectarse desde PC oficina (Tailscale activo en ambas):

```
# Opción 1: hostname Tailscale
mstsc /v:pc-casa.tail89f35a.ts.net

# Opción 2: IP Tailscale
mstsc /v:100.91.248.66
```

Login: usuario y password del user de la PC casa.

■■ **Importante**: con auto-login activado, RDP va a "patear" tu sesión local — la pantalla local se bloquea cuando entrás por RDP. Es comportamiento normal de Windows Pro. Para tener sesiones simultáneas necesitás Windows Server o un parche tipo `RDP Wrapper` (no recomendado, hackish).

Tip: si la PC está minimizada a la barra cuando entrás por RDP, Windows puede cambiar la resolución. Para preservar resolución exacta: en `mstsc` → Mostrar opciones → Pantalla → marcar la resolución específica.

4.6.5 Wake-on-LAN (encender PC remotamente — opcional)

Si la PC casa se apaga (por corte de luz que duró más que la UPS, o si la apagás explícitamente), podés encenderla remotamente con WoL **siempre que esté enchufada y el motherboard soporte**.

Pasos en BIOS de PC casa:

- Habilitar **Wake on LAN** / **PCI-E Power On** / **ErP** off.

Pasos en Windows:

```
# Identificar la NIC y habilitar magic packet
Get-NetAdapter | Format-List Name, InterfaceDescription, MacAddress
# Anota el MAC. Luego en Device Manager → propiedades de la NIC → Power Management → "Allow this
```

Desde PC oficina (Tailscale activo):

- WoL **NO atraviesa Tailscale** directamente (es L2, Tailscale es L3). Hay 2 caminos:
- (a) Tener un dispositivo siempre prendido en la red de casa (router con WoL, NAS, Raspberry Pi) que mande el magic packet a pedido. Si hay otro device Tailscale en la misma LAN que la PC casa, ese hace de proxy WoL.
- (b) Cloudflare Tunnel + worker que mande WoL — más complejo.

Recomendación práctica: **dejá la PC siempre prendida** (sección 4.6.1). WoL es solución a un problema que no deberías tener.

4.6.6 Antivirus / Defender — exclusiones

Windows Defender puede bloquear/lentificar el MCPO Windows host (es un proceso Python que escucha en TCP). Excluir paths:

```
# Como Administrador
Add-MpPreference -ExclusionPath "F:\gaston-workspace"
Add-MpPreference -ExclusionPath "C:\tools"
Add-MpPreference -ExclusionPath "C:\Python314"
Add-MpPreference -ExclusionProcess "C:\Python314\python.exe"
Add-MpPreference -ExclusionProcess "mcpo.exe"
```

4.6.7 UPS (recomendado, si tenés)

Si tenés una UPS con USB, instalá su software (APC PowerChute, CyberPower PowerPanel) y configurá:

- Apagar containers limpio (`docker compose stop`) cuando la batería caiga al 30%.
- Si la batería se restaura, reiniciar todo (con auto-login + scheduled tasks ya queda).

Sin UPS, simplemente confiar en `restart: unless-stopped` de los containers + scheduled tasks.

4.7 Credenciales y secretos — dónde están y cómo no perderlas

■■ **REGLA IMPORTANTE**: este documento NO contiene passwords reales. Las claves están dispersas en archivos del sistema. Esta sección es un **mapa** de dónde están, para que puedas migrarlas a la oficina sin perder nada.

4.7.1 Inventario de secretos del stack

Servicio / clave	Dónde se guarda	Cómo accederla
OpenWebUI admin user	`webui.db` tabla `auth`	email `gastonrpersoglia@gmail.com` (la pass es bcrypt, no se puede recuperar — si se olvida, reset via `webui.db`)
OpenWebUI WEBUI_SECRET_KEY	`docker-compose.yml` línea 28	grep `WEBUI_SECRET_KEY` en compose
MCPO Bearer (Linux + Windows)	`start-mcpo-windows.ps1` línea 12 + `docker-compose.yml`	`mcpo-gaston-secret` (lo veo en logs y configs)
OpenRouter API key	`docker-compose.yml` env OPENWEBUI	`sk-or-v1-...` (ver compose)
Cerebras API key	OpenWebUI config	en `tool_server.connections` de webui.db
Groq API key	idem	idem
Github Models token	idem	idem
Cloudflare account ID + token	`cloudflare-mcp .env`	`C:\tools\cloudflare-mcp\.env`
Hostinger API token	`hostinger-mcp .env`	`C:\tools\hostinger-mcp\.env`
Tailscale auth (login persistente)	`%ProgramData%\Tailscale\`	implícito tras login GUI
Polymarket private key	`.env` del MCP	`F:\gaston-workspace\polymarket-weather-mcp\.env` (NO existe aún — ponerla cuando vayas a tradear live)
Polygonscan API key	idem	gratis en polygonscan.io/myapikey
FRED API key	idem	gratis en fred.stlouisfed.org/docs/api/api_key.html
BLS API key	idem	gratis en bls.gov/developers
BingX API + Secret	`mcpo-config.json` env de `ccxt`	(testnet keys, ver mcpo-config.json)
Supabase access token	`mcpo-config.json` args de `supabase`	(proyecto CannTrace `sqdqvhjlmdweuuncwlfb`)

Servicio / clave	Dónde se guarda	Cómo accederla
Telegram TG_APP_ID + TG_API_HASH	`mcpo-windows-config.json` env de `telegram`	ver config

4.7.2 Cómo migrar todo a la oficina sin que falte nada

Forma 1 — copiar archivos directos (rápida):

- Workspace completo: `F:\gaston-workspace\` (incluye TODOS los `.env` de los proyectos)
- Tools custom: `C:\tools\*\.env`
- OpenWebUI volume: `docker volume inspect openwebui_data` para ver path real, copiarlo entero (incluye `webui.db` con todos los providers ya configurados)
- LightRAG storage: `F:\gaston-workspace\gaston-trading\tools\LightRAG\`
- Token-savior DB: ver siguiente

Forma 2 — password manager (más limpio, recomendable a largo plazo):

- Instalá **Bitwarden** (gratis) o **KeePassXC** (offline, gratis) en ambas PCs.
- Migrá todas las API keys ahí. En cada `.env` poné un comentario que apunte a la entry de Bitwarden.
- Ventaja: si rotás una key, lo hacés una sola vez.

4.7.3 Token-savior DB (memoria persistente)

La DB del token-savior tiene TODO lo que aprendí en sesiones contigo (incluyendo toda la lista de la sección 2 de este doc).

```
# Path típico (verificá ejecutando token-savior memory_db.get_db_path)
$tsDb = "C:\Users\<vos>\AppData\Local\token-savior\memory.db"
# O lo que sea — el venv lo tiene en algún lado
Get-ChildItem -Recurse "C:\tools\token-savior-venv" -Filter "*.db"
```

Para migrar: copiá ese `.db` al mismo path en la PC oficina. Las memorias se inyectan automáticamente al iniciar Claude Code (vía hook `SessionStart`).

4.7.4 Passwords del usuario Windows

Si vas a hacer auto-login (sección 4.6.3), necesitás conocer la password del user de Windows. Si la olvidaste:

- Usá Microsoft Account recovery (si tenés cuenta MS).
- O reseteá vía installer disk (modo "Repair").
- NUNCA pongas la password en plain text en este documento ni en ningún `.env` versionado.

4.7.5 Backup de credentials antes de mudarte

Antes de empezar la migración, exportá:

```
# Volumen Docker OpenWebUI (incluye toda la config + chat history)
docker run --rm -v openwebui_data:/data -v ${PWD}:/backup alpine `
  tar czf /backup/openwebui_data_backup.tgz -C /data .
```

```
# Workspace completo
robocopy F:\gaston-workspace F:\backup\gaston-workspace /MIR /R:3 /W:5

# Tools
robocopy C:\tools F:\backup\tools /MIR /R:3 /W:5

# .claude (skills + plans + settings)
robocopy "$env:USERPROFILE\.claude" F:\backup\.claude /MIR /R:3 /W:5

# Token-savior DB
robocopy C:\tools\token-savior-venv F:\backup\token-savior-venv /MIR /R:3 /W:5
```

Después podés llevar ese `F:\backup\` en un pendrive cifrado o subirlo a un cloud privado (Hostinger Object Storage, R2 Cloudflare, etc.).

5. Comandos diarios útiles

Levantar / bajar stack

```
# Levantar todo (después de boot)
cd F:\gaston-workspace\ai-stack
docker compose up -d
powershell -ExecutionPolicy Bypass -File start-mcpo-windows.ps1

# Bajar todo
cd F:\gaston-workspace\ai-stack
docker compose down
# Matar MCPO Windows host
$mcpo = Get-NetTCPConnection -LocalPort 8771 -State Listen -ErrorAction SilentlyContinue
if ($mcpo) { Stop-Process -Id $mcpo.OwningProcess -Force }
```

Verificar salud de cada servicio

```
docker ps                                                    # containers up
curl http://localhost:3000/health                            # OpenWebUI
curl http://localhost:9622/health                            # LightRAG
curl http://localhost:8642/v1/models                         # Hermes
curl -H "Authorization: Bearer mcpo-gaston-secret" `
     http://localhost:8771/polymarket-weather/openapi.json # MCPO Windows
```

Tailscale

```
tailscale status                # ver tailnet
tailscale ip                    # IP propia
tailscale ping pc-casa          # ping a casa
tailscale serve status          # ver qué servicios estás exponiendo
```

Polymarket-weather MCP

```
# Ver tools disponibles
curl -H "Authorization: Bearer mcpo-gaston-secret" `
     http://localhost:8771/polymarket-weather/openapi.json | jq '.paths | keys'

# Test rápido — METAR de NYC
curl -X POST -H "Authorization: Bearer mcpo-gaston-secret" `
     -H "Content-Type: application/json" `
     -d '{"icao": "KLGA"}' `
     http://localhost:8771/polymarket-weather/weather_metar_md
```

6. Troubleshooting conocido (lecciones del 2026-04-27)

Problema	Síntoma	Solución	
Fibertel DNS hijack	`*.polymarket.com` resuelve a `*.com.ar` (208.115.249.235)	Cambiar DNS Windows a 1.1.1.1 — script `scripts\set_dns_cloudflare.ps1` . Requiere admin.	
MCPO container no encuentra `python.exe`	Tools registrados con type=stdio + command Windows fallan	Mover el MCP a `mcpo-windows-config.json` (host MCPO :8771), NO al de Linux/Docker.	
FastMCP HTTP 421 "Invalid Host"	Cliente externo no puede conectar SSE	`FastMCP(transport_security=TransportSecuritySettings(allowed_hosts=["*"]))`	
Tools devuelven JSON crudo en chat Open WebUI	El cuadrito gris muestra dict, model no procesa	(1) `function_calling: native` en config global. (2) Hacer wrappers `_md` que devuelvan str Markdown.	
LightRAG `insert_file` retorna success pero archivo no aparece	Colisión de basename (varios `SKILL.md`)	Usar `POST /documents/upload` con nombre único en vez del MCP.	
Polymarket Data API endpoint 404	`/leaderboard` ya no existe	Usar `/v1/leaderboard?orderBy=pnl	vol&limit=N`.
28-abril-2026: stablecoin upgrade USDC.e → PolyUSD	Approvals on-chain pueden invalidarse	El handler `pmx_approve_post_stablecoin_upgrade` re-aprueba al nuevo address. Esperar 1h downtime.	

Problema	Síntoma	Solución	
30-marzo-2026: weather markets ahora tienen fees	Cálculos de EV asumiendo fee=0 dan trades perdedores	Llamar siempre `pm_get_fee_rate(token_id)` antes de calcular EV.	

7. Polymarket-weather-mcp — referencia rápida

Workflow diario sugerido (paper)

```
1. safety_status_md             → ver bankroll / drawdown / pausa
2. strat_scan_all               → scan unificado (weather + economy + oil + negrisk)
3. strat_rank_opportunities     → top 10 por EV ajustado
4. strat_explain_decision       → narrativa de cada candidate
5. (paper) pmx_paper_place_order
6. db_log_trade + db_log_signal → persistencia
7. db_stats_md                  → review diario
```

Reglas de safety hardcoded (NO desactivar)

1. Bet máximo: 15% bankroll vivo + cap absoluto $20 (lo menor).
2. Daily stop-loss: -25% bankroll/día → pausa 24h.
3. Drawdown pause: -40% acumulado → pausa 2 semanas.
4. Drift detection: rolling 30 trades, win rate < 50% → pausa auto.
5. Auto-skip días sin gaps con EV > 5%.
6. **Modo paper obligatorio mínimo 7 días antes de pasar a live.**
7. **Cada orden real requiere `confirm=True` por llamada (no batch).**

Realidad numérica (ser honesto)

• "Siempre gane" matemáticamente NO existe.

• Mes 1 con $100 + Aggressive: optimista +5-25%, esperado -5-15%, malo -20-40%.

• Drawdown peak típico: -25 a -35% en algún momento.

• 84% de traders pierden en Polymarket. Los del 16% que ganan tienen edge real medible — el MCP da las herramientas para medir si lo tenés vos.

Estrategias implementadas (cuándo usar)

• `strat_gopfan2_scan` : weather <$0.15, sell >$0.45. Edge medio, muchos trades.

• `strat_design_ladder` : 3-5 buckets adyacentes con EV positivo (neobrother style).

• `negrisk_detect_opportunities` : arbitraje multi-outcome con sum YES ≠ $1.

• `strat_hodl_candidates` : HIGH-prob holds (Sharky style). DESACTIVADO por default — ROI bajo con $100.

8. Apéndice — Paths importantes

```
F:\gaston-workspace\
■■■ ai-stack\
■   ■■■ docker-compose.yml            ← stack containers
■   ■■■ mcpo-config.json              ← MCPO Linux (puerto 8770)
■   ■■■ mcpo-windows-config.json      ← MCPO Windows (puerto 8771)
■   ■■■ start-mcpo-windows.ps1        ← launcher MCPO Windows
■   ■■■ mcpo-windows.log              ← logs MCPO Windows
■   ■■■ hermes-shim\                  ← Hermes shim image
■■■ polymarket-weather-mcp\           ← MCP nuevo
■   ■■■ server.py                     ← FastMCP entry, 100 tools
■   ■■■ core\
■   ■   ■■■ config.py                 ← env vars
■   ■   ■■■ format.py                 ← formatters Markdown (_md)
■   ■   ■■■ weather\                  ← NOAA + Open-Meteo + ICAO + consensus
■   ■   ■■■ polymarket\               ← Gamma + CLOB + Data + NegRisk
■   ■   ■■■ chain\                    ← Polygonscan + known_traders
■   ■   ■■■ economy\                  ← FRED + BLS + consensus
■   ■   ■■■ strategy\                 ← EV + Kelly + signal + safety + bayesian
■   ■   ■■■ storage\                  ← SQLite schema + db ops
■   ■   ■■■ util\                     ← cache + ratelimit + http + logging
■   ■■■ scripts\
■   ■   ■■■ set_dns_cloudflare.ps1    ← fix DNS hijack
■   ■   ■■■ rollback_dns.ps1          ← revert DNS
■   ■   ■■■ start_sse_server.bat      ← start SSE (alternativo)
■   ■   ■■■ add_owui_tool_server.py   ← agregar tool a OpenWebUI DB
■   ■   ■■■ fix_function_calling.py   ← native mode
■   ■■■ data\
■       ■■■ logs\                     ← runtime logs
■       ■■■ trades.db                 ← SQLite de trades (cuando opere)
■■■ memory\                           ← exposed via Obsidian
■■■ INSTRUCTIVO_OFICINA.md            ← ESTE ARCHIVO

F:\Polymarket\
■■■ polymarket_completo.md            ← doc Polymarket completa (679+ líneas con apéndice)

C:\tools\                             ← MCPs custom Python
■■■ cloudflare-mcp\
■■■ hermes-admin-mcp\
■■■ mcpo-admin-mcp\
■■■ openwebui-admin-mcp\
■■■ hostinger-mcp\
■■■ tailscale-mcp\
■■■ telegram-mcp\

C:\Users\Gaston\.claude\
■■■ skills\
■   ■■■ polymarket-weather\SKILL.md   ← skill nueva
■   ■■■ ...                           ← otras skills
■■■ plans\
■   ■■■ 2-3-4-wiggly-galaxy.md        ← MEGAPLAN polymarket-weather
■■■ settings.json                     ← config Claude Code
```

9. Próximos pasos sugeridos

Prioridad	Tarea	Cuándo
Alta	Instalar Tailscale en oficina + verificar conectividad	Ahora

Prioridad	Tarea	Cuándo
Alta	`tailscale serve` los 3-4 servicios principales	Después de Tailscale
Media	Probar OpenWebUI + polymarket-weather desde oficina	Después de serve
Media	Configurar `POLYGONSCAN_API_KEY` (gratis) y `FRED_API_KEY`	Cuando uses tools chain/econ
Media	Fase 3 polymarket-weather: paper-trading + cron resolución diaria	Después de testear bien
Baja	Migrar Hermes a la 5090 cuando esté en oficina (vLLM oficina)	Cuando quieras inferencia local pesada
Baja	Replicar workspace completo en oficina (Camino B)	Solo si querés independencia total

> Cualquier cosa que olvidé o que tengas duda, abrí Claude Code y preguntame: tengo TODO esto en memoria persistente (token-savior). Las memorias se inyectan automáticamente al iniciar sesión.