[GREY_PAPER.md]

Grey Paper

The Null Hypothesis Foundation

v1.0 // February 2026

[TABLE_OF_CONTENTS.md]

[PREAMBLE.md]

Preamble

This is not a white paper. White papers have citations, graphs, and the implicit promise that someone with an MBA reviewed it. None of that happened here.

This is a grey paper. It sits somewhere between a manifesto and a README. It explains what The Null Hypothesis Foundation is, why it exists, and how we think about building software. It is not optimized for investors, because we do not have any. It is not optimized for press, because they have not called.

It is written for anyone who finds themselves building software at 2 AM and wondering whether the industry has always been this strange, or whether it got worse recently. We think it got worse recently. But we also thought that five years ago, so take that with appropriate skepticism.

If you are looking for a pitch deck, you are in the wrong document. If you are looking for a honest accounting of a small software company trying to exist without pretending to be something it is not, keep reading.

[MISSION.md]

Mission

01

Our mission is not to change the world. Our mission is to build things that work, sell them to people who need them, and remain solvent long enough to do it again.

We build security and productivity tools. The security tools help people understand what their networks are actually doing. The productivity tools help people manage information without handing it to a third party. Both categories share a common principle: your data stays on your machine, because we do not want it and cannot afford the liability of having it.

The foundation exists because somebody needed to sign the contracts, own the domains, and be legally responsible when the CI pipeline deploys to production on a Friday. That somebody is us. We are not a startup. We are not a studio. We are a small, independent software company that chose the word "foundation" because it sounded more permanent than we feel.

> *“We don't have enough money to be evil.”*

[VISION.md]

Vision

02

The vision is not scale. The vision is durability.

We want to be the kind of software company that still exists in ten years, not because it got acquired or went public, but because it kept making useful things and people kept paying for them. This is apparently a radical position in an industry where the expected lifecycle of a company is: raise money, grow unsustainably, pivot, get acquired, shut down the product everyone actually used.

We would rather be small and solvent than large and dependent on someone else's patience. Our runway is revenue. Our growth strategy is making things people want. Our exit strategy is not having one.

This is not a principled stand against ambition. We would happily be large if that happened organically. But we will not sacrifice the quality of the work or the autonomy of the team to get there faster. Speed is a vanity metric when your destination is sustainability.

[PHILOSOPHY.md]

Philosophy

How we think about building software, using AI, choosing tools, and staying sane in an industry that rewards the opposite.

03.1 On AI

We are AI-sometimes. The rest is just code we wrote ourselves, like animals.

AI is a tool. It is a good tool for some things and a bad tool for other things. We use it where it helps: generating boilerplate, summarizing transcripts, identifying patterns in data that would take a human too long to spot. We do not use it where it does not help: making product decisions, writing copy that needs to sound like a human wrote it, or anywhere the cost of being wrong is higher than the cost of being slow.

We will never call ourselves AI-native, AI-first, or AI-powered. We are software-powered. Some of that software uses machine learning models. Some of it uses if-else statements. The user does not care about the distinction, and neither should we.

The current AI discourse has a religiosity to it that makes us uncomfortable. Tools do not require belief. You do not need to have faith in a hammer. You pick it up, you use it, you put it down. If it breaks, you get a different hammer. We feel the same way about large language models.

> *“You do not need to have faith in a hammer.”*

03.2 On Intentional Development

Every dependency is a trust decision. Every abstraction is a debt instrument. Every framework is a bet on someone else's roadmap.

We ship deliberately. This does not mean we ship slowly. It means we think about what we are shipping before we ship it. We read the changelogs. We audit the dependencies. We ask whether this library that saves us forty lines of code is worth adding an organization we have never met to our supply chain.

Most of the time the answer is yes. Dependencies exist for a reason and reinventing the wheel is not a personality. But we want the answer to be a conscious decision, not an accident of muscle memory. npm install is a power tool and power tools deserve respect.

We prefer boring technology for infrastructure. The database should be boring. The deploy pipeline should be boring. The auth layer should be boring. Save the interesting decisions for the product, where interesting actually matters.

03.3 On Stack Research

Stack pragmatism over stack loyalty. We do not have a favorite language. We have a favorite outcome: software that works, that we can maintain, and that does not make us mass-close GitHub issues at 3 AM.

Our current stack is Next.js, TypeScript, Tailwind, and Cloudflare. This is not because we believe these are the best tools ever made. It is because they are good enough tools that we understand well enough to ship with confidence. If something better comes along, we will evaluate it honestly and migrate if it makes sense. We will not migrate because a conference talk made it sound exciting.

We are deeply suspicious of technology choices driven by resume optimization. If you pick a tool because it will look good on your LinkedIn, you have optimized for the wrong metric. Pick tools that make the product better. Your LinkedIn will survive.

> *“Boring infrastructure is reliable infrastructure.”*

03.4 On Humor as Infrastructure

The humor is not the product. The humor is how we stay sane while building the product.

Most of the tech industry is performing. Companies perform seriousness. Founders perform vision. Marketing copy performs enthusiasm nobody feels. The performance is exhausting, and it is so universal that simply not performing reads as a brand strategy. It is not a strategy. It is just honesty, and honesty is funny when most of the industry is doing whatever the opposite of honesty is.

We write our landing pages the way we write our Slack messages. If that tone makes you uncomfortable, you are probably not our customer, and that is fine. We would rather have fewer customers who get it than more customers who think we are trying too hard. We are trying exactly the right amount.

The sarcasm is not a wall. It is a door. It says: we are humans who build software, and we find the gap between how this industry presents itself and how it actually works to be genuinely, consistently funny.

03.5 On People

Software is made by people, used by people, and breaks because of people. Every interesting problem in software engineering is eventually a people problem.

We build software for people who care how it works. That narrows the market and improves the product. Our users read documentation. Our users file useful bug reports. Our users understand that software has edges and corners and that perfection is a process, not a release.

Internally, we value clarity over charisma, competence over credentials, and shipping over talking about shipping. We do not do stand-ups because we are standing all the time. We do not do retros because our retro is just reading the error logs and wincing.

We would rather work with people who are quietly good at their jobs than people who are loudly good at describing their jobs. The tech industry has too many main characters. We are looking for supporting cast who actually read the script.

> *“Every interesting problem in software engineering is eventually a people problem.”*

03.6 On Anonymity

The people behind The Null Hypothesis Foundation choose to remain semi-anonymous. This is not secrecy for the sake of mystique. It is a deliberate decision about how we want our work to be evaluated.

Software should be judged by what it does, not by who made it. The industry has a persistent habit of assigning value to people based on where they worked before, what school they attended, or which company logo sits next to their name on LinkedIn. We find this unhelpful. A good commit does not become better because it was written by someone who used to work at a company you have heard of. A bad architecture does not become good because the architect has a prestigious degree.

We would rather you use our tools, read our code, file a bug report, and form an opinion based on the thing itself. Not on whether we seem credible enough to have built it. Credibility is earned at the function level, not the biography level.

This may change in the future. If it stops serving us or starts looking like we are hiding something rather than simply preferring to let the work speak, we will reconsider. For the time being, such is the nature of The Null Hypothesis Foundation.

> *“Credibility is earned at the function level, not the biography level.”*

[PRINCIPLES.md]

Principles

04

Four operating principles. They are not values, because values are what companies print on walls and then ignore during quarterly planning. These are closer to habits. Things we actually do, not things we aspire to.

01 Privacy by paranoia

We don't collect your data. Not because we're ethical — we just don't want the liability. Every tool runs local. Your packets, your problems.

02 Ship it, then read the error logs

Perfect is the enemy of deployed. We believe in learning from production, mostly because staging never works anyway. Also we can't afford staging.

03 AI is a tool, not a personality

We use it where it helps. We don't use it where it doesn't. We will never describe ourselves as 'AI-native' or 'AI-first.' We are 'AI-sometimes.' The rest is just code we wrote ourselves, like animals.

04 Revenue would be nice

We're an independent shop trying to make products people actually pay for. No VC runway. No pivot-to-enterprise. Just shipping software and hoping the math works out before the savings don't.

[CLOSING.md]

Closing Transmission

05

This document will be updated as we learn more about what we are doing. Given that we are still figuring it out, expect revisions.

If any of this resonated, we are probably building something you might want to use. Check the projects. Try the tools. If they work, tell someone. If they do not work, tell us. We read every email, mostly because we do not get that many.

We are a small company, building real software, trying to stay honest about the process. That is the whole pitch. There is no second page of the pitch deck. There is no slide with a hockey stick graph. There is just this, and the things we ship, and the hope that the two are enough.

> EOF_

The Null Hypothesis Foundation
root@nullfoundation.org
nullfoundation.org

∅

The Null Hypothesis Foundation

𝕏 @RejectNull // reject the null hypothesis

Terms of Service Privacy (Pending) Security (Ironically) Status Page (It's Fine)