Andy O’Neil

The Process Readiness Checklist

The Questions I Ask Before I Build

For service business owners doing $500K - $2M in revenue
who are tired of automating the wrong things

By Andy O'Neil

Why I Ask These Questions

After 800+ co-building sessions, I've learned where automation projects go wrong.

It's rarely the tool. It's rarely the team. It's usually the process underneath.

(It's always the process underneath. I just say "usually" to be polite.)

People come to me ready to build. They've mapped it out. They've picked the tools. They want to move fast. And sometimes we do.

But first, I ask questions. Six of them. They're not complicated, but they're revealing. Think of them as a metal detector for buried problems. Except instead of finding old coins, we're finding the reasons your last automation attempt made things worse.

These questions are the diagnostic. This is Step 1 of how I work. Most of the value happens here, before we build anything.

If you pass these questions, we move to the build. If you don't, we fix what needs fixing first. Sometimes the answer is "not yet." Sometimes it's "never." Both are valuable answers.

A diagnostic that tells you "don't build this" saves you from spending $5K on something that makes things worse. That's not failure. That's the system working exactly as intended.

Here are the questions.

Question 1: Tell Me About Your Business

The first thing I need to know: Is your business model still forming, or is it solid and set?

This might seem like a strange place to start. You came to talk about automating a process. Why am I asking about your business model?

Because automating a moving target is how you spend $5K building something you'll tear down in 6 months. Ask me how I know. *(Please don't actually ask me. It's painful.)*

If your service offerings are still shifting. If your pricing model changes every quarter. If you're not sure who your ideal client is yet. If you're still figuring out what you actually sell. We stop here.

Not forever. Just for now.

Get the model solid first. Then we talk.

Signs your model is still forming:

You've changed your core offering twice in the last year. You say yes to projects outside your main service because you need the revenue. Your pricing is different for every client. You're not sure which services are profitable and which aren't. You describe your business differently depending on who's asking.

(If you checked all five, congratulations! You're an entrepreneur. Also, we're not building anything yet.)

Signs your model is solid:

You can describe what you do in one sentence. You've said no to work that doesn't fit. Your pricing has been stable for 6+ months. You know which services make money and which don't. New clients look a lot like your best existing clients.

If your model is still forming, that's okay. It's actually good that we're talking now instead of after you've built automation around something that's about to change.

But we're not building yet. Stabilize first. I'll be here when you're ready.

Question 2: Do You Actually Do This Process?

If not, I want to talk to the person who does.

This isn't a slight. I'm not questioning your knowledge of your own business. But I've learned something from watching hundreds of sessions:

The boss doesn't always know what the person doing the work really does.

I've watched owners describe a process in detail, completely confident. Then I watch the person who actually does it. Different steps. Different order. Workarounds the owner didn't know about. Exceptions that happen all the time but never got reported up.

The owner describes how it should work. The employee shows how it actually works. These are often two very different things.

(This is not a commentary on anyone's management skills. It's just how institutional knowledge works. The stuff that lives in people's hands doesn't always make it up the org chart.)

If you're the one doing the process every day, great. Walk me through it.

If someone else does it, I need them in the room. Not because you're wrong. Because the details live in their hands, not in your head.

Who should be in the conversation:

The person who does this task most often. Anyone who handles exceptions when they come up. The person who fixes it when something breaks.

Questions to ask them before we meet:

What's the most annoying part of this process? What do you do that isn't in the official SOP? How often does something unexpected happen? What information do you wish you had sooner?

The gap between "how we think it works" and "how it actually works" is where automation projects go to die. I'd rather find that gap now than after we've built something.

Question 3: Walk Me Through It

Not the overview. The actual steps.

I'm going to ask you to do this process while I watch. Or describe it in enough detail that I can see it happening.

(Yes, this feels weird. Yes, it's like having someone watch you parallel park. Yes, it's necessary.)

I'm listening for three things:

What happens before this process starts?

Every process has a trigger. Something that kicks it off. A form submission. An email. A phone call. A calendar event.

I need to understand what happens upstream. Because sometimes the problem isn't in your process. It's in what feeds into it. Garbage in, garbage out. Or as I like to call it, "garbage in, automated garbage out."

If the process starts with incomplete information, automation won't fix that. It'll just process incomplete information faster.

What happens after it ends?

Where does the output go? Who uses it? What do they do with it?

Sometimes a process works fine in isolation but creates problems downstream. The output format is wrong. The timing is off. Critical information doesn't make it to the next step.

I need to see the whole arc, not just the middle.

How often do you make exceptions or hit edge cases?

This is the big one.

Every process has exceptions. The question is how many and how weird.

If you follow the standard process 95% of the time and handle exceptions 5% of the time, we can probably automate the 95% and flag the 5% for human review.

If you follow the standard process 60% of the time and the other 40% is "it depends," we have a problem. That's not a process. That's a decision tree that lives in your head. (And decision trees that live in your head are notoriously difficult to import into automation tools.)

When exceptions are frequent, I ask two follow-up questions:

Can they be eliminated altogether?

Sometimes exceptions exist because nobody ever cleaned them up. The edge case that happens every week could be eliminated by changing something upstream. A better form. A clearer instruction. A different policy.

Before we systematize exceptions, let's see if we can kill them.

If not, how do we systematize them?

If the exceptions are real and unavoidable, we need to make them handleable. That means documenting them. Creating decision rules. Building branches into the automation that route exceptions appropriately.

This is work. But it's work that needs to happen before automation, not after.

Question 4: Does the Same Person Own It Start to Finish?

And is it done the same way every time?

Ownership matters.

If the same person handles this process from trigger to completion, they see the whole picture. They know what works and what doesn't. They feel the pain of problems they create for themselves downstream.

If ownership changes hands three times, nobody sees the whole picture. Person A creates a problem. Person B works around it. Person C deals with the consequences. Nobody connects the dots.

(This is how companies end up with processes that make everyone miserable and no one can explain why. It's like a game of telephone, except instead of a phrase getting garbled, it's your operational efficiency.)

When I see frequent handoffs, I see places where information gets lost. Where assumptions go unverified. Where "I thought you were handling that" lives.

Signs of ownership problems:

Multiple people touch the process but nobody "owns" it. Different people do it on different days. There's confusion about where one person's responsibility ends and another's begins. The same mistakes keep happening at handoff points.

And is it done the same way every time?

If Person A does it one way, Person B does it another way, and Person C has their own approach, you don't have a process. You have a habit that varies by who's working that day.

Automation requires consistency. The machine does exactly what you tell it, every time. If your humans aren't consistent, the automation will encode one version and break for all the others.

Before we automate, we need to pick one way. Document it. Get everyone doing it the same way. Then automate.

(I know, I know. Getting humans to do things consistently is harder than the automation itself. Welcome to my world.)

Question 5: Is the Information Needed Available When It's Needed?

If not, what are the blockers?

Every process requires inputs. Data. Decisions. Approvals. Files. Context.

When I watch people work, I notice how often they have to stop and go get something. Chase down an answer. Wait for a response. Look up information that should have been provided already.

These pauses are invisible in a process map. But they're where time actually goes.

Automation can't fix missing information. It just chases it faster.

If your team is constantly emailing clients for missing details, automation will just send those emails faster. You'll still be waiting for responses. The bottleneck hasn't moved. You've just made the bottleneck fancier.

We need to solve the information problem first.

Common information blockers:

Intake forms that don't ask the right questions. Previous steps that don't capture what downstream steps need. Decisions that require approval from someone who's hard to reach. Context that lives in someone's head instead of in the system. Historical information that's not connected to current workflows.

For each blocker, I ask:

Can we get this information earlier in the process?

Can we get it automatically instead of manually?

Can we eliminate the need for it altogether?

Can we design around not having it?

Sometimes the fix is simple. Add a field to the intake form. Sometimes it's structural. Change when and how decisions get made.

Either way, we fix it before we automate. Otherwise, we're just building faster chaos.

(Faster chaos sounds like a great band name but a terrible operational strategy.)

Question 6: If We Automated This Tomorrow, Would the Output Be Correct 97%+ of the Time?

Not 80%. Not "mostly." 97%+.

This question forces honesty.

Automation amplifies whatever you feed it. If your current process produces good outputs 97% of the time, automation will produce good outputs 97% of the time, faster.

If your current process produces good outputs 70% of the time, automation will produce bad outputs 30% of the time, faster. And at scale, 30% wrong is a disaster.

(30% wrong is how you end up apologizing to clients in bulk. Don't ask.)

When I ask this question, I'm listening for hesitation.

If you immediately say "yes, absolutely," that's a good sign.

If you pause, or hedge, or say "well, usually," we need to dig in. What's causing the failures? Are they fixable? Are they predictable?

If you're not at 97%, you have three options:

Tighten the process.

Reduce variability. Standardize inputs. Add verification steps. Make the process more reliable before you automate it.

Reduce the exceptions.

Go back to Question 3. Can we eliminate edge cases? Can we systematize the ones we can't eliminate? Fewer exceptions means higher baseline accuracy.

Accept that this one stays manual.

Some processes aren't ready. Some might never be ready. That's okay.

A manual process that works is better than an automated process that fails 30% of the time. Automation isn't always the answer.

(I know. I'm an automation consultant telling you that automation isn't always the answer. This is the quality nuance you came here for.)

If you're not at 97%, I'd rather tell you that now than take your money to build something that creates problems.

(This is the recipe for not building a 5-star reputation.)

What Happens After These Questions

If you pass these questions, we move to Step 2: the co-build.

We build together in live sessions. I don't disappear and come back with something you don't understand. You see how it works. You understand why it works. When we're done, you own it. You can maintain it. You can extend it.

That's the point.

If you don't pass these questions, we fix what needs fixing first.

Maybe your business model needs to stabilize. Maybe we need to get the person who does the work into the conversation. Maybe we need to eliminate exceptions or fix information gaps or standardize how the process runs.

This isn't failure. This is the diagnostic doing its job.

Sometimes the answer is "not yet."

The process isn't ready, but it could be. We fix the gaps, then revisit.

Sometimes the answer is "never."

The process is too variable, too context-dependent, too human. Automation would make it worse. We leave it alone.

Both are valuable answers.

A $500 diagnostic that saves you from a $15,000 mistake is the best money you'll spend.

That's the 1-2 punch. Diagnose first. Build second. Skip Step 1, and Step 2 goes badly wrong.

About Andy O’Neil

My name is Andy O'Neil, and I run Weblytica, a consulting practice that helps small business owners figure out where AI and automation actually make sense for their business.

I've been freelancing since 2012. I've spent a lot of time inside systems that look fine on paper but break down in practice. That taught me something important: most organizations don't have a technology problem. They have a clarity problem. They can't see where their systems are actually breaking down.

How I Work

Most of my work happens in live co-building sessions over Zoom. I've done over 800 hours of these calls, sitting with business owners and walking through their actual workflows in real time. We look at where work piles up. We find where the real bottleneck is. Then we build the fix together.

This isn't consulting where someone hands you a report and walks away. This is working side by side until the problem is solved and you understand exactly what we built.

What Makes Me Different

I was trained as a University of Arkansas Cooperative Extension Service County Educator, which means I taught by doing, not by lecturing. I rarely stood at the front of a classroom, instead I spent time walking or driving on a farm working alongside the rancher. Most days, my “classroom” was a field, my pickup truck or a 4-H club meeting.

That's the approach I bring to automation consulting. I don't give you a framework and wish you luck. I “walk the field” of your business to help diagnose your systems, and then build the solutions you need alongside you until you can do it yourself.

Email me directly at **andy@weblytica.com** or **Book a Clarity Call**