# Bunch

By Austen Tucker · 2026

Bunch's catch-up screen for Addie Arcade, with "Help me resume my day" and its main action buttons.

I have DID.

Over the past six years, I have experimented with every kind of system I could find to make my life easier: notes, checklists, shared documents, reminders, and now AI. Bunch is the story of one of those experiments. It is not a product announcement, a million-dollar idea, or a pitch for the cool MCP server I built. It is a build log about what it looks like to use AI to accommodate a person with DID.

## Explore Bunch

Bunch: a context system for continuity across memory gaps

Visit Bunch — Bunch's home on the web.

Install the Bunch demo plugin and skill — try Bunch in Codex with fictional people, shared tasks, notes, and catch-up records served by system.thearcades.me. The demo is read-only and needs no Bunch account.

Once installed, ask: "Show the Demo system. Then show Benny's open tasks and the reminder Fenton left him."

Browse Bunch on GitHub — the code behind this build log.

## Why I built it

## If you have not heard of DID, here is the short version

DID is a response to childhood trauma. That trauma interrupts an important part of a child's development, leaving gaps that can range from inattentiveness to complete memory holes in the way they move through a day. One human way we make sense of that is through alters: distinct personality states who may come forward for different tasks, wants, or needs. We—the alters—share one life and one body, but we do not always share continuity. I can lose track of time, context, conversations, people, and decisions I

participated in. That is enough context for this essay. You do not need a clinical explainer to understand the problem Bunch is trying to solve.

The alters are not optional. I cannot ignore what they need or want any more than I could ignore the need to go to the bathroom. Living responsibly means making room for all of us: giving each alter time, care, and a real say in the life we share. In practice, I am not living one life as a single person. I am living a collection of lives in one body.

Bunch is a private companion built around that need. It gives me somewhere to leave notes, keep the context that might otherwise evaporate, and make a handoff legible to whoever is here next. It is a memory aid. It is a memory prosthetic. It is a memory cybernetic, and I'm standing on that because that technically makes me a cyborg. It makes asking "what happened?" less like falling through a trapdoor.

Bunch used to be called DIDdy: DID plus "ditty," a little song. I eventually realized that was not the right name. Bunch, after a bunch of monkeys, felt friendlier and more like the thing it actually is: a group living one shared life.

The first thing I did with Bunch was ask an honest question: what do I actually need to accommodate myself? Answering it took honest conversations with family and friends about my limitations. Two answers kept returning. I do not reliably retain context about the people in my life, and my memory is shot. Amnesia barriers between alters make that more than an inconvenience. If I am Addie Arcade and I am fronting after Twilight Arcade was out, I need a way to ask: what did he do? Whatever he committed to, I am now responsible for being there for. The commitment does not disappear because the person who made it is not here. I need to be able to find it, understand it, and follow through.

## Build snapshot

- Product surface: A private, MCP-first companion for notes, handoffs, presence, and alter profiles.
- Core problem: Preserve enough context that whoever is here next can understand what happened and follow through.
- Data model: Hosting records responsibility; fronting records presence. The distinction is intentional: DID is complicated, and more than one person may be around at the same time. We invented the data model to increase our sense of system responsibility and preserve continuity.
- Accessible reference UI: An alter lineup with profile pictures that let each alter be visibly themselves. That UI had to be able to render in-harness so that it could access my context. (More on that later.)
- Storage: Private Blob storage for images and a Postgres record of people checking in and out.

## What I did

## I started with the verbs, not the screens

I started by asking a game-design question: what are the verbs I need this tool to support? Not which buttons should it have, or which screens should I build, but what do I need to be able to do? The answers became the MCP. MCP, or Model Context Protocol, lets an AI harness retrieve context and take approved actions through tools instead of making me live inside one more app.

- Arrive — Start a fronting episode when an alter explicitly says they are here.
- Leave — End that fronting episode when its end is explicitly reported.
- Host — Record who is carrying responsibility for the shared life through a period of time.
- Catch up — Retrieve the return window, including unresolved carryover.
- Remember — Create and retrieve private notes, to-dos, and decision records.
- Connect — Suggest and confirm important threads so they can become future context.
- Know us — Create and maintain alter profiles, pictures, and appearance references.
- Trace coverage — Read the recorded history of who was out when.

Those are not dashboard features. They are the vocabulary Bunch gives me for interacting with my own life. Once I knew the verbs, the MCP surface followed from them. “Switch” still exists as a compatibility command, but the newer model splits it apart on purpose: hosting is responsibility, while fronting is presence, and neither necessarily cancels the other.

## I built the first useful slice

This is how I start projects, especially AI projects: get to something useful before trying to get everything right. The first version does not need to be the final system. It needs to let a real person do one real thing that matters. For Bunch, that meant making the core verbs available soon enough to learn whether they could carry the weight I was putting on them.

For Bunch, the first slice was the immediate problem in front of my face. What is the biggest thing I can solve right now, with the tools I already have? I looked through my prototypes and started with the answer that kept coming up: I did not know who was out. I also did not have an easy way for each alter to have a profile picture, so I could recognize them in a lineup. That became the MVP: record presence, build an accessible lineup, and let each alter be visibly themselves.

Nine cartoon animal characters posed together on a sunny beach, kites flying behind them. Four sit in front — a mouse, a cat, a grinning raccoon, and a rabbit in a red vest — and five stand behind, including a cat in a bright patterned sweater, a squirrel in overalls, a blue-haired wolf, and a blond figure in a denim jacket. Most wear glasses. Several flash peace signs.

The lineup. A bunch of monkeys, give or take a few species.

That immediately gave me a technical checklist. Images had to be something I could upload, view, and retrieve—not an afterthought—because I often generate them in the AI tools where I am already working. Private Blob storage handled that. Then I needed to know who had been out recently, so Bunch needed a Postgres record of people checking in and out of the system. The storage choices were not glamorous. They were simply the smallest reliable answers to the things I needed to be true.

Bunch is deliberately simple under the hood. It has a small blob-storage database, a simple set of fields and rules for each alter, and no bloat. Its data model distinguishes between a host—the person carrying responsibility for keeping the shared life running through the week—and the person who is fronting right now. Those are not always the same person. Whoever is fronting may be here for work, play, sleep, or something else entirely. The system needs to know enough to meet them where they are, without pretending it can reduce a whole person to a dashboard.

## What I learned

## Continuity is the value

My bet is not that user experience is going away. Good components and good interfaces still matter. It is that the destination-app model is becoming less important than the ability to reach my data from the harness where I am already working. For an accessibility tool, that difference matters. The value is not a beautiful place for me to go. It is being able to get the right context when I need it.

> *Building my own accommodations is not a new reflex for me. I have had low vision my entire life, and the world has rarely arrived accessible by default. If I want a studio app I can actually use to edit things, I build one. Adobe is not workable enough for me. CapCut has been a horrendous accessibility experience. I have not gone deeper into film editing partly because I still do not know of a good accessible editor. When the tools I want do not make room for me, I have learned to make the room myself.*

## The first useful thing is better than the final imaginary thing

Starting with the data and the verbs gave us something we could use. And one thing it unlocked was discovering how much my system loves making images of itself. We can't see a lot of fine detail, so those images tend to be cartoonish. They've turned out to be really useful for figuring out who's out. Instead of asking, "Oh gosh, which of the ten names is it?" we can pull up the lineup and ask, "Which of these images appeals most to you?" Usually, we can look and go, "Oh, that's me."

Why does it matter to know who's who? Because it lets us track people down later if we need to.

## Sample catch-up — invented details illustrating Bunch’s behavior

This is a sample of what Bunch can say when someone returns. The behavior is real; the details here are invented.

Bunch: Hi, Lucy. It’s been four days since you last switched in. You were in Holland, Michigan, on vacation. Here are the high notes since then:

- Work: It’s a slow week. Not much happening.
- Home: D is away on a solo vacation and will return on Friday.
- To-do: We need to write the next chapter of It Takes a Zoo.

## Current best

By the end of week one, the first slice worked inside an MCP harness. The reference web UI was rough, and that was fine. When I asked who was fronting, Bunch could retrieve who had been out over the past week. That is a huge change for me. I have been using version one for a week now, and it has already changed the way I live my life.

## Limits and open questions

- The reference web UI is still rough; the MCP surface is the current useful interface.
- Hardening and measurement remain ahead of the first working slice.
- Version two is about what happens after the first win: measuring, hardening, and improving a tool once it has earned a place in a real life.