Building a Smarter Home With Home Assistant

May 11, 2026 / Gavin Jackson

home-assistant homelab proxmox lxc automation iot solar smart-home

I have finally started spending some proper hands-on time with Home Assistant.

This is one of those projects I have known about for years, mostly from the edges of the homelab and self-hosting world. It has always had that slightly dangerous appeal: install it to solve one small problem, then gradually discover that half the house has turned into an automation surface.

My goal for this first pass was modest. I wanted to get Home Assistant running in my existing Proxmox homelab, see what it could discover on the network, and work out whether it was worth becoming part of my normal home infrastructure.

The short version: yes, it is.

The longer version is that I was surprised by how quickly it went from "new service in the lab" to "this is now aware of my solar, TVs, air conditioning zones, and phone." That is a pretty compelling first hour.

Installing Home Assistant on Proxmox

My homelab already runs on Proxmox VE, so I installed Home Assistant as an LXC container using my PVE Scripts Local setup, based on the Proxmox VE Helper-Scripts community project.

I have written previously about why I like these scripts in a homelab context. They take common self-hosted applications and turn the initial deployment into something repeatable: create the container, install the application, set sensible defaults, and let me get on with evaluating the actual service.

For a first Home Assistant deployment, an LXC container is a nice fit. It is lightweight, quick to provision, easy to snapshot, and it lives beside the other services I already run in Proxmox.

The official Home Assistant story often leans toward Home Assistant OS, especially if you want the most appliance-like experience with add-ons managed from the UI. That may still be where I end up if Home Assistant becomes critical infrastructure in the house. For now, the LXC approach suits my current goal: learn it, test it, and keep it easy to rebuild.

After the container came up, I opened the web UI, created an account, and stepped through onboarding. Nothing dramatic. No database to stand up, no giant configuration file to write before the first screen appeared, no weird ceremony. It felt refreshingly practical.

The first pleasant surprise: auto-discovery

Once Home Assistant was running, it immediately started finding things.

The most interesting discovery was my Enphase solar setup. Home Assistant has an Enphase Envoy integration that can talk to the Envoy gateway and expose data from the system, including the microinverters behind the panels. That matters because solar monitoring is one of the most obvious places where local visibility is useful.

Vendor apps are fine for checking status, but they are not the same thing as owning the data flow inside your own home automation system. If Home Assistant can see production data, then later it can become part of broader energy decisions: when to run appliances, when to prefer battery, when to charge something, or when to simply make the data visible on a dashboard I control.

It also found my Sony TVs. Home Assistant's Sony Bravia TV integration supports auto-discovery for compatible models, and that is exactly the kind of integration that makes the platform feel useful quickly. Media devices are not the highest-stakes part of the house, but they are a good test of whether Home Assistant can become a single control plane rather than another isolated dashboard.

At this point I had not done much configuration. I had a new container, a new account, and Home Assistant had already started building a picture of the network.

That is a strong first impression.

Adding the air conditioner

The next step was the one I actually cared about day to day: air conditioning.

My ducted air conditioning setup uses AirTouch with a Panasonic unit behind it. Home Assistant supports AirTouch through integrations such as AirTouch 4 and AirTouch 5, depending on the controller generation. Once added, the useful part is that Home Assistant can represent the AC and its zones as climate entities.

This is where Home Assistant started to feel less like "device monitoring" and more like home control.

The AirTouch app already works, but bringing the AC into Home Assistant changes the shape of what is possible. Instead of the air conditioner being trapped inside its vendor app, it can participate in the same interface and future automations as everything else.

That unlocks a few practical ideas:

- Turn the AC off automatically when no one is home.
- Pre-cool or pre-heat the house based on presence, weather, or time of day.
- Avoid running climate control hard when solar production is low or battery reserve is important.
- Create room or zone-based controls that match how we actually use the house.
- Use voice control later without exposing every vendor account separately.

I am not automating all of that yet. This first round was simply about getting the AC visible and controllable. But the important architectural step is done: the air conditioner is now part of the Home Assistant model of the house.

Areas make the model click

One thing I liked more than expected was assigning devices to areas.

It sounds like a small UI housekeeping task, but it matters. A smart home should not primarily feel like a list of vendors and device names. It should feel like a model of the house.

So instead of thinking in terms of "Sony Bravia TV" or "AirTouch climate entity," I can start thinking in terms of:

- living room
- bedrooms
- office
- kitchen
- outdoor areas
- solar and energy

That becomes more important as the system grows. One or two devices can live in a flat list. Twenty devices cannot. Once lights, sensors, switches, media players, climate zones, batteries, and automations are involved, the house model needs structure.

Areas give Home Assistant a useful map. They also make dashboards, permissions, voice control, and automations easier to reason about later.

The phone app made it real

The next step was installing the Home Assistant app on my phone.

Home Assistant has an official Mobile App integration for iOS and Android. Once connected, the phone becomes more than just a remote control. It can receive notifications, expose sensors, and eventually provide presence information for automations.

For now, my immediate use is simple: I can control the air conditioner from Home Assistant on my phone.

That sounds basic, but it is a useful threshold. A homelab service is only valuable in the house if it survives contact with normal life. If controlling the AC requires VPNs, bookmarks, awkward dashboards, or explaining which app to use, it will not stick. If it is just an app on the phone with the devices grouped by area, it has a chance.

The Home Assistant app also hints at where this can go next. Presence detection, actionable notifications, mobile widgets, and location-aware automations all become possible once the phone is part of the system. I will be careful with that, because location data deserves a higher privacy bar than a light switch, but the capability is there.

Who is behind Home Assistant?

Part of what makes Home Assistant interesting is that it is not just another smart home product from a large platform vendor.

Home Assistant was started by Paulus Schoutsen in 2013. Nabu Casa describes the origin nicely: it grew from a script that turned on lights at sunset into one of the major open source home automation frameworks. The project is written in Python, has a very large contributor community, and has become one of the most significant open source projects in the smart home space.

Today, Home Assistant is a project of the Open Home Foundation, which focuses on privacy, choice, and sustainability for smart homes. That framing is important. The smart home industry has a bad habit of turning basic household functions into cloud dependencies, subscription opportunities, and abandoned-device problems.

Home Assistant's bias is different: local control first, cloud when needed, and user ownership of the data. The official Home Assistant site puts that local model front and centre: Home Assistant communicates with devices locally where possible and falls back to cloud only when there is no other option.

That is exactly the posture I want in a home lab and, frankly, in a home.

There is also a commercial company involved: Nabu Casa, founded in 2018 by the founders of Home Assistant and Home Assistant OS. Nabu Casa funds development and offers Home Assistant Cloud, which provides conveniences such as remote access and easier voice assistant integration while still extending a local Home Assistant instance rather than replacing it.

That balance is healthy: open source core, foundation governance, commercial support where it makes sense, and a product direction that does not require turning the whole home into someone else's cloud tenant.

What people use Home Assistant for

The obvious use case is smart lights, but Home Assistant is much broader than that.

The public Home Assistant integrations catalog is enormous. It covers lighting, climate, media players, solar inverters, batteries, EV chargers, weather, presence detection, network devices, security systems, voice assistants, and a long tail of very specific hardware.

The public Home Assistant Analytics dashboard also gives a useful sense of scale. At the time of writing, the opted-in analytics dataset showed roughly 624,000 active installations, with averages around 13 automations, 28 integrations, and 357 states per installation. Those numbers are a good reminder that many Home Assistant users are not just turning on a bulb from a phone. They are building an operating layer for the house.

The most common practical uses seem to cluster around a few themes.

Lighting and switches

This is the classic smart home entry point. Smart bulbs, smart switches, dimmers, scenes, motion sensors, and schedules are all natural Home Assistant territory.

The value is not just "turn light on from phone." The value is combining context:

- turn lights on when motion is detected
- dim lights in the evening
- turn everything off when the house is empty
- use physical buttons without being locked to one vendor
- keep local control even if an internet service is down

This is probably where I will expand next after the energy pieces.

Energy monitoring

Energy is one of the more interesting uses because it connects directly to cost, resilience, and sustainability.

Home Assistant has first-class home energy management features, and my Enphase discovery is a good start. Adding the Tesla Powerwall would make this much more useful. Once solar, battery, grid import/export, and household load are visible in one place, dashboards become genuinely valuable.

The longer-term opportunity is automation: shifting loads, avoiding unnecessary grid draw, and making smarter decisions about climate control.

Climate control

This is now my first real use case.

Air conditioning is expensive, comfort-sensitive, and easy to forget. It is also a perfect example of why a smart home should understand more than one device at a time. The AC should eventually be able to respond to presence, room usage, time of day, weather, solar production, and battery reserve.

That does not mean building a fragile maze of automations. It means starting with a few simple, reliable rules and expanding only when they prove useful.

Media and notifications

TVs, speakers, casting devices, and voice assistants are common Home Assistant integrations. Media control is convenient, but notifications might be more useful long term.

I can imagine alerts such as:

- solar production has dropped unexpectedly
- Powerwall reserve is below a threshold
- AC has been left running in an unused zone
- a homelab service is down
- a door, sensor, or device has changed state unexpectedly

This is where Home Assistant can become a bridge between the house and the homelab.

Presence and routines

Presence detection is where smart homes either become useful or creepy.

Used carefully, presence can make automations feel natural: turn things off when everyone leaves, adjust climate before people arrive, send reminders only when someone is home, or avoid noisy notifications at night.

Used carelessly, it becomes surveillance in your own house. My bias is to keep presence automations simple, visible, and easy to disable.

Next steps

My first Home Assistant setup is now useful, but it is still early. The next phase is where it becomes interesting.

Integrate Alexa

Alexa integration is high on the list because voice control is still one of the most natural interfaces for shared spaces.

Home Assistant supports the Amazon Alexa Smart Home Skill, but the manual path requires internet-accessible HTTPS, an Amazon Developer account, and an AWS Lambda function. The simpler route is Home Assistant Cloud through Nabu Casa, which avoids opening ports and handles the Alexa connection more cleanly.

I need to decide which path fits my risk tolerance. For a family-facing control plane, Home Assistant Cloud may be worth it simply because it reduces the amount of custom internet-facing glue I have to own.

Add the Tesla Powerwall

The Tesla Powerwall integration is probably the most important next energy step.

Right now, Enphase gives me solar visibility. Powerwall would add battery state, grid status, load, reserve, and energy flow (fortunately I'm using Powerwall 2, which is supported). Once that data is in Home Assistant, I can build a much more complete picture of the house as an energy system rather than a collection of unrelated devices.

That is the part I am most excited about. Solar data by itself is interesting. Solar plus battery plus load plus climate control is actionable.

Start adding smart lights

Lighting is the obvious expansion path, but I want to be deliberate.

The smart home graveyard is full of cheap devices that depend on fragile cloud services. I would prefer devices that work locally, support open standards, or at least have a strong Home Assistant integration. Matter, Zigbee, Z-Wave, Shelly, ESPHome, Hue, and local MQTT-based devices are all worth looking at.

> ***Smart lights without replacing the switches***
>
> *One option I am particularly interested in is adding Shelly modules behind some of the existing light switches. A few work colleagues put me on to these, and I can see why they are popular with Home Assistant users.*
>
> *The appeal is that I may not need to replace every visible wall switch or commit to smart bulbs everywhere. A Shelly module can sit behind an existing switch or in the lighting circuit, making the circuit visible to Home Assistant while preserving the normal physical switch experience. That matters in a shared house. If the smart home layer is unavailable, visitors and family should still be able to use a light switch like a light switch.*
>
> *Home Assistant has an official Shelly integration, and the fit looks good for the kind of setup I want. Shelly devices can be auto-discovered, Home Assistant communicates directly with the device, and the cloud connection is not required. Depending on the device generation and configuration, relay devices can also be represented as light entities rather than generic switches.*
>
> *For me, Shelly modules look like a good middle ground:*
>
> - *keep existing wall switches and faceplates*
> - *add Home Assistant control without making the house feel unfamiliar*
> - *avoid relying on a cloud service for day-to-day switching*
> - *gain energy monitoring on selected circuits if I choose the right modules*
> - *build lighting automations gradually instead of replacing everything at once*
>
> *The caution is electrical work. Anything behind a wall switch is mains wiring, so this is not a "just tinker with it on the weekend" job. I will need to check the wiring, neutral availability, box depth, load type, dimming requirements, and local electrical rules, then use a licensed electrician where required.*

My likely pattern will be:

- start with a small number of high-value rooms
- prefer switches where possible so normal wall controls still work
- trial Shelly modules on a couple of simple lighting circuits before standardising
- avoid making lights unusable when Home Assistant is down
- document what I install and why
- keep automations boring until the basics are reliable

Smart lighting should make the house easier to live in, not turn every room into a troubleshooting exercise.

Build dashboards that match the house

The default Home Assistant UI is useful, but I will want dashboards that match how we actually think.

I can see a few useful views:

- a climate view for AC zones
- an energy view for solar, battery, grid, and load
- a living areas view for lights and media
- a homelab view for infrastructure status
- a mobile-friendly quick control view

The trick will be restraint. Dashboards can easily become a cockpit. The best version should answer the obvious household questions quickly: is the AC on, what is using power, are the important things healthy, and what needs attention?

Add a few automations, slowly

I do not want to start by automating everything.

The first automations should be boring and reversible:

- notify if the AC is left on too long
- turn off selected devices when nobody is home
- alert on unusual solar or battery behaviour
- set climate defaults at predictable times
- run simple evening or bedtime lighting scenes once smart lights exist

The rule for home automation should be the same as production infrastructure: if nobody understands why it happened, it is too clever.

Early verdict

Home Assistant has made a good first impression.

The installation was quick, the auto-discovery was better than expected, the AirTouch integration gave me an immediate practical win, and the mobile app makes it feel like something the house can actually use.

The bigger story is not that I can control an air conditioner from another app. The bigger story is that Home Assistant can become the local integration layer for the house: solar, battery, climate, lights, media, phones, notifications, and eventually voice control.

That is why the project is interesting. It is not just a dashboard. It is a way to make the home understandable and programmable without handing the whole thing to a cloud platform.

For a homelab person, that is a very appealing idea.

Downloaded from https://www.gavinj.net/post/home-assistant-first-impressions
Generated November 8, 2026. Copyright Gavin Jackson. All rights reserved.