Build Anyway.

A Business Owner's Guide to Leading Through Technology You Don't Fully Understand

JEFF GIPSON

Technology Leader · Product Builder · AI Strategist

BUILD ANYWAY

A Business Owner's Guide to Leading Through Technology You Don't Fully Understand

Jeff Gipson

CONTENTS

FOREWORD

Steadfast, Immovable, Always Abounding

I want to tell you something about myself before you read this book.

I am a person of faith. I have been for as long as I can remember. I grew up in a home where faith was woven into the fabric of daily life, and it has stayed woven into mine — through the welding shop, through the early years of building a company in my father's basement, through the acquisition, through the sleepless nights of building software I didn't yet know how to build.

I don't lead with that in business settings very often. Not because I'm ashamed of it, but because I believe that what I've built and how I've built it should speak for itself. Results don't require a theological footnote.

But this is a foreword, which means it's the part of the book where I get to say what I actually think before we get into the practical business of the chapters that follow. And what I actually think is this: the way I have approached every hard thing in my career — every pivot, every rebuild, every moment of not knowing if it was going to work — has been shaped by something I read in scripture a long time ago and have never been able to get out of my head.

> *"Be ye stedfast, unmoveable, always abounding in the work of the Lord — forasmuch as ye know that your labour is not in vain." — 1 Corinthians 15:58 (KJV)*

Steadfast. Immovable. Always abounding.

I'm not going to soft-pedal what drives me. Faith is not a footnote in my life — it is the foundation. It shapes how I work, how I lead, how I respond when things fall apart, and how I keep going when the quiet voice at the end of a long day asks whether any of it is going to work. That foundation is real, and this foreword is the place to say so.

What I will also say is this: you don't have to share my faith to get something from what I've learned. Because those three principles — steadfast, immovable, always abounding — have functioned for me as something very close to an operating philosophy. Not just spiritually. Professionally. In every build, every setback, every moment where the tester feedback said nothing is working and the quiet voice at the end of a long day asked whether it was ever going to get there.

Steadfast means you don't change course every time the wind shifts. You commit to a direction, you do the work, and you trust the process even when the process is painful.

Immovable means you don't let doubt, criticism, or the size of the obstacle in front of you knock you off what you've committed to. You can acknowledge the obstacle. You can study it, work around it, go through it. But you don't let it move you off the work.

Always abounding means you don't do just enough. You give the work everything you have. A million lines of code. Eight thousand commits. Problems that won't let you sleep because your mind is still working on them long after the day ends. You abound in the work — not because someone is watching, but because the work deserves it.

> *Steadfast. Immovable. Always abounding. Three principles that describe every builder I have ever respected.*

I believe those three principles describe every builder I have ever respected — regardless of their faith, their background, or the industry they work in. They describe my father, who built businesses with integrity for decades and passed something essential down to me in a phrase I spent years unpacking. They describe my cousin Matt, who showed up to learn something he'd never done and is still showing up twenty years later. They describe the developers on my team who are doing the hard work of figuring it out when it would be easier to stop.

They describe, I hope, the person reading this book.

I didn't set out to write a book. I'm a builder, not a writer. What I set out to do was document something — a story, a career, a set of hard-won lessons about what it actually takes to build something real in a moment when the tools are changing faster than most people can track.

The business owners I think about most when I write are the ones who are where I was ten years ago. Smart, capable, experienced in their industry — and staring down a technology landscape that feels like it belongs to someone else. Someone younger, someone more technical, someone who grew up with this stuff in a way they didn't.

I want to tell those business owners: it doesn't belong to someone else. It belongs to anyone willing to be steadfast, immovable, and always abounding in the work of figuring it out.

That's a posture, not a skill set. And it's available to anyone.

> *The tools belong to anyone willing to be steadfast, immovable, and always abounding in the work of figuring it out.*

The chapters that follow are practical. They're full of real stories, real numbers, and real lessons about building with AI that I hope you can apply to your own organization. They don't require you to share my faith or my background or my particular path through the staffing industry.

They just require the same thing every hard thing requires: the decision to start, the commitment to keep going, and the refusal to be moved by the inevitable moments when it isn't working yet.

Your labor is not in vain.

I have staked my career on that promise. I believe it with everything I have. And I hope that by the time you finish this book, you'll have a few more reasons to believe it too.

There's one more verse I'll leave you with before we get into it. It's from Ecclesiastes, and it's short enough that I won't spend much time explaining it. I think it explains itself:

> *"Whatever your hand finds to do, do it with all your might." — Ecclesiastes 9:10*

That's the whole book, really. Whatever you're building — do it with all your might.

— Jeff Gipson
Jackson, Missouri

INTRODUCTION

Now We Actually Have to Build Something

I made my very first sales call to sell a website. And I sold it.

Nine thousand, six hundred dollars. First call. Done.

The first thought I had after I hung up the phone was: *Oh crap. Now we actually have to build something.*

Here's what made that moment interesting: I had never built a website before in my life.

> *I had never built a website before in my life.*

I wasn't a developer. I didn't have a computer science degree. I hadn't spent years learning the craft. What I had was a problem I wanted to solve, a phone that worked, and a father who had spent my entire life telling me the same thing:

"If you want something, pretend like you already have it."

Fake it till you make it.

WHAT THIS BOOK IS ABOUT

This is not a technology book. I want to be clear about that upfront.

I'm not going to teach you how to write code, how to build an AI model, or how to architect a software platform. There are plenty of books for that, and frankly, the people who write them are far more qualified than I am.

This is a book about something harder than learning technology. It's about leading through technology you don't fully understand — and doing it anyway.

Because that's what business ownership actually looks like most of the time. You're not the expert in the room. You're the person responsible for the room. And right now, if you own a business of any size, you are being asked to make decisions about artificial intelligence, about automation, about software — and you probably feel at least a little bit like I felt after that first phone call.

Oh crap. Now we actually have to build something.

The difference between the business owners who figure it out and the ones who don't isn't technical knowledge. It's something closer to what my father was describing all those years ago. It's the willingness to say yes before you're ready — and then go figure out how to make it true.

WHO I AM

I started my working life as a welder. Twelve-hour shifts, six days a week. I was good at it, and I was paid well for it, but I hurt my back and had to find something else.

That pivot took me through sales, through executive recruiting, and eventually to co-founding a company with my father called Recruiters Websites — a web design and digital marketing firm for the staffing industry. We started it in 2012. I had never built a website before. We grew it to nearly a million dollars in annual payroll, and in 2022 we sold it to Haley Marketing Group, the largest marketing firm in the staffing industry.

Today I sit on the leadership team at Haley Marketing Group. I lead product development for RogIQ, an AI-powered marketing platform built to transform how companies grow. My team recently rebuilt a chatbot from scratch in about fifteen hours — a chatbot that the traditional approach had consumed nearly two years and a substantial budget trying and failing to deliver.

I'm forty years old. I'm mostly self-taught. And I'm still figuring it out.

That's kind of the point.

HOW TO READ THIS BOOK

Each chapter in this book is built around a real story from my career — a moment where I didn't know what I was doing, did it anyway, and learned something I couldn't have learned any other way.

Along the way, I'll draw out the lessons that I think apply to any business owner navigating technology right now. Not because my path was the right one — I made plenty of mistakes, and you'll hear about those too — but because the mentality that got me through is the same one I believe will get you through.

We'll talk about why planning matters more than building. Why clarity is more valuable than technical skill. Why the business owners who are winning with AI right now aren't necessarily the most technical ones — they're just the ones who started.

According to research cited by bestselling author Geoff Woods in *The AI-Driven Leader*, 100% of companies believe that AI is the future of their business. Only 5% are actually doing anything about it.

This book is for the other 95%.

> *Figure out what is impossible. Then do it.*

That's what my dad's phrase always came back to for me. Not deception. Not pretending to be something you're not. Just the audacity to move before you have all the answers.

Let's get started.

— Jeff Gipson

Jackson, Missouri

CHAPTER ONE

Building Is Building

Before I built websites, before I led software teams, before I had any idea what artificial intelligence was or what it could do — I was a welder.

Twelve-hour shifts. Six days a week. It was hard work and I was paid well for it, and for a while, I genuinely loved it. There's something about welding that not everyone understands until they've done it. You're taking raw material and making it into something that didn't exist before. You're building. And when you step back at the end of a shift and look at what you made, there's a satisfaction in that that's hard to replicate.

I didn't leave welding because I wanted to. I left because my back gave out. I was facing surgery that I refused to have, which meant I needed to find another way to earn a living. So I did what most people do when they don't know what's next: I tried a few things.

Sales. I'd always been good at talking to people, so I tried sales. Sold cars for a while. It was okay. Not the kind of sales I wanted, but it was something.

Then my father offered me a different path.

MY FATHER'S BUSINESS

My dad had spent his career in the staffing and insurance industries. By the time I came to work with him, he had built an executive search firm specializing in the insurance industry. He brought me on, and I worked there for about ten years.

I was okay at it. Probably could have been better if I'd really loved it. But recruiting wasn't my passion. I did the work, I learned the business, I got good enough — and the whole time, something was quietly building in the back of my mind.

My dad needed a website.

That sounds like a small thing. It wasn't.

THE DEVELOPER WHO CHANGED EVERYTHING

We hired a local developer to build the website for my dad's company. The process was painful from the start. Communication was difficult. Getting anything done took forever. Deadlines came and went. And when the site finally arrived, it turned out to be largely boilerplate — a template dressed up to look custom.

The thing that stuck with me wasn't the mediocre product. It was the combination: mediocre product, painful process, and a price tag that didn't match either one. We found out later that most of what she'd built was off-the-shelf. Which raised an obvious question: if it's all boilerplate, why did it take so long?

> *She didn't out-skill us. She just had the job. And I thought — I can do this better.*

I looked at the staffing and recruiting industry around that same time and noticed something. The websites were, almost without exception, old, ugly, and outdated. An entire industry that was in the business of connecting people with opportunity — and their online presence looked like it was built in 2003 and never touched again.

Those two observations collided into an idea. What if we built a company that did this right? Better communication. Better product. Built specifically for the staffing and recruiting world.

That was the vision for Recruiters Websites. My dad and I started it in 2012.

I had never built a website before in my life.

FIGURING IT OUT

My first hire was my cousin, Matt McKenzie. Matt came from a graphic design background — he'd been working for a local sign company. He wasn't a web developer either. But he understood design, he was willing to learn, and honestly, he was someone I trusted.

Matt still works with me today, at Haley Marketing Group. That's not a small thing. When you build something from nothing with someone, and more than a decade later you're still working together — that tells you something about how you both approached the work.

What we did in those early days was simple: we figured it out. We started with templates we found on the web. We knew we wanted to build on WordPress. And then we taught ourselves everything we needed to know — CSS, HTML, PHP, whatever the job required. We'd run into a problem, solve it, learn from it, and move on to the next one.

There was no master plan. There was a client, a deadline, and two people determined not to fail them.

> *There was no master plan. There was a client, a deadline, and two people determined not to fail them.*

We grew slowly at first, and then faster. We added digital marketing. We built API integrations. We took on projects I didn't know how to do until I was already halfway through doing them. Over ten years, we built hundreds of websites for staffing and recruiting companies across the country.

I also took a six-month Ruby on Rails development course at a local co-working space during that time. Liked it enough that I ended up teaching the course for a few years afterward — two nights a week, helping adults who came in with no technical background walk out six months later ready to find jobs as developers. That role lit something up in me that I still carry today. There's something powerful about watching someone discover they're capable of more than they thought.

THE THING I DIDN'T REALIZE

It took me years to understand something about my own story.

I didn't change careers when I left welding. Not really. I kept doing the same thing I'd always done. I just changed the material.

In the welding shop, I was taking raw metal and building something that hadn't existed before. In the website business, I was taking a blank screen and building something that hadn't existed before. The satisfaction was the same. The process was the same. Show up, figure out what's needed, build it, and make sure the person who asked for it walks away better off than they were before.

I think this matters for the business owners reading this book, because a lot of people assume that technology is a different kind of thing — that it requires a different kind of person. Someone born into it. Someone who grew up writing code and dreaming in algorithms.

That hasn't been my experience. My experience is that building is building. The tools change. The principles don't.

> *Building is building. The tools change. The principles don't.*

What makes someone effective with technology isn't a background in technology. It's the mentality of the person holding the tool. Are they willing to figure it out? Are they

willing to fail, learn, and try again? Are they more focused on the outcome for the customer than on protecting themselves from looking inexperienced?

Those aren't tech skills. Those are character traits. And they're learnable at any age, at any stage of a career.

I know because I learned them in a welding shop.

———

WHAT THIS MEANS FOR YOU

Building is building. The tools change, the principles don't. You don't need a technical background to lead through technology — you need the mentality of someone willing to figure it out. The skills that make someone effective with technology are character traits, not credentials, and they're learnable at any age.

In the next chapter, we'll talk about what happens after the first sale — and why rebuilding from scratch eight times in one year turned out to be the best education I ever got.

CHAPTER TWO

Failing Forward (Eight Times)

John Maxwell wrote a book called *Failing Forward*. The core idea is simple: failure isn't the opposite of success. It's the path to it. What separates people who ultimately achieve something from people who don't isn't whether they fail — everybody fails — it's whether they fail in a direction that moves them closer to where they're trying to go.

I didn't read that book before any of it. I lived it — from the welding shop to recruiting, from recruiting to Recruiters Websites, from Recruiters Websites to the acquisition, and from the acquisition to building RogIQ. Every transition in my career has been a version of the same lesson: failure isn't a detour from the path. It is the path.

The sharpest example of that lesson came in the first year of building RogIQ — the AI platform I'll tell you more about in later chapters — where we rebuilt the entire application from scratch eight times. Eight complete restarts. Eight times we got far enough into the build to realize that what we had wasn't what we needed, tore it down, and started over.

Eight times.

From the outside, that probably sounds like chaos. From the inside, it was the best education I've ever received.

WHAT FAILURE ACTUALLY COSTS

There's a common assumption in business that failure is expensive. And sometimes it is — financially, reputationally, relationally. I'm not going to pretend that every failure is painless or that there are no real consequences.

But there's another kind of cost that doesn't get talked about as much: the cost of not failing. The cost of moving so carefully, so cautiously, that you never actually learn what you need to learn to build something great.

When Matt and I started building websites together, we didn't know what we were doing. We used templates. We made mistakes. We delivered things that, looking back, weren't as good as they should have been. But every project taught us something the previous one hadn't. Every client interaction showed us a gap we hadn't seen before. Every problem we ran into became a problem we knew how to solve the next time.

> *Every problem we ran into became a problem we knew how to solve the next time.*

That compounding effect — where each failure makes you incrementally more capable — is one of the most powerful forces in business. But it only works if you're actually in motion. You can't compound lessons you never earned.

The business owners I've watched struggle most with technology aren't the ones who tried something and failed. They're the ones who waited. Who studied. Who attended one more conference, read one more article, hired one more consultant to tell them what they already suspected — that they needed to do something — and then went back to waiting.

Waiting feels safe. It isn't.

THE DISCOVERY THAT CHANGED EVERYTHING

Here's what we learned from rebuilding RogIQ eight times: planning matters more than building.

That sounds obvious when you say it out loud. It wasn't obvious to us at the start. At the start, we did what most people do when they're excited about a new idea: we went straight to building. We had a vision, we had energy, and we started writing code.

The problem with jumping straight to building — especially when you're building with AI-assisted development tools — is that the AI can only be as good as the instructions you give it. If you can't clearly describe what you're building, where it's going, and how a specific feature should behave from input to output, you're going to get something that technically works but doesn't actually do what you need.

We learned this the hard way. Multiple times.

> *We found out that planning was more important than developing. There was another 90% after the first 90% was complete.*

That second sentence is the one that will humble you if you let it. You think you're almost done. The feature looks like it's working. And then you actually use it — really use it, the way a customer would — and you realize there's an entire other layer of complexity you hadn't accounted for. Edge cases. Error states. The thing that happens when a user does something you didn't anticipate.

That second 90% is where most of the real work lives. And you can't get there faster by building faster. You get there faster by thinking harder before you start.

HOW TO ACTUALLY PLAN

I want to be specific here, because "plan more" is advice so general it's almost useless.

What we found works is this: don't try to plan the whole application at once. Narrow it down. Pick one feature. One specific piece of functionality. And then answer three questions about that feature before you write a single line of code:

What should it do? Not in general terms — specifically. What is the user trying to accomplish? What action do they take to start the process, and what does the end state look like when it's working correctly?

How should it function? What are the steps between start and finish? Where does data come from? Where does it go? What should happen if something goes wrong?

What should the output be? Exactly. Not approximately. If the feature generates content, what does good content look like versus bad content? If it processes data, what format should that data be in when it comes out the other side?

When you can answer those three questions clearly — in plain language, without jargon — you're ready to build. And if you're using AI tools to help you build, those answers become the context the AI needs to actually produce something useful.

> *Clarity is the skill. Not coding. Clarity.*

This is the insight I most want business owners to take from this chapter. Because clarity is something every business owner already knows how to develop. You do it every time you write a job description, brief a vendor, or explain your value proposition to a new prospect. You're describing what you need, how it should work, and what success looks like.

Building with technology — including AI — is the same process. The technology just executes on your clarity instead of a human being. This became the foundation for how we eventually structured RogIQ — not as a tool that generates content, but as a system that executes on clearly defined outcomes.

Around the time Recruiters Websites was hitting its stride, I started teaching a six-month Ruby on Rails development course at a local co-working space. Two nights a week, plus some extra time when students needed additional help.

The students in that course came in with nothing. No technical background, no coding experience, no idea whether they could actually do this. Six months later, they were job-ready developers.

I loved that role more than I expected to. There was something about watching someone go from "I don't know if I can do this" to "I just built that" that never got old. It reminded me of what it felt like when Matt and I figured out our first WordPress site. That moment when the thing works and you realize you were more capable than you thought.

What I noticed in those students was the same thing I'd noticed in myself: the ones who succeeded weren't necessarily the most naturally talented. They were the ones who were willing to be bad at something long enough to get good at it. The ones who didn't interpret a mistake as evidence that they didn't belong.

Victoria Kenward, Co-CEO of Haley Marketing Group, has a phrase she comes back to often: *"Feedback is a gift."* The first time I heard it, it sounded like something you put on a motivational poster. The more I've thought about it, the more I think it's one of the most practically useful ideas in business.

Feedback — including the feedback you get from failing at something — is information. It tells you what's not working so you can figure out what will. The people who treat that information as a threat are the ones who stop learning. The ones who treat it as data are the ones who keep getting better.

> *"Feedback is a gift." — Victoria Kenward*

Eight rebuilds. Hundreds of websites. Years of teaching. All of it pointing toward the same conclusion: the willingness to be wrong, openly and repeatedly, is one of the most valuable assets you can bring to any new challenge.

Including the challenge of figuring out what AI means for your business.

———

WHAT THIS MEANS FOR YOU

Before you build anything, answer three questions: What should it do? How should it function? What should the output be? Planning matters more than building — especially with AI tools. Clarity is the skill, not coding. Treat failure as data, not a verdict.

In the next chapter, we'll talk about a different kind of lesson — the one that came from knowing when to let go. How a family business becomes an acquisition, why I told the broker exactly who to call, and what happened in that office on the day the deal closed.

CHAPTER THREE

Benchmarking the Competition (Then Selling to Them)

From the very beginning of Recruiters Websites, we had our eyes on Haley Marketing Group.

They were the largest marketing firm in the staffing industry. Well-established, well-respected, doing work we admired. When we were trying to figure out what kind of company we wanted to be and what the market expected from a firm like ours, Haley was the benchmark. We studied what they did, how they positioned themselves, what their clients said about them. Not to copy them — but to understand the standard.

Ten years later, I sold my company to them.

And I was the one who told the broker to make the call.

KNOWING WHEN TO LET GO

By 2022, Recruiters Websites had grown to nearly a million dollars in annual payroll. My father, who had been my partner since the beginning, was looking toward retirement. He came to me and told me he was thinking about selling the website division.

My first instinct was to buy it myself. Take it over, keep building it, see how far I could take it. That's the founder's reflex — hold on, don't let go, this thing is yours.

But when I actually looked at the numbers and what it would take to carry a million dollars in payroll on my own, I knew it wasn't the right move. Not because I couldn't have figured it out eventually — maybe I could have — but because the risk wasn't

proportionate to where I was or where I wanted to go. Letting that instinct override the math would have been ego, not strategy.

> *Letting that instinct override the math would have been ego, not strategy.*

So we worked with a business broker. They set us up to interview with a couple of potential buyers. We went through the process. And none of it felt right.

I don't know exactly how to describe what "not feeling right" means in a business context — it's part instinct, part pattern recognition from years of working with clients and partners. You just know when you're sitting across from someone who sees the world differently than you do, and you know that difference is going to matter.

So I went back to the broker and told them to contact David Searns at Haley Marketing Group.

THE COMPANY YOU'VE BEEN WATCHING

There's something disorienting about finally meeting the person whose company you've been studying for a decade. You know their work. You've thought about them strategically. You've positioned your own business partly in response to what they've built. And then you're sitting across a table from them, and they're just a person.

David's first priority in that conversation wasn't financials. It was fit. Values fit. Cultural fit. Were we the kind of people and the kind of company that belonged inside what Haley Marketing Group was building? That question, coming first, told me something important about how he led.

The due diligence process that followed was extensive. Attorneys. Accountants. Document requests. Financial records. Projections. The kind of scrutiny that a large, established company applies when it's considering bringing a smaller one into the fold.

On our side of the table: my mother was doing our payroll and accounting.

> *On our side of the table: my mother was doing our payroll and accounting.*

I say that not as an embarrassment but as a point of pride. We were a family business in the fullest sense of the word. My father co-founded it. My cousin Matt built it with me from day one. My brother Steve was doing sales. My mother kept the books. We didn't have a finance department or a legal team on retainer. We had people who cared about what we'd built and did whatever the business needed them to do.

Pulling together what David needed during that due diligence process was a significant effort for a company our size. But we did it. Because that's what the situation required, and figuring out what the situation requires and then doing it is just how we operated.

FLORIDA

Somewhere in the middle of the process, David invited us down to Florida to meet in person. My brother Steve came with me. We sat down with David and Victoria Kenward — his wife and Co-CEO — and talked through what a partnership would actually look like. Not just the transaction, but the future. What we'd be building together. What role we'd play inside a larger organization. Whether the vision aligned.

We didn't walk out of that meeting with a signed deal. But we walked out knowing that this was probably where we were headed. The conversation had a different quality than the others we'd had during the process. It felt less like an interview and more like a planning session.

Sometime shortly after that meeting, we came to an agreement. The acquisition was happening.

THE DAY OF THE CLOSE

On the day the acquisition was finalized, I was the only one in the office.

Everyone else had the day off. The transition between payrolls meant there was no reason for the team to come in — legally, they weren't being paid by either entity that day. So I sat in a building we had built together, by myself, and watched the paperwork go through.

I became emotional. Not out of sadness, and not out of any kind of overwhelming happiness. Something in between, or maybe something entirely different — I still don't have the right word for it. There's a particular feeling that comes with the end of a chapter you've given a decade of your life to. It doesn't map cleanly onto any single emotion.

What I kept coming back to, sitting there alone, was my dad.

> *Dad would have reassured me that we did the right thing, that it was time, and that he was proud of me.*

I believe that. I believed it then and I believe it now. We did the right thing. We built something real, grew it as far as we could responsibly take it, and found it a home where it could keep growing. That's not failure. That's the whole point.

WHAT THIS HAS TO DO WITH YOU

I've told this story in some detail because I think it contains a lesson that goes beyond acquisitions and family businesses.

The lesson is about knowing where you want to land before you start moving.

When we decided to sell, I didn't just hand the decision to a broker and accept whoever showed up. I had a specific destination in mind based on ten years of watching the market, understanding our industry, and knowing what kind of organization shared our values. I told the broker exactly who to call. That's not luck. That's the result of paying attention for a long time.

The same principle applies to technology decisions. The business owners who navigate this moment well aren't the ones who chase every new tool or react to every headline. They're the ones who know what they're building, where they want to end up, and which technologies actually serve that destination.

That requires the same thing the acquisition required: clarity about what matters, patience with the process, and the willingness to say no to things that don't fit — even when those things are technically available to you.

Fake it till you make it doesn't mean say yes to everything. It means move with intention toward something specific, even when you don't have all the answers yet.

> *Move with intention toward something specific, even when you don't have all the answers yet.*

I knew from the beginning of Recruiters Websites that Haley Marketing was the standard. I just didn't know yet that one day I'd be sitting at their table, not as a competitor, but as part of the team.

Pay attention to who's setting the standard in your space. You never know where that attention leads.

———

WHAT THIS MEANS FOR YOU

Know where you want to land before you start moving. Pay attention to who is setting the standard in your space. Saying no to what doesn't fit is as important as saying yes to what does. Move with intention toward something specific, even when you don't have all the answers yet.

In the next chapter, we'll pull back the curtain on what building something real actually looks like — the million lines of code, the sleepless nights, the moments of doubt, and what it costs in real life to build something you believe in.

CHAPTER FOUR

A Million Lines of Code

Nobody talks about the hard part.

They talk about the launch. The demo. The jaw-drop moment in the planning meeting. They talk about what got built and what it does and what it means for the business. Those are the parts worth sharing, and I've shared them in this book.

But there's another part of the story. The part that lives between the idea and the outcome. The part that happens when something isn't working and you don't know why and the testers are telling you that nothing is working and the quality isn't there and you're starting to wonder, quietly, if it's ever going to get there.

I want to talk about that part. Because if you're going to build something real — with AI or without it — you're going to live in that part for a while. And knowing that it's normal, that it's part of the process, that the people who got through it aren't the ones who never doubted but the ones who kept going anyway — that might be the most useful thing in this book.

WHAT SIX MONTHS ACTUALLY LOOKS LIKE

To understand what RogIQ V2 is, you have to understand what V1 was first.

V1 was over a year of building, deploying internally to our own marketing teams, learning, and discovering. It was never meant to be the final product — it was the curriculum. V1 is where we figured out how to work with AI as a development partner. It's where we learned, the hard way, that planning wasn't just important — it was everything. You cannot give an AI tool vague instructions and expect precise output. The

quality of what you build is directly proportional to the quality of how you think before you build.

V1 is also where we discovered what the product actually needed to be. What features mattered to real users and which ones only seemed like good ideas from a distance. What the marketing teams who used it every day actually needed versus what we thought they needed when we started. That gap — between assumed requirements and real ones — is where most products fail. V1 closed that gap for us, at the cost of a year of effort, before we ever wrote a line of V2. What we were really building wasn't just a product. It was a system for turning clarity into execution — which is ultimately what RogIQ became.

> *V1 wasn't a failed product. It was a paid education. Without it, V2 wouldn't have been possible.*

V2 is a six-month build. But it's built on an eighteen-month foundation. In those six months, my team has written over a million lines of code. We've logged nearly 8,000 commit messages to the repository — each one a small record of a problem solved, a feature added, a bug fixed, a decision made.

In 2025, I ranked in the top 10% of all Cursor users worldwide. Cursor is the AI-assisted development environment my team lives inside. The top 10% ranking isn't a badge I'm sharing to impress anyone. I'm sharing it because it's a measure of something real: the sheer volume of time and effort that went into learning how to use these tools at the level where they actually change what's possible. You don't land there by dabbling. You land there by outworking the problem.

A million lines. Eight thousand commits. Top 10% of a platform used by developers worldwide. Six months of V2 — and a year of V1 that made it possible.

From the outside, that might sound like a lot of output. From the inside, it feels like a river that never stops moving. There is always something that needs attention. Always a bug that surfaced in testing, a feature that needs refinement, an edge case that nobody

anticipated until a real user ran straight into it. The application is never done. It's only ever at the current version of done, which is different from the version of done that's coming.

> *The application is never done. It's only ever at the current version of done.*

I have three other people working with me. Good people. People who care about what we're building and bring real skill to the work. But at this stage of the build — the founder-level, get-it-off-the-ground stage — the weight lands here. That's not a complaint. It's just the truth of what it means to be the person responsible for something that doesn't exist yet. Someone has to hold the vision. Someone has to make the call when there's no clear right answer. Someone has to still be working when everyone else has stopped.

Most nights, that's me.

THE RABBIT HOLES

Building with AI is not the frictionless experience that some people describe it as. The tools are powerful — genuinely, remarkably powerful — but power without direction is just chaos moving fast.

The rabbit holes are real. You pull on one thread and it leads somewhere you didn't expect. You solve one problem and the solution reveals three more. You get deep into a particular approach — an hour in, two hours in — and then realize the approach is wrong and you have to back out and start from a different direction. The AI will follow you down every one of those holes with equal enthusiasm. It doesn't know that you're headed the wrong way. That's still your job.

There's a particular kind of exhaustion that comes from this kind of work. It's not physical exhaustion, although the late nights contribute to that. It's the exhaustion of sustained decision-making. Of holding a complex system in your head and trying to see all of its moving parts at once. Of caring, deeply, about whether something works — and then going to bed with it unresolved and waking up still thinking about it.

I've gotten up in the middle of the night because a solution came to me. Not an alarm, not a notification — just a thought, surfacing out of sleep, that showed me a different way to approach the problem I'd been stuck on. I've learned to keep something nearby to write it down. I've learned that those middle-of-the-night insights are sometimes the best ones, because the part of my brain that was forcing the problem has gone quiet and something else has had a chance to work on it.

NOTHING IS WORKING

There was a period during the RogIQ V2 build when the tester feedback was brutal.

Nothing is working. Everything we tried is failing. The quality isn't good enough.

That feedback didn't come from people who wanted the product to fail. It came from people who wanted it to succeed — which made it harder to hear, not easier. When someone invested in your success tells you that it isn't working, you can't dismiss it. You have to sit with it.

Part of what made those moments so disorienting was something Victoria Kenward put into words better than I could: *"We have to figure out what done is."* It sounds simple. It isn't. When you're deep inside a build, with moving pieces and evolving requirements and testers pushing back from every direction, the definition of done keeps shifting. And when you don't have a clear, shared picture of what done actually looks like, you can be

90% of the way there and have no idea — because there's another 90% waiting on the other side of what you thought was the finish line.

> *"We have to figure out what done is." — Victoria Kenward*

That second 90% is the part nobody warns you about. The feature works. The output looks right. And then a real user touches it and you discover an entire layer of complexity you hadn't accounted for — edge cases, error states, the thing that happens when someone does something you didn't anticipate. The gap between "it works" and "it's done" is where builds go to die if you're not honest about it upfront.

I sat with all of it.

The honest truth is that there were moments during that period where I didn't know if we were going to get there. Not loud, dramatic doubt — just a quiet, persistent question that showed up at the end of long days: *Is this actually going to work?*

What got me through it wasn't certainty. I didn't have certainty. What I had was the same thing I've had at every hard point in my career: the inability to quit on something I believed in. A tolerance for discomfort that I'd built up over years of building things that didn't work before they worked.

And the knowledge, from experience, that the wall usually precedes the breakthrough.

THE SHIFT

The shift came when we stopped pushing and started thinking.

We took a step back. Instead of continuing to iterate on what wasn't working, we looked at the problem differently. Changed our approach to how we were working with the AI. Reframed what we were asking it to do and how we were asking it.

Within a few days, the product that had been producing output nobody wanted was producing marketing material that was genuinely impressive. Not "good enough." Actually good. The kind of output that made the testers, who had been the harshest critics, say: *okay, now we're talking*.

A few days. After weeks of struggle.

That's the part of building with AI that nobody prepares you for: the breakthroughs are not always gradual. Sometimes you're stuck, and stuck, and stuck — and then something clicks and the thing that seemed impossible last week is working today and you're looking at what you built and thinking, *I cannot believe this exists and we made it*.

> *I cannot believe this exists and we made it.*

That feeling is worth every 2am. Worth every rabbit hole. Worth every piece of feedback that said it wasn't good enough yet.

I don't know how to manufacture that feeling for you. I can only tell you it's real, and that it's waiting on the other side of the hard part, and that the only way to get there is through.

THE COST IN REAL LIFE

I want to be honest about something that business books don't always say out loud.

This cost something. Not just professionally. In real life.

I have a wife I've been married to for over twenty years. Five kids. A family that matters to me more than anything I've built or will ever build. And for the better part of two years — the V1 year of learning, and now the V2 year of building — I have been partially somewhere else. Not absent — I'm present, I show up, I'm there — but there's a part of

my mind that is always on the application. Always running in the background. Always half a step away from the problem I left on the desk.

My family understands what drives me. They know that when I'm in this mode, it's not that I've checked out — it's that I'm building toward something I believe in, something that matters for our future and for the company and for the people who depend on the product working. They give me that grace, and I don't take it for granted.

But I'd be doing you a disservice if I told you that building something real — building something that matters — is free. It isn't. It costs time and sleep and presence and mental energy. The return on that investment is real. So is the investment.

> *Building something that matters is not free. It costs time and sleep and presence and mental energy. The return is real. So is the investment.*

This is also an investment on Haley Marketing Group's side. They've committed resources, time, and trust to building this platform. That weight is something I carry, and it should be. When other people have bet on what you're building, you owe them everything you've got.

What I want to say to anyone reading this who is in the middle of their own hard build: the cost is real, and it's worth naming. Don't pretend it isn't there. Don't minimize it to the people around you who are feeling it too. Be honest about what it's taking — and be honest about why it's worth it.

For me, it's worth it because I can see what we're building. I can see what it does for the clients who use it. I can see where it's going. And I know, from everything that came before this, that the things worth building are never the easy ones.

WHAT AI DOESN'T CHANGE

I want to push back gently on something you might have heard about AI-assisted development — the idea that it removes the hard parts.

It doesn't.

What it does is compress the timeline and amplify your output. Work that used to take weeks can take days. Problems that used to require specialists can be approached by generalists with the right tools and the right mindset. The ceiling on what a small, motivated team can build has been raised dramatically.

But the hard parts — the decisions, the doubt, the sustained effort, the willingness to tear something down and rebuild it, the 2am problem-solving, the emotional weight of responsibility — those are still there. AI doesn't carry those for you. It just makes sure that when you show up to do the work, the tools are more powerful than they've ever been.

> *AI makes sure that when you show up to do the work, the tools are more powerful than they've ever been. The showing up is still yours.*

A million lines of code. Eight thousand commits. A year learning how to build it right, and six months building it. Sleepless nights and rabbit holes and tester feedback and quiet doubt and sudden breakthroughs.

That's what building looks like. With AI or without it.

The tools changed. The work didn't.

———

WHAT THIS MEANS FOR YOU

The hard parts — decisions, doubt, sustained effort — don't go away with AI. Plan for a learning phase before your real build. Name the cost honestly — to yourself, your team, your family. The wall usually precedes the breakthrough. AI compresses the timeline and amplifies your output, but the showing up is still yours.

In the next chapter, we'll talk about where all of this started — a bad blog post, a secret prototype built in forty hours, and the planning meeting where I showed a room full of leaders something nobody knew I'd been building.

CHAPTER FIVE

The Bad Blog Post That Changed Everything

In the spring of 2023, I opened ChatGPT for the first time and asked it to write a blog post.

The blog post was awful.

It was also the most exciting thing I'd seen in years.

I don't remember the topic. It doesn't matter. What I remember is staring at the screen and feeling something shift. The output was generic, a little stiff, obviously machine-written if you knew what to look for. But it existed. In seconds. From a prompt I typed in plain English. No code, no technical knowledge, no waiting — just a question and an answer.

I sat with that for a while. Then I started asking it harder questions.

FRUSTRATION AS FUEL

At the time, Haley Marketing Group was struggling with service delivery. We had humans in the loop responsible for producing content for clients — blog posts, social media, marketing copy — and the system wasn't working. Deadlines were being missed. Clients were unhappy. The volume of work we'd committed to was outpacing our capacity to deliver it.

I've been in that position before. It's an uncomfortable place to sit — watching something not work, knowing it needs to change, trying to figure out what the change looks like. And I'd learned enough by then to know that my instinct when I'm frustrated with a broken process is usually worth paying attention to.

The question I kept coming back to was simple: why are we doing this with humans when AI can do it?

Not because humans aren't valuable. They are. But the part of the process that was breaking — the high-volume, repeatable content generation — was exactly the kind of work that AI was beginning to handle well. The humans in the loop could be doing something more valuable than producing the fifteenth version of the same blog post format.

> *"Why are we doing this with humans when AI can do it?"*

That question became the seed of everything that followed.

FORTY HOURS IN SECRET

I started building a prototype. I didn't announce it. I didn't ask for permission or run it up the chain of command first. I just started building.

The tool was called Amplify AI at the time — it wasn't yet called RogIQ. It was a mix of CRM functionality and AI-powered content writing. The idea was to give our clients a way to generate marketing content at scale without the bottlenecks we were experiencing on the human delivery side.

AI-assisted development in those early days was a very manual process. Write a little bit of code. Hit a problem. Switch over to ChatGPT to work through the problem. Take the solution back to the code. Write a little more. Hit another problem. Repeat.

It was slow. It was sometimes frustrating. And I had about forty hours in it before I showed it to anyone.

I want to be honest about something: I didn't know if it was going to work. I didn't know if the idea was sound or if the execution was anywhere close to what it needed to be.

Building something in secret for forty hours before showing anyone is a particular kind of vulnerable — you've invested enough that you care about the reaction, but not so much that you can't handle being told to start over.

> *Building something in secret for forty hours before showing anyone is a particular kind of vulnerable.*

But I had enough to show. So I showed it.

THE PLANNING MEETING

I brought the prototype to one of our leadership planning meetings. The reaction in that room was the same reaction I'd had sitting alone at my computer the first time I used ChatGPT. Jaws dropped. Not because what I'd built was perfect — it wasn't — but because of what it represented. Here was a working prototype, built by one person in a few weeks, that addressed a real problem we'd been struggling with. It was tangible. It was demonstrable. And it pointed toward something much larger.

I've been on the other side of that moment — watching someone else demonstrate something that made me realize the world was different than I'd thought. There's a particular quality to it. A kind of quiet recalibration that happens in the room.

This time, I was the one holding the demo.

> *This time, I was the one holding the demo.*

That felt significant. Not because of ego — but because of what it meant about the path I'd been on. The welder who taught himself to build websites. The website builder who taught himself to build software. The software builder who was now standing in front of a leadership team showing them what AI could do for their business.

None of those transitions happened because I had a plan. They happened because I kept asking what was broken and then tried to fix it.

WHAT AMPLIFY AI BECAME

That prototype eventually became RogIQ — Haley Marketing Group's AI-powered marketing platform, built for companies ready to grow smarter with AI. That rough prototype — forty hours of building before anyone knew it existed — became the seed of everything RogIQ is today. It took a lot of work between that first demo and a shippable product. More rebuilds. More planning sessions. More moments of thinking we were almost done and then discovering the next 90%.

But the direction was set from that planning meeting. The organization had seen what was possible. And there was no going back to pretending the old way was good enough.

I think about that moment sometimes when I talk to business owners who are still on the sidelines with AI. Still watching. Still studying. Still waiting to feel ready.

The thing I want to tell them is that readiness is not a feeling you arrive at before you start. It's a feeling you develop by starting. The forty hours I spent building that prototype before anyone knew about it were forty hours of becoming ready. Not ready to have all the answers — ready to be in the conversation.

> *Readiness is not a feeling you arrive at before you start. It's a feeling you develop by starting.*

The bad blog post opened a door. The forty hours walked me through it. The planning meeting made it real for everyone else.

Your version of that sequence is waiting for you. The question is when you're going to start.

———

WHAT THIS MEANS FOR YOU

Frustration with a broken process is a signal, not a complaint — pay attention to it. Build something before asking for permission. Readiness is a feeling you develop by starting, not one you arrive at before you start. The question isn't when you'll feel ready. The question is when you're going to start.

In the next chapter, we'll get into the story that best illustrates what's actually at stake: how my team rebuilt in fifteen hours what the traditional approach had spent nearly two years and a substantial budget failing to deliver — and what that means for your business right now.

CHAPTER SIX

The New Standard

I want to talk about something that most people in the technology industry are thinking but not saying out loud.

The standard has moved. Not slightly. Fundamentally. And the developers, the teams, and the organizations that don't recognize that are going to find out the hard way.

What used to be considered strong output from a development team — a handful of features delivered per sprint, a few complex problems solved per week — is no longer the benchmark. The benchmark has been reset by what AI-assisted development makes possible. And the gap between teams that have internalized that and teams that haven't is widening every month.

That's not a prediction. That's what I'm watching happen in real time.

DEVELOPERS ARE PRODUCERS

I think about developers the same way I think about salespeople.

In sales, there's a simple truth: the right number of activities produces the right output. If you make the right number of calls, send the right number of emails, have the right number of conversations — the results follow. It's not magic. It's volume applied with skill and consistency. A salesperson who makes ten calls a day in a world where their peers are making a hundred isn't working differently. They're underperforming by the new standard, whatever their intentions are.

Development works the same way. A developer is a producer. Their job is to produce output — features, fixes, improvements, solutions. The right activities, applied consistently and with skill, produce the right output. Before AI, that equation had a

ceiling. A developer could only move so fast. There were limits on how much could be written, tested, debugged, and shipped in a given period of time.

> *AI didn't change the equation. It raised the ceiling. And then it raised it again.*

What used to take two or three days now takes an hour. What used to require a team of five can now be executed by a team of two who know how to use their tools. The activities are still required — the planning, the thinking, the problem-solving, the quality review. But the time between activity and output has compressed dramatically.

Which means the expectation has changed. The developer who produces what they produced two years ago isn't holding steady. They're falling behind.

THE SKILL NOBODY IS TALKING ABOUT

Getting good at AI-assisted development is its own skill. It is not a feature of other skills. It is not something you pick up automatically by being a good developer. It requires deliberate practice, genuine curiosity, and a willingness to completely rethink how you approach a problem.

What I see from developers who try AI and give up on it is almost always the same thing: they go in with a simple prompt. No context. No planning. No thought about what the AI needs to know in order to produce what they need. They type *I need X* and they get garbage and they conclude that the tool doesn't work.

The tool works. The prompt doesn't.

> *The tool works. The prompt doesn't.*

Communicating effectively with an AI development tool is a craft. You have to learn how it thinks, what it needs, how to give it context that's specific enough to be useful without

being so prescriptive that you constrain the solution. You have to learn when to let it run and when to pull it back. You have to learn how to recognize when it's heading the wrong direction and course-correct before you're an hour into a dead end.

The developers on my team who have mastered this are producing output that would have been unimaginable two years ago. The ones who haven't — who dip into AI occasionally, who treat it as a shortcut rather than a methodology, who haven't invested the time to actually get good at it — are not going to be a fit for where we're going. That's not a harsh judgment. It's math. When the standard moves and you don't move with it, the gap isn't personal. But it is real.

Now is the time. Not next year, not when the tools mature further, not when the organization mandates it. Now. The developers who are honing this skill today are building an advantage that compounds. The ones who are waiting are watching that advantage grow in someone else's hands.

WHAT A DEV TEAM LOOKS LIKE IN FIVE YEARS

I'll tell you what I believe, because I think it's important and because most people who believe it aren't saying it yet.

The junior developer role, as it has traditionally existed, is gone.

Not diminished. Not restructured. Gone. The tasks that junior developers have historically handled — the routine work, the boilerplate, the straightforward implementations under senior supervision — are being handled by AI agents with a speed and consistency that no junior developer can match. The entry-level rung of the development career ladder has been automated away, and the industry hasn't fully reckoned with what that means yet.

> *The entry-level rung of the development career ladder has been automated away. The industry hasn't fully reckoned with what that means yet.*

What replaces it is something different. Not a developer in the traditional sense. Something closer to an orchestrator. A human sitting at the center of a network of AI agents, directing them, reviewing their output, making the judgment calls that require context and nuance and understanding of what the business actually needs. That human might have three, four, five machines running simultaneously — three, four, five different projects moving forward in parallel — managing agents the way a conductor manages an orchestra.

One person. Multiple agents. Output that would have required a department.

I'm not speculating about a distant future. I'm describing a trajectory that is already visible in the tools we use today. The strides that Cursor and other AI development environments have made between 2025 and the first months of 2026 alone are significant enough that the pace of change is clearly accelerating, not slowing. What we could do six months ago and what we can do today are meaningfully different. What we'll be able to do six months from now will be different again.

> *What we could do six months ago and what we can do today are meaningfully different. What we'll be able to do six months from now will be different again.*

WHAT THIS MEANS FOR BUSINESS OWNERS

If you lead an organization that has a development team, or that works with developers, or that is considering building anything with software — this matters to you directly.

First: evaluate your team not just on what they've delivered, but on how they're working. Are they using AI tools? Are they getting genuinely good at them, or just dabbling? Are

they treating AI as a methodology or as an occasional shortcut? The answers to those questions tell you more about where your team will be in two years than any performance review.

Second: recalibrate your expectations. If your development team is telling you that something will take weeks when you know, from what you've seen elsewhere, that it should take days — that's a conversation worth having. Not an accusation. A conversation about how they're working and whether the tools and methods match the moment.

Third: hire differently. The skills that made a great developer five years ago are not the same skills that make a great developer today. Technical depth still matters. But so does the ability to communicate clearly with AI tools, to plan before building, to think like an orchestrator rather than a line-by-line coder. When you're interviewing developers, ask them how they use AI in their workflow. The answer will tell you everything.

> *When you're interviewing developers, ask them how they use AI in their workflow. The answer will tell you everything.*

And fourth: understand that this is not a threat to good developers. It's an opportunity. The developers who embrace this moment are going to be able to do more, build more, and deliver more value than any developer in history. The ceiling has never been higher. The question is whether they're willing to reach for it.

Two years of building RogIQ — a year of V1, six months of V2, a million lines of code, nearly 8,000 commits, top 10% of Cursor users worldwide — taught me what this new standard actually looks like from the inside. It's demanding. It's relentless. It requires more from everyone, every day.

It also produces things that weren't possible before. And that, in the end, is the point. RogIQ is built around this new standard — not as a tool layered onto old workflows, but as something designed for how work actually gets done now.

———

WHAT THIS MEANS FOR YOU

Evaluate your team on how they are working, not just what they have delivered. AI-assisted development is its own skill — it requires deliberate practice. Recalibrate your expectations for what is possible. When interviewing developers, ask how they use AI in their workflow. The answer will tell you everything.

In the next chapter, we'll bring it all the way home — back to the welder, back to my father's phrase, and to the one thing I want you to do when you close this book.

CHAPTER SEVEN

Fifteen Hours vs. Two Years

Let me tell you about a chatbot.

Not a particularly glamorous piece of technology. Chatbots have been around long enough that they've lost whatever novelty they once had. You've interacted with one at the bottom corner of a website, probably been frustrated by it, and moved on. They're useful when they're built well. They're a waste of everyone's time when they're not.

For nearly two years, one of our internal team members — a skilled, dedicated person working the traditional way — had been focused on building one for Haley Marketing Group. This wasn't a lack of effort. It was close to an 80 or 90 percent focus for the better part of two years. The investment in time and resources was substantial. But the tools available, and the methods required to use them, meant that progress was slow and the finish line kept moving.

My team rebuilt it in fifteen hours.

I'm not telling that story to embarrass anyone. The person who spent two years on that project is talented, and the work they did was real. I'm telling it because the gap between two years and fifteen hours is not a story about who's smarter or more capable. It's a story about what's changed — and what it means for every business owner who has been told that building something like this is slow, expensive, and out of reach. When the tools change that dramatically, the timeline changes with them.

WHAT ACTUALLY HAPPENED

When we decided to rebuild the chatbot ourselves, I didn't sit down with a detailed technical specification and a team of engineers. I sat down with a clear understanding of

what the chatbot needed to do, the AI-assisted development tools my team had gotten very good at using, and the accumulated knowledge from rebuilding RogIQ eight times.

About ten hours of focused development. Another five or so of troubleshooting, testing, and refinement. Fifteen hours total, and we had something ready to deploy. We have a customer on it right now. More are coming.

That's not magic. That's what happens when you combine clarity about the goal, the right tools, and a team that has developed genuine competency through repetition and iteration.

> *That's what happens when you combine clarity about the goal, the right tools, and a team that has developed genuine competency through repetition and iteration.*

But here's the question I know you're sitting with: *That's great for Jeff, but I'm not a developer. I don't have a technical team. What does this have to do with me?*

Everything. Let me explain.

YOU DON'T HAVE TO BUILD IT. YOU HAVE TO LEAD IT.

I want to say something clearly, because I think it's the most important thing in this book:

I am not a developer. At least, I wasn't.

Everything I know about software development, I taught myself. I took a course. I built things badly, rebuilt them, built them again. I figured it out because I had to and because I was willing to. Not because I had a natural gift for it or a background that prepared me.

And the skills that made the difference — the skills that closed the gap between two years and fifteen hours — are not technical skills. They're leadership skills that you already have.

The first one is **clarity.** Knowing precisely what you're building, what it needs to do, and what success looks like. We talked about this in Chapter Two — the planning questions that changed how we built RogIQ. The same principle applies here. Part of what made the two-year effort so difficult was that the scope kept shifting, the requirements kept evolving, and the definition of done was never fully locked down. That's not a people problem. That's a clarity problem — and clarity is something leaders provide.

The second one is **the right people.** You don't have to write the code. You have to find people who can, equip them with the best tools available, give them clear direction, and create an environment where figuring things out is expected and failure is treated as information. That's a culture question, not a technical question. And culture is something business owners build every day.

The third one is **the willingness to start.** Geoff Woods, bestselling author of *The AI-Driven Leader*, notes that 100% of companies believe AI is the future of their business. Only 5% are doing anything about it. The gap between believing something and acting on it is not a knowledge gap. It's a courage gap. And closing it doesn't require a technical background. It requires the same thing every other hard business decision has required: deciding that the discomfort of moving is preferable to the risk of staying still.

> *The gap between believing something and acting on it is not a knowledge gap. It's a courage gap.*

I want to spend a moment on this, because I think a lot of business owners are operating with a distorted picture of what AI is and what it isn't.

AI is not a replacement for human judgment. It is not going to run your business for you, make your strategic decisions, or understand your customers the way you do. Anyone selling you that vision is selling you something you don't need and probably can't use.

What AI actually is, at its most practical: a force multiplier. It takes the work that used to require significant human time and reduces that time dramatically. It takes the knowledge that used to be locked inside specialists and makes it more accessible. It takes the processes that used to require large teams and makes them executable by smaller ones.

My team built in fifteen hours what had taken nearly two years the traditional way. Not because we're smarter. Because we had better tools and we knew how to use them. AI didn't replace my team's judgment — it amplified their ability to execute on it. That speed is not an outlier. It's the direction. And it's exactly the kind of execution RogIQ is designed to enable consistently.

> *AI didn't replace my team's judgment. It amplified their ability to execute on it.*

For a business owner, the question is not "should I use AI?" That question has been answered. The question is "where in my business is human time being spent on work that AI could handle, and what could my people be doing instead?"

Start there. Not with the technology. With the problem.

FIGURING IT OUT AS A CORE VALUE

One of Haley Marketing Group's core values is figuring it out. It's not phrased as elegantly as some corporate values statements — it doesn't have a slogan or a tagline. It's just the expectation that when you run into something you don't know how to do, you don't stop. You figure it out.

I've come to believe that this is the most important cultural ingredient for navigating a period of rapid technological change. Not specific technical skills — those change too fast to be the foundation. Not AI literacy in some abstract sense. Just the deep, organizational commitment to figuring things out when they're hard.

The people on my team who have thrived in this environment are not always the most technically gifted. They're the ones who treat a new problem as an interesting challenge rather than a threat. The ones who look something up they don't know, try something they've never done, ask for help without embarrassment, and iterate until they get it right.

That mentality is learnable. It's also hireable. When you're building or rebuilding a team for an AI-enabled future, the question to ask in every interview isn't "what do you know?" It's "how do you handle what you don't know?"

> *The question to ask in every interview isn't "what do you know?" It's "how do you handle what you don't know?"*

That's the team that will still be standing five years from now. Not because they predicted exactly how the technology would evolve — nobody can do that — but because they're built for the kind of continuous adaptation that this moment demands.

Fifteen hours versus two years. The difference wasn't talent. It wasn't effort. It was the tools, the clarity, and the willingness to build a new way.

———

You don’t have to build it — you have to lead it. Clarity, the right people with the right tools, and the willingness to start are leadership skills you already have. AI is a force multiplier, not a replacement for judgment. Start with the problem, not the technology.

In the final chapter, we’ll close the loop. Back to the welder, back to my dad’s phrase, and forward to the one thing I want you to do when you finish this book.

CLOSING CHAPTER

Figure Out What's Impossible. Then Do It.

I want to take you back to the welding shop one more time.

Not because the story ends there — it doesn't, not by a long shot — but because something about that place has stayed with me in a way I've been trying to put into words for years.

There's a moment in welding, when the arc strikes and the metal starts to flow, where you have to trust your hands completely. You can't hesitate. You can't second-guess. The work requires full commitment to the motion, even when you can't yet see exactly how it's going to turn out. You move, and the piece takes shape, and then you step back and see what you made.

That moment — the commitment before the outcome — is what my father was describing when he said fake it till you make it. Not deception. Not pretending to be something you're not. The willingness to commit to the motion before you can see where it lands.

Everything I've built in my life has started with that moment.

THE PATTERN

Look at the arc of this story and you'll see the same thing happening over and over.

A website company started by someone who had never built a website. A $9,600 sale closed before a single line of code was written. A family business grown to nearly a million dollars in payroll by two cousins who taught themselves everything they needed to know. A Ruby on Rails course taught by someone who'd only finished it months earlier. An acquisition engineered to land exactly where I wanted it to. A prototype built

in forty hours and shown to a leadership team who didn't know it existed. A chatbot rebuilt in fifteen hours that the old way of building couldn't finish in two years.

None of those things were done by someone who had all the answers before they started. Every single one of them began with a decision to move before the path was clear.

> *Every single one of them began with a decision to move before the path was clear.*

That's the pattern. And it's not unique to me. It's the pattern behind every business that has ever been built by someone who shouldn't have been able to build it according to conventional wisdom.

You don't need permission. You don't need a background in technology. You don't need to understand everything before you start.

You need to start.

WHAT MY DAD'S PHRASE REALLY MEANT

I've spent most of this book unpacking a phrase my father said to me so many times it became part of how I think. And I want to be honest about something: I'm not sure my dad meant it as deeply as I've come to take it.

He was a practical man. A businessman. When he said fake it till you make it, he probably meant something fairly straightforward: project confidence, do the work, and trust that competency will follow commitment.

But the more I've lived it, the more I think there's something deeper in it. Because the version of "faking it" that has actually worked for me isn't about projecting false confidence. It's about acting on a belief in a future version of yourself that doesn't fully exist yet.

When I made that first sales call, I was acting as if I were already the kind of person who built websites. When I showed up to that planning meeting with the prototype, I was acting as if I were already the kind of person who led AI product development. When I told the broker to call Haley Marketing Group, I was acting as if I already knew where I belonged.

In each case, the belief came first. The reality followed.

> *The belief came first. The reality followed.*

That is, I think, the truest version of what my father meant. And it's the thing I most want you to carry out of this book.

YOU'RE ALREADY DOING IT

Here's something I want you to consider before you close this book:

Every business owner who has ever hired someone smarter than them — you've already done this.

Every business owner who has ever adopted a new tool, a new platform, a new process they didn't fully understand yet — you've already done this.

Every business owner who has ever entered a new market, taken on a new type of client, said yes to a project they weren't entirely sure they could deliver — you've already done this.

You have been faking it till you make it your entire career. You just may not have called it that.

AI is not a different kind of challenge. It's the same challenge, with different vocabulary and faster timelines. The business owners who navigate it well will be the ones who apply the same instincts that got them this far: move before you're ready, learn from

what breaks, build toward something specific, and trust that the capability follows the commitment.

> *The capability follows the commitment.*

I'm forty years old. I started my career in a welding shop. I've built companies, sold companies, taught adults to code, and led the development of AI products that are changing how an entire industry operates. I don't say that to impress you. I say it to make a point:

If I can figure it out, you can figure it out.

The only question is whether you're willing to commit to the motion before you can see where it lands.

Figure out what's impossible.

Then do it.

———

About the Author

Jeff Gipson is a technology leader and product builder at Haley Marketing Group, the staffing industry's largest marketing firm. He leads development of RogIQ, an AI-powered marketing platform for businesses ready to grow with AI, and serves on the company's leadership team. Before joining Haley Marketing Group, Jeff co-founded Recruiters Websites, a web design and digital marketing firm serving hundreds of staffing companies nationwide, which he grew and sold in 2022. He

is largely self-taught, has over two decades of experience in the staffing industry, and lives in Jackson, Missouri with his wife and five children.

WHAT TO DO NEXT

If you take nothing else from this book, take this: don't try to transform your entire business with AI all at once. That's where people get stuck. Start smaller. Move faster. Learn by doing. Do this in the next seven days. Not next quarter.

Step 1: Pick One Problem. Not ten. Not a full transformation. One. Look for something in your business that is repetitive, time-consuming, and often inconsistent. Content creation. Client communication. Internal documentation. Lead follow-up. Pick the one that's already frustrating you.

Step 2: Define What Good Looks Like. Before you touch a tool, get clear: What should the output actually be? What does done look like? If you can't describe it clearly, you're not ready to build. Clarity is still the skill.

Step 3: Build a Rough Version. Not perfect. Not scalable. Just real. Use whatever tools you have access to — ChatGPT, AI writing tools, automation platforms, internal resources. The goal is not to get it right. The goal is to get something that exists.

Step 4: Put It in Front of Real Use. Don't hesitate. Use it internally. Test it with a small group. Run it alongside your current process. You will immediately see what works, what doesn't, and what you didn't think about.

Step 5: Iterate Faster Than You Are Comfortable With. Build, test, adjust, repeat. Speed matters more than perfection.

Step 6: Then Scale. Only after something works do you scale it — across teams, across clients, across workflows. Not before.

This is exactly the gap we built RogIQ to solve. Not as a replacement for your team, but as a way to accelerate output, remove bottlenecks, and make this kind of iteration

possible at scale. You can do this without RogIQ — we did, at the beginning. But once you understand what's possible, the next step is having the right system to support it.

Pick one problem. Define it. Build something. Then do it again tomorrow.

CONTINUE THE CONVERSATION

If this book gave you something to think about, the best next step is simple:

Start building.

Jeff leads product development for RogIQ at Haley Marketing Group — an AI-powered marketing platform built to help businesses of all kinds grow smarter with AI.

HALEY MARKETING GROUP

haleymarketing.com

ROGIQ

rogiq.ai

CONNECT ON LINKEDIN

linkedin.com/in/ai-innovation-now

"Whatever your hand finds to do, do it with all your might." — Ecclesiastes 9:10

Build Anyway.

"You don't need a background in technology to lead through it. You need the right mentality — and the courage to start."

Jeff Gipson started his career as a welder. He taught himself to build websites, grew a company to nearly a million dollars in payroll, and sold it to the largest marketing firm in the staffing industry — a firm he had spent a decade studying as the benchmark.

Build Anyway is not a technology book. It's a book about leading through technology you don't fully understand — and doing it anyway. The business owners winning with AI right now aren't the most technical. They're just the ones who started.

If you've ever felt like the tools belong to someone else — this book is for you.

Jeff Gipson
Technology Leader & Product Builder
Haley Marketing Group

rogiq.ai • haleymarketing.com