**Claude Flow v2.0.0-alpha.88 correct command syntax  
  
ONESHOT:  
🎯 Der komplette Befehl:**

npx claude-flow@alpha hive-mind spawn "Bearbeite alle offenen Issues im Ordner $(pwd)/issues. Stelle sicher dass WIRKLICH ALLE AUFGABEN, auch Teilaufgaben in den vorhandenen Issues durchgeführt, abgeschlossen und getestet sind, bevor du das Issue auf closed setzt. Bei Fehlern oder unlösbaren Aufgaben, Meldung im Terminal und Issue auf offen lassen. Beachte description, additional und comments. Abarbeitung step by step: Issue\_1.json, Issue\_2.json usw. Nach erfolgreicher Bearbeitung state=closed setzen." --claude --verbose

**🔄 Alternative ohne npx (wenn lokal installiert):**

claude-flow hive-mind spawn "Bearbeite alle offenen Issues im Ordner $(pwd)/issues. Stelle sicher dass WIRKLICH ALLE AUFGABEN, auch Teilaufgaben in den vorhandenen Issues durchgeführt, abgeschlossen und getestet sind, bevor du das Issue auf closed setzt. Bei Fehlern oder unlösbaren Aufgaben, Meldung im Terminal und Issue auf offen lassen. Beachte description, additional und comments. Abarbeitung step by step: Issue\_1.json, Issue\_2.json usw. Nach erfolgreicher Bearbeitung state=closed setzen." --claude --verbose

**📝 Was macht was:**

* hive-mind spawn = Hive Mind Modus mit automatischem Verhalten
* "Task..." = Dein vollständiger Task als String
* --claude = Aktiviert Claude Code CLI Integration
* --verbose = Zeigt alle Schwarmaktivitäten im Terminal

**Die Config wird automatisch aus dem .claude/ Ordner geladen. Falls deine Config noch in .claude-flow/config.json liegt:**

# Vorher einmalig ausführen:

cp .claude-flow/config.json .claude/config.json  
  
Der Befehl cp .claude-flow/config.json .claude/config.json bewirkt folgendes:

**📋 Was macht der Befehl:**

bash

cp .claude-flow/config.json .claude/config.json

**Kopiert** deine Konfigurationsdatei vom **alten** Speicherort zum **neuen** Speicherort:

* **VON:** .claude-flow/config.json (alter Pfad)
* **NACH:** .claude/config.json (neuer Pfad)

**🔄 Warum ist das nötig?**

Claude Flow v2.0.0-alpha.88 hat den Standard-Konfigurationspfad geändert:

* **Alte Versionen:** Suchten nach .claude-flow/config.json
* **Neue Version (v2.0.0):** Sucht nach .claude/config.json

**Ich speichere die config.json jetzt in beiden verzeichnissen (Lauch Hive-Mind Button löst es aus) -> Aktuelle Agent Konfiguration aus „Agent Config“**

Das ist der One-Shot Befehl, den du direkt im WSL ausführen kannst!

**ENDE ONESHOT**

**Ordner ignorieren (claude & claude-flow):**

Claude Flow/Eingabe-Objektive

Der Grund, warum in deinem Hive-Mind-Log plötzlich eine Fremdanweisung („Bearbeite das erste offene Issue…“) auftaucht, ist sehr wahrscheinlich, dass **Claude Flow/Claude Code standardmäßig alle Dateien im Arbeitsverzeichnis indiziert** – auch Dokumentationsdateien. Wenn sich z. B. in 01\_Documentation ein Markdown- oder JSON-File mit einer solchen Anweisung befindet, liest Claude diese beim Start und behandelt sie wie eine Arbeitsbeschreibung (Objective). Es ist also kein fest im Code verbauter Befehl, sondern eine Folge der Indexierung aller Projektdateien.

Um zu verhindern, dass *Claude Code* (und damit auch Claude Flow) bestimmte Ordner scannt, musst du diese explizit ausschließen. Eine .gitignore-Datei wirkt hier nicht – Claude Code bietet stattdessen ein eigenes Ignorier-Mechanismus über die Konfiguration. Laut Anthropic-Dokumentation legst du in einer projekt-spezifischen Datei .claude/settings.json unter dem Schlüssel permissions.deny Muster fest, die nicht gelesen werden sollen. Zum Beispiel:

{  
 "permissions": {  
 "deny": [  
 "Read(./01\_Documentation/\*\*)"  
 ]  
 }  
}

Dieser Eintrag bewirkt, dass **alle Dateien im Ordner 01\_Documentation vollständig ausgeblendet** werden – Claude Code darf sie nicht lesen oder indexieren. Entsprechende Beispiele findet man in den offiziellen Anthropic-Einstellungen: Dort heißt es etwa, man solle mit permissions.deny sensible Dateien und Verzeichnisse wie .env oder build/ ausschließen[[1]](https://docs.anthropic.com/en/docs/claude-code/settings#:~:text=%7B%20,). Einmal so konfiguriert, erscheint in den Logs folgender Hinweis: *“Files matching these patterns will be completely invisible to Claude Code”*[[2]](https://docs.anthropic.com/en/docs/claude-code/settings#:~:text=This%20replaces%20the%20deprecated%20,accidental%20exposure%20of%20sensitive%20data).

Praktisch gehst du so vor: Erstelle im Projektverzeichnis (dort, wo du claude-flow aufrufst) den Ordner .claude/ und darin eine Datei settings.json. Füge unter "permissions":{"deny":[...]} dein gewünschtes Muster hinzu – etwa "Read(./01\_Documentation/\*\*)". Speichere ab und starte Claude Flow neu. Ab sofort ignoriert Claude Code den Ordner und liest dessen Inhalte **nicht mehr ein**. Damit verschwindet auch die ungewollte Anweisung aus den Objectives.

**Quellen:** Anthropic-Dokumentation zu Claude Code: Beispiel für permissions.deny in .claude/settings.json[[1]](https://docs.anthropic.com/en/docs/claude-code/settings" \l ":~:text=%7B%20,) und Hinweis darauf, dass mit diesem Mechanismus Dateien unsichtbar gemacht werden (Ersatz für ältere ignorePatterns)[[2]](https://docs.anthropic.com/en/docs/claude-code/settings#:~:text=This%20replaces%20the%20deprecated%20,accidental%20exposure%20of%20sensitive%20data).

[[1]](https://docs.anthropic.com/en/docs/claude-code/settings" \l ":~:text=%7B%20,) [[2]](https://docs.anthropic.com/en/docs/claude-code/settings#:~:text=This%20replaces%20the%20deprecated%20,accidental%20exposure%20of%20sensitive%20data) Claude Code settings - Anthropic

<https://docs.anthropic.com/en/docs/claude-code/settings>

The error "Unknown command: --mode" occurs because **Claude Flow v2.0.0-alpha.88 uses subcommands instead of flags** to specify operational modes. The --mode flag doesn't exist in this version - you need to use hive-mind as a subcommand directly.

**Your command needs this correction**

Instead of your current command:

claude-flow --mode hive-mind --behavior auto --config --verbose --task

Use this correct syntax:

npx claude-flow@alpha hive-mind spawn "your task description" --claude --verbose

The fundamental difference is that hive-mind is a subcommand that comes directly after claude-flow@alpha, not a value passed to a --mode flag. This architectural change in v2.0.0-alpha.88 means all operational modes (hive-mind, swarm, sparc) are now top-level subcommands rather than flag parameters.

**Complete hive-mind command options**

The hive-mind subcommand offers several operations beyond just spawning tasks. Here are the primary commands you'll use:

For **initializing a hive-mind topology** with specific agent counts:

npx claude-flow@alpha hive-mind init --topology mesh --agents 8

For **spawning a hive-mind task** with your desired parameters:

npx claude-flow@alpha hive-mind spawn "build REST API with authentication" --agents 8 --claude --verbose

For **interactive configuration** when you're unsure of the exact parameters:

npx claude-flow@alpha hive-mind wizard

For **monitoring active hive operations**:

npx claude-flow@alpha hive-mind status

**Mapping your original parameters**

Your original command attempted to use several parameters that need different handling in v2.0.0-alpha.88:

**Auto behavior** is now handled through the --strategy auto flag when using swarm mode, or is automatic in hive-mind mode. The hive-mind automatically coordinates agent behaviors based on the task complexity.

**Config file usage** requires a separate initialization step:

npx claude-flow@alpha init --force

npx claude-flow@alpha config show

**Verbose output** remains available as --verbose and can be appended to most commands.

**Task parameter** is now provided as a quoted string argument directly after the spawn command, not as a separate flag.

**Alternative: simpler swarm mode**

For many use cases, the **swarm mode provides similar multi-agent capabilities** with simpler syntax:

npx claude-flow@alpha swarm "your task description" --strategy auto --parallel --verbose

This mode automatically handles agent coordination and is recommended unless you specifically need the persistent multi-agent mesh topology that hive-mind provides. Swarm mode excels at one-off tasks, while hive-mind is designed for ongoing complex projects requiring sustained agent collaboration.

**Complete workflow example**

Here's how to properly execute a task using hive-mind mode with all your desired parameters:

# Step 1: Initialize the project configuration

npx claude-flow@alpha init --force

# Step 2: Start hive-mind with your task

npx claude-flow@alpha hive-mind spawn "your specific task here" --claude --verbose

# Step 3: Monitor progress (optional, in another terminal)

npx claude-flow@alpha swarm status --real-time

This approach gives you the auto-behavior coordination, verbose output, and proper configuration management you were seeking with your original command.

**Getting help with specific commands**

Claude Flow v2.0.0-alpha.88 includes comprehensive help documentation accessible through:

npx claude-flow@alpha --help

npx claude-flow@alpha hive-mind --help

These commands will show all available flags and options specific to each subcommand, ensuring you can fine-tune the behavior for your specific needs.