

AI Agents for Daily Life Tasks

Victor Dibia

I have been building AI agents for three years.

I have come to believe they should not be just for **work, but for **life**.**

The principles that apply at work translate directly to life tasks.

Non scholae, sed vitae discimus

Not for school, but for life.

My high school English teacher. What we learned in school was meant to be applied outside it, not just to pass exams.

Most agent use is work

- Deep research
- Writing code
- Slide decks

Beyond that, not much.

I think this leaves value on the table.

About me



Victor Dibia · Principal Engineer, Agentic Systems & Data

- Previously Microsoft Core AI and Microsoft Research
- Core developer for [AutoGen](#), 52K stars
- Creator of AutoGen Studio and Agent Optimizer in Foundry Agent Service
- Author, *Designing Multi-Agent Systems* · PhD · IEEE Senior Member

What you'll take away

- Where agents pay off outside work, from six things I did this year
- Why the hour you spend first matters more than the model you pick
- Seven principles and practices you can copy

Use Cases

- Supervising a bathroom remodel
- Building a bathroom cabinet
- Installing wall slats in my office
- An app for my Peloton bike
- Reviewing my taxes and financial position
- Tracking my global to-dos

Supervising a bathroom remodel



Claude and Gemini helped with the design, iterating on images until my wife had what she wanted, then sourcing: from a photo of the tile, two stores that carried the exact pattern, and the aisle.

Typically you would shop across multiple stores to find one.

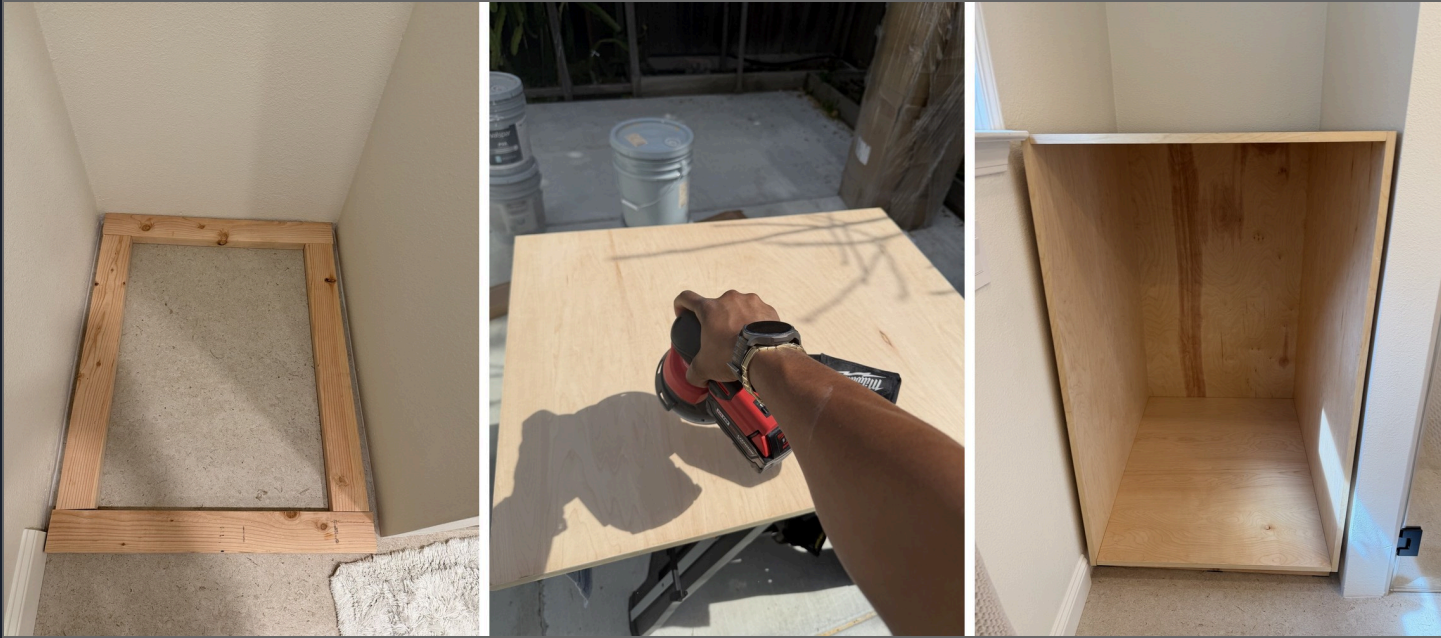
It caught a mistake before it was sealed in the wall

At each stage I sent progress photos and asked what to check before the next one.

Gemini flagged that **insulation was missing from a section of wall. Caught before the walls closed, it was fixed early. Later it is far more costly, or never detected.**

The goal is not to interfere, but to understand the work well enough to ask the right questions, while respecting the expertise of the people doing it.

Building a bathroom cabinet



I had never built one.

Claude helped with navigating build constraints (the cabinet had to be assembled *inside* the bathroom, since the doorway is narrower than the shelf), the joinery, making it moisture resistant, and which tools to buy with no equipment to start.

Claude also makes mistakes

The width of the **drawer slides** was never carefully calculated. That only showed when it was time to fit the drawers in, and I had to resize them.

It is an iterative process. Nothing is perfect the first time.

Installing wall slats



Claude helped with the product and supplier, comparing several, then the measurements, the cuts, and the installation approach.

It also suggested gluing the panels to the wall. Standard, but it did not match my preference for ease of removal, and I had never said so. I asked what the side effects were and it came back with screws.

Fitness tracking

In 2022 I knew roughly how to do this. It was many weekends of work, so I never started.

- A **first version in a weekend**: reads the bike's sensor data at 10Hz, tracks sessions, talks to my watch
 - Then daily use, with fixes dictated from my phone during the workout
 - More consistent in the five weeks since than in the rest of the year
-

Four years of wanting this.

Ride

Workouts

Sessions

Trends

Advanced

Home

Ride

Workouts

Sessions

Trends

Advanced

Watch

8:15 AM

All

★ Favorites

Mine

Endurance

Intervals

HIIT

Climb

Test

Recovery

Import

+ New workout

JUST RIDE

🕒 1:00:00

Just Ride

No structure — ride as you like. Everything is still recorded.

≡ 1 steps

Intensity ●●●●●

TEST

🕒 1:04:00

FTP Test (20 min)

Classic 20-minute test. FTP = 95% of your 20-min average. Pace it: steady, slightly building,...

≡ 12 steps

Intensity ●●●●●

ENDURANCE

🕒 53:00

Sweet Spot 3×8

Three 8-minute blocks at 88–94% FTP with 4-minute recoveries. The best bang-for-buck...

≡ 8 steps

Intensity ●●●●●

INTERVALS

🕒 52:00

VO2max 5×3

Five 3-minute efforts at 106–120% FTP, 3 minutes easy between. Breathing hard by...

●●●●●

HIIT

🕒 24:00

Tabata 2×8

Two sets of 8 × (20 s all-out / 10 s rest), 4 min between sets. Short, brutal, done in 20.

●●●●●

ENDURANCE

🕒 45:00

Endurance 45

45 minutes in zone 2 with three short cadence drills. Conversational pace; this is where the...

●●●●●

● Sensor >

⌘ ● Watch: pairable >

Peloton Bike (Gen 1)10 Hz · 400989 samples192.168.1.116:8080v0.2

Reviewing my taxes and financial position

Three years ago, working on AutoGen, we talked about a North Star: the tasks everybody needs to do and nobody wants to do.

I still use TurboTax and a CPA. I checked the agent's numbers against the CPA's. They were the same.

Your CPA can help because they can see all your documents. An agent can help for the same reason. Everything you would hand to TurboTax, in one folder.

Tracking my global to-dos

One markdown file per thread. Frontmatter names the next concrete action, a size, and what it is blocked on. A build script regenerates one board.

Size describes the `next action`, not the project. A multi-month thread whose next step is a thirty-minute decision is small.

Committed so GitHub renders it, which makes the whole board readable from a phone. Thirty minutes at a swim class is enough to move something.

Principles and Practices

- Start in interview mode
- Dictate rather than type
- Assemble the context
- Move between local and remote
- Supervise the trajectory
- Build strong verification loops
- Record recurring notes

1. Start in interview mode

An hour or two describing the problem and answering questions about it, before anything is built.

- Explore every edge case first
- It becomes a design document in the repo
- I read all of it before anything gets built

Not formal spec-driven development. A long careful conversation about the domain.

2. Dictate rather than type

I speak much faster than I type.

- Makes an hour of interview mode practical at all
- Lets me give feedback to several sessions without typing
- Use a good voice model: bad dictation is garbage in

3. Assemble the context

Private repos or folders, organized so an agent can parse and use them.

Git is optional. The organization is not.

The test: could someone else sit down and work from this folder? Setup compounds. Once an expenses repo existed, everything else went under it.

4. Move between local and remote

- **Cloud agents** keep running after you close the laptop. The clean answer, if your org can set it up.
 - **Remote control** keeps execution on your machine and moves only the control surface to your phone. That is what I use.
-

From a phone I can take pictures, connect to my desktop and share updates. That is exactly what made the remodel supervision work.

5. Supervise the trajectory

Outsource the grunt work, never the **understanding. The buck stops with me.**

- Ask what the side effects are, before you accept the approach
- Read the trajectory that produced the summary

A task whose output you build yourself forces you to understand it. A task whose output is a document lets you skip it.

6. Build strong verification loops

You cannot read everything, so automate the half a machine can judge.

- Deterministic hooks that run in seconds
- Clean rules rather than opinions: structures you can identify in code, constructions that are just not accepted
- They become the evaluation and verification gate

Two of them checked every paragraph of the article this talk came from.

7. Record recurring notes

- A preference becomes an entry in CLAUDE.md or memory
 - A procedure graduates to a **skill**
-

My writing process is one. This talk's article was written under it.

Conclusion

Agents can be applied to everyday life tasks, and the quality of life improvement is significant.

The principles that keep you effective here translate back to [work](#).

Starting in interview mode, dictating instead of typing, assembling the context, moving between local and remote, supervising the trajectory, building verification loops, and recording what recurs are the same things that get the most out of agents for work tasks like software engineering.

If you have found ways to reclaim your own time with this, I would like to hear it.

