# Set up DeepSeek's super-agent

A walkthrough for installing and configuring dsh-superagent, DeepSeek's open-sourced agent that works one GitHub repo (and optionally your team chat) from your own notifications.

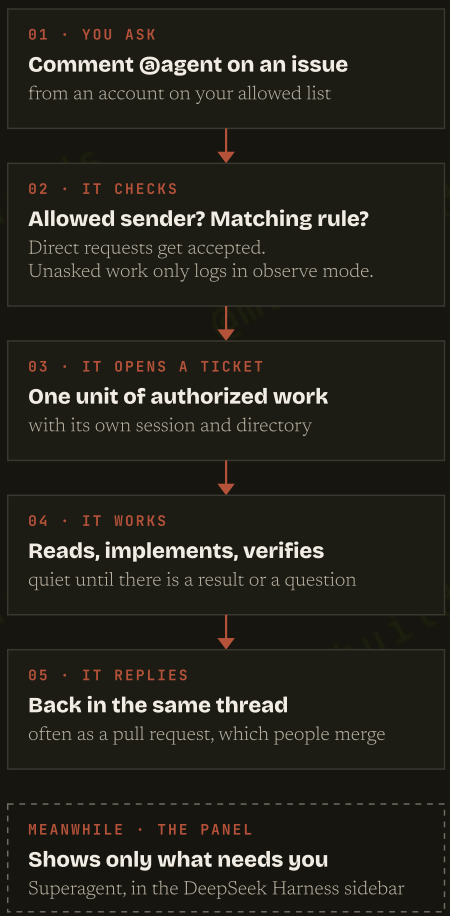
> READ THIS FIRST
>
> DeepSeek's own README calls this a "research prototype for reference only, not an official DeepSeek product," with much of the engineering done by coding agents, and it may still contain bugs. This guide is written from DeepSeek's own docs (read 3 Nov 2026), not from a test install, and every command is copied from them.

---

OVERVIEW

## How it works

You mention the agent on a GitHub issue. It checks who asked against your rules, opens a Ticket, does the work in its own session, and answers in the same thread, usually with a pull request.



01 / WHAT YOU'LL HAVE

## What this gets you

You get an agent that watches one GitHub repository (and, if enabled, Feishu/Lark chats). An allowed comment, `@<agent login> <request>`, on an issue opens a `Ticket` (the docs' term for one unit of authorized work): it works in its own session and directory, then replies in the same thread, often with a pull request. A web `panel` inside DeepSeek Harness shows what needs your attention. Setup has real steps: accounts, a machine, an interview, a test, so expect one sitting.

---

02 / BEFORE YOU START

## Prerequisites

Check these first; the setup agent probes most automatically, but it's faster to know going in.

| NEEDED | VERSION / DETAIL |
|---|---|
| Node.js | `22.19+` or `24+` (`package.json` `engines.node`: `^22.19.0 \|\| ≥24.0.0`) |
| pnpm | Required; the repo's `package.json` pins `packageManager: pnpm@11.7.0` |
| git | `2.32` or later — older git can't read the credential configuration dsh-superagent writes |
| tmux | Required to run the service; no minimum version stated |
| Python | `3.8+` |
| GitHub CLI (`gh`) | `2.48` or later |
| DSH (DeepSeek Harness) | Must satisfy `engines.dsh` in `package.json`: `^0.2.1-alpha.2` |
| Model provider | Configured for DeepSeek Harness already. dsh-superagent uses the profile's default model unless you name another in Eval or Execution settings |
| Repository | Must be on **github.com**. GitHub Enterprise Server is not supported |
| `git-lfs` | Install it if your repository uses Git LFS |
| Feishu (optional) | `lark-cli` (tested with 1.0.90) and a Feishu or Lark app with a bot |

Check what you already have:

```
node -v; pnpm -v; git --version; tmux -V; python3 --version; gh --version | head -1; dsh --version
```

### Accounts you need

- A dedicated GitHub account for the agent, never your own. A GitHub App can't fill the role either, since dsh-superagent reads the account's notifications.
- Optional: a Feishu bot app, if your team wants chat requests too.

> RUN IT ISOLATED
>
> Both presets (worker and router) run `danger-full-access` as you: a worker can run any program by path and read your home directory; wrappers choose credentials, not isolation. The docs recommend: "For isolation, use a dedicated OS account containing only the agent's credentials," before a real repo.

---

03 / INSTALL

## Install DeepSeek Harness and get the repo

The exact install block from the repo's deployment doc; run once, on a machine with no `superagent` profile yet:

```
npm install -g @deepseek-ai/dsh@0.2.1-rc.3
git clone https://github.com/deepseek-ai/dsh-superagent-prototype.git dsh-superagent && cd dsh-superagent
pnpm install && pnpm build
dsh superagent --from-default-profile web --dump-config > /dev/null
dsh plugin --profile superagent add "link:$PWD"
```

> ABOUT THE CLONE LINE
>
> The deployment doc writes this as `git clone <this repository> dsh-superagent`; the public repo is `github.com/deepseek-ai/dsh-superagent-prototype`, cloned into a folder named `dsh-superagent`.

Run the profile-creation command only on a fresh install: it replaces an existing profile and deletes its `cordis.patch.yml`, including your machine configuration.

### Machine configuration

Copy `examples/profile.cordis.patch.yml` to `~/.dsh/profiles/superagent/cordis.patch.yml`, replacing it, and fill in every `<placeholder>`; that example sets:

```
- id: webserver
  config:
    host: 127.0.0.1
    port: 3082
- id: dsh-superagent
  config:
    stateDir: <absolute path, e.g. /home/me/.dsh/state/superagent>
```

`stateDir` must be an absolute path holding the database, lock, command shims, git configuration and Ticket work directories; back it up before upgrades, and enable `dsh-superagent-feishu` in the same file for Feishu.

---

04 / GUIDED SETUP

## Run the guided setup

Hand this to your coding agent, using the README's exact instruction:

```
Read SETUP.md in this repository and set up dsh-superagent with me.
```

It asks one question at a time, each with a one-word recommendation, collects every answer, shows the full plan, and executes only after you confirm, in this order:

### Sources: where requests come from

1. Requests from GitHub only, or also Feishu chats? REC: GITHUB ONLY, UNLESS YOUR TEAM WORKS IN FEISHU
2. Which github.com repository, as `owner/name`?
3. Which GitHub account does it act as? REC: A DEDICATED ACCOUNT, NEVER YOUR OWN, EVEN IF IT DOESN'T EXIST YET
4. If Feishu: which chats, by name? IT LOOKS UP THEIR IDS ONCE THE BOT IS CONFIGURED

### Policy: who may ask for work

1. Which GitHub logins may ask for work? REC: ONLY YOUR OWN LOGIN, TO START NARROW
2. If Feishu: who may ask there? REC: EVERYONE IN THE FIRST CHAT, KEPT IN SYNC WITH ITS MEMBERS

### Execution: what the agent does with a Ticket

1. Which local clone does each Ticket use? REC: A NEW CLONE UNDER ~/SRC/<NAME>

2 Whose identity are commits authored as? REC: THE AGENT ACCOUNT, SO ITS WORK IS VISIBLY ITS OWN

3 Can it auto-merge its own pull requests once approved? REC: NO, PEOPLE MERGE UNTIL ITS PULL REQUESTS HAVE BEEN RELIABLE

The agent never enables auto-merge as its own account, because GitHub attributes the merge to whoever enabled it. A yes on 9 therefore needs another account's GitHub CLI signed in and named in Delivery requirements.

### Defaults it applies without asking

`SETUP.md` settles these without asking:

- Profile name: `superagent`, separate from your own, restarts without closing your sessions
- Port: `3082`
- State directory: `<home>/.dsh/state/superagent`, as an absolute path
- Branch prefix: `agent/`
- Accept mode: `observe`, logs proactive work without doing it
- Hourly check, from your own profile, repairs it when down
- With Feishu: bot reads only mentions, DMs on

Setup also schedules that hourly check from your normal DSH profile; full prompt in `docs/006-deployment.md`, "Hourly check".

---

05 / OPERATOR-ONLY STEPS

## What only you can do

Before executing, it lists steps only you can do: create accounts, sign in CLIs, accept invitations, and never asks for a secret in chat.

### Agent GitHub account

Create a GitHub account for the agent with repo write access, and give it its own CLI:

```
mkdir -p ~/.local/bin
cat > ~/.local/bin/gh-agent <<'EOF'
#!/bin/sh
GH_CONFIG_DIR="$HOME/.config/gh-agent" exec gh "$@"
EOF
chmod +x ~/.local/bin/gh-agent
gh-agent auth login
gh-agent api user --jq .login
```

Sign in and accept the repository invitation as the agent account.

#### Feishu bot (optional)

Create a self-built Feishu app with a bot, publish it, then give the bot its own `lark-cli` configuration:

```
LARKSUITE_CLI_CONFIG_DIR="$HOME/.lark-cli-dsh-superagent" lark-cli config init --ap
p-id <cli_...> --app-secret-stdin
LARKSUITE_CLI_CONFIG_DIR="$HOME/.lark-cli-dsh-superagent" lark-cli config strict-mo
de bot --global
cat > ~/.local/bin/lark-bot <<'EOF'
#!/bin/sh
export LARKSUITE_CLI_CONFIG_DIR="$HOME/.lark-cli-dsh-superagent"
exec lark-cli "$@"
EOF
chmod +x ~/.local/bin/lark-bot
```

Keep this configuration separate from any personal one; to remove only the bot's, delete its directory, never run `lark-cli config remove`.

Auto-merge needs another account's GitHub CLI signed in, never the agent's. Tell your setup agent once done; it verifies each login first.

---

06 / START AND OPEN THE PANEL

## Start it, check it, open the panel

```
scripts/dsh-superagent start
scripts/dsh-superagent status
```

`start` runs `dsh --profile superagent` in a tmux session named `dsh-superagent` (or `dsh—<profile>`), waits up to a minute for the service or a setup report, and refuses a duplicate instance.

| EXIT CODE | MEANING |
|---|---|
| `0` | Runs normally |
| `1` | Not running, running twice, unreachable, or the runtime itself errored — a restart may fix it |
| `2` | Running, but a part such as a Source or Policy sync is erroring — a restart will not fix this |
| `3` | Running, waiting for setup — a restart will not fix this |

Open the URL printed in the log and choose **Superagent** in the DeepSeek Harness Web sidebar for the panel.

---

07 / FIRST TEST

## Try it

From an allowed GitHub account, open an issue or post a comment containing:

```
@<agent login> <request>
```

Within a minute (default polling), the Overview shows a `Ticket`: the agent acknowledges once, works quietly in its own session and directory, delivers the result in the same thread, often as a pull request, and writes again only when ready, stuck, or needing your decision.

> ONE THING TO KNOW
> Editing an existing issue's body to add the mention doesn't count. Post a new comment instead.

---

08 / WORKING PROACTIVELY, SAFELY

## Accept modes: off, observe, auto

A direct, allowed request always works in any mode; what changes is whether it can act on something nobody asked, based on a written `rule` your team enabled.

| MODE | EFFECT |
| --- | --- |
| `off` | Nothing inferred becomes a candidate: not an unaddressed message, not a bot's message, not the follow-up of someone off the allowed list |
| `observe` (default) | Inferred candidates are assessed; a positive assessment is recorded as "would accept" and ignored – nothing is actually accepted |
| `auto` | A positive assessment of an inferred candidate actually accepts it |

> RECOMMENDED PATH
>
> Start in `observe` (the shipped default), review what it would have accepted in the panel's Eval tab, then switch to `auto` once that looks right.

### Auto-merge

Default: no. The agent never enables it as its own account. With a merge account named, the default rules allow it only for the Ticket's own pull request, once it's ready, approved and under 200 changed lines, and withdraw it before any new revision.

---

09 / DAY TO DAY

## Running it day to day

The panel lists what needs you first, then system health, then history. A healthy state takes no space.

### Stopping or pausing work

Pause processing (Execution tab) stops new evaluations, Ticket launches and replies, not a running Executor turn or Sources still recording events. Stopping one Ticket from its detail page is a manual stop, ended until a new request.

### When it's noisy

Make prompts shorter and stricter, or pause it, rather than explaining the noise or using it for volume.

### Upgrading

1. Wait until no Executor is in the middle of a turn.
2. Back up `<stateDir>/orchestrator.sqlite`.

```
git pull && pnpm install && pnpm build
scripts/dsh-superagent restart
```

A restart ends running Assignments, starts replacements with recovery notes, keeps your prompt edits, and applies new defaults elsewhere (review **Default updated**); keep that backup, since a newer version's database won't open in an older one.

### Stopping and restarting the service itself

```
scripts/dsh-superagent stop
scripts/dsh-superagent restart
```

Both wait up to five minutes for Executors to finish, without force-killing or touching other tmux sessions; only an intentional `stop` pauses hourly recovery, cleared by `start`.

---

10 / TROUBLESHOOTING

## Common problems

From the deployment doc's troubleshooting table.

| SYMPTOM | CAUSE AND FIX |
| --- | --- |
| `The agent GitHub CLI is not set.` | Set the agent's CLI on the GitHub card |
| `The agent GitHub CLI cannot read its account` | Sign the wrapper in with `auth login` |
| `@... cannot read ...` or `@... cannot push to ...` | Give the agent account write access to the repository |
| A Ticket fails with `The prepare command failed: ...` | The output names the cause, such as a wrong clone path; fix the command, then retry the Ticket |
| `git does not read the github.com credential configuration dsh-superagent writes` | Install git 2.32 or later |
| `status` exits 2 | A part reports an error, which a restart does not fix; `errors` names each part with its detail |
| `status` exits 3 | Setup waits: `setup` says what's missing, and the overview links to the setting that fixes it |
| Nothing happens after a mention | Check the author is on the Source's allowed list, the Source is on, and processing isn't paused. A mention added by editing an issue's body is never received; mention the agent in a comment instead |

11 / SAFETY CHECKLIST

## Before you point this at a real repository

- [ ] **Dedicated agent GitHub account, never your own.** Not a GitHub App either, it reads account notifications.
- [ ] **Dedicated OS account.** Presets run with `danger-full-access` as you.
- [ ] **Short allowed list to start.** Widen once it's proven.
- [ ] **Observe mode before auto.** Review "would accept" decisions first.
- [ ] **No auto-merge at first.**
- [ ] **Read what it posts under your team's name.** Correct the prompts when wrong.
- [ ] **Back up `<stateDir>/orchestrator.sqlite` before every upgrade.** Newer DB, older version, won't open.

- [ ] **Remember what this is.** A research prototype, not official, may contain bugs.

---

12 / SOURCES

## Everything above comes from

Read 3 Nov 2026, from a local checkout of the dsh-superagent repository and the live DeepSeek Harness page.

**dsh-superagent repo** — README.md, SETUP.md, package.json, cordis.patch.yml, scripts/dsh-superagent, examples/profile.cordis.patch.yml

**Docs** — docs/000-philosophy.md, docs/002-authorization.md, docs/003-execution.md, docs/005-panel.md, docs/006-deployment.md

**DeepSeek Harness** — deepseek.com/en/harness/ — fetched 3 Nov 2026

---

dsh-superagent is a DeepSeek research prototype, MIT licensed, not official; this guide reports only what its docs say and hasn't been run or verified end to end.

@mingbuilds · written from DeepSeek's docs, 3 Nov 2026