

FREE CHECKLIST

Project Readiness Checklist

12 questions to answer before you hire a developer

WHAT YOU'LL WALK AWAY WITH

- A written one-page brief you can send to any developer or agency
- A clear "version 1" scope, separated from the nice-to-haves
- The gaps you still need to close, listed before they cost you money
- A quick readiness score so you know what kind of first conversation to ask for

How to use it: set aside 30 to 60 minutes with the person who makes final decisions. Answer in plain language. "We don't know yet" is a valid answer, and writing it down is the point: costly surprises in software projects tend to come from decisions nobody made early. Finish with the readiness score at the end.

The 12 questions

Check a box when the statement is true for you today. Write short answers in the boxes. Circle anything you can't answer yet: those are your open questions.

1 What problem are you solving, and for whom?

If you can't state the problem in one or two sentences, a developer can't size the solution. Lead with the pain, not the feature list.

- ☐ I can describe the problem without naming a technology or a feature.
- ☐ I know who feels this problem most (a role, a team, or a customer type).

THE PROBLEM, IN ONE OR TWO SENTENCES

3 What does "done" look like?

Define success as something you can observe after launch, such as "orders are entered once, not three times" or "managers see this week's numbers without asking." This becomes your acceptance criteria.

- ☐ I have 2 to 4 outcomes I could check on a specific date after launch.
- ☐ I know how we measure each one today (even roughly).

OUTCOMES WE'LL CHECK AFTER LAUNCH

2 What happens today?

The current process, including spreadsheets, email threads and workarounds, is your real requirements document. It shows the steps, the exceptions, and the data you already have.

- ☐ I can list the steps of the current process from start to finish.
- ☐ I know which tools are involved (spreadsheets, apps, paper, email).
- ☐ I can name the most common exceptions or "special cases."

CURRENT STEPS, TOOLS AND WORKAROUNDS

4 Who are the users, and how many?

User roles drive permissions, screens and security. The number of users and where they work (desk, phone, field, customers on the public internet) changes the design and the hosting cost.

- ☐ I've listed each type of user and what they need to do.
- ☐ I know roughly how many people are in each group, now and in a year.
- ☐ I know which devices they'll use (desktop, phone, tablet, kiosk).
- ☐ I know whether outside users (customers, vendors) need access.

User type / role	How many	Device(s)	What they need to do

5 What must version 1 do, and what can wait?

A smaller first release ships sooner, costs less, and teaches you what users actually need. Sorting features now is the single best way to control budget.

- ☐ I've separated "must have at launch" from "should have" and "later."
- ☐ Everything in "must have" ties back to the problem in Question 1.

Must have (launch)	Should have (soon after)	Later / maybe

6 What systems and data does it need to touch?

Integrations are where estimates most often go wrong. Each connection needs access, documentation, test data, and someone on your side who can approve it.

- ☐ I've listed every system it must read from or write to (accounting, CRM, email, files, payments, an existing database).
- ☐ I know who owns each system and who can grant access or API keys.
- ☐ I know where the starting data lives and whether it needs cleanup or migration.

System / data source	Read / write	Owner / who grants access	Notes (API? export only? messy data?)

7 Are there security, privacy, or compliance requirements?

Personal data, payments, health or financial records, and customer contracts can all add requirements around hosting, logging, access and data location. Find out now, not during launch week.

- ☐ I know whether we'll store personal information (names, emails, addresses, IDs).
- ☐ I know whether payments, health, financial or children's data is involved.
- ☐ I've checked customer contracts, insurance or industry rules for requirements (data location, audits, retention).
- ☐ I know who signs off on security for us (owner, IT, a client's security team).

SENSITIVE DATA AND REQUIREMENTS WE KNOW ABOUT

8 Who decides, and how much time can they give?

Projects stall when nobody can answer questions or approve work quickly. A good developer will need a few hours a week from a decision-maker who knows the business.

- ☐ One named person can make day-to-day product decisions.
- ☐ That person can commit time every week (reviews, questions, testing).
- ☐ I know who else must approve budget, scope changes, or launch.

Day-to-day decision-maker	
Hours per week they can give	
Final approver(s) for budget / launch	
Who tests before launch	

9 What is your budget range?

A range is not a commitment. It lets a developer propose the right-sized approach instead of guessing. Include ongoing costs, not just the build.

- ☐ I have a range I'm comfortable sharing for the first release.
- ☐ I've thought about ongoing costs: hosting, licenses, support, and future changes.
- ☐ I know whether funding is fixed (one budget) or can be phased (release by release).

First release: range	
Ongoing monthly / yearly comfort level	
Fixed or phased?	

10 What is driving the timeline?

"As soon as possible" isn't a date. A real deadline (a season, a contract, an event, a system being retired) tells the developer what to protect and what to cut if needed.

- ☐ I know whether there's a hard date, and why it's hard.
- ☐ I know what happens if we miss it.
- ☐ I know of busy periods when our team can't test or train.

KEY DATES AND THE REASON BEHIND EACH**11 Who will run it after launch?**

Software needs care after it ships: hosting, security updates, backups, user questions and small changes. Decide who handles that before you build, because it affects the technology choices.

- ☐ I know who will handle user questions and small issues.
- ☐ I know whether we want ongoing support from the developer or plan to take it in-house.
- ☐ Hosting, domains and cloud accounts will be in our company's name.
- ☐ I know how we'll learn about outages or errors.

OUR PLAN FOR SUPPORT, HOSTING AND UPDATES**12 What do you need from the developer relationship?**

Agree on the working relationship up front: how you'll communicate, who owns what, and how changes are handled. These are easy to settle at the start and hard to fix later.

- ☐ I know how often I want updates and in what form (demo, written summary, call).
- ☐ I expect to own the source code and have access to the repository.
- ☐ I want admin access to every account created for the project.
- ☐ I know how I want changes in scope to be quoted and approved.
- ☐ I know whether time zone overlap or on-site visits matter to us.

MUST-HAVES FOR THE WORKING RELATIONSHIP

Your readiness score

Count the questions you could answer with confidence, meaning most boxes checked and something useful written down.

Questions answered	Where you are	What to ask a developer for
10 to 12	Ready for estimates	Send your brief and ask for a proposal with a phased plan, assumptions written out, and a fixed scope for version 1.
6 to 9	Ready for a discovery call	Ask for a short paid or free discovery session to close the open questions before anyone quotes the full build.
0 to 5	Still defining the problem	Start with Questions 1, 2, 5 and 8. A workshop or scoping engagement may be a better first purchase than a build.

MY SCORE: ____ / 12 · QUESTIONS I STILL NEED TO ANSWER:

Before you send your brief

- ☐ Turn your answers into one page: problem, users, version 1 scope, systems, constraints, dates, budget range.
- ☐ Attach examples: screenshots of the current spreadsheet, sample reports, a sketch of a key screen.
- ☐ List your open questions honestly. Good developers will help you answer them.
- ☐ Ask every developer the same questions so you can compare proposals fairly.
- ☐ Ask how they handle change requests, testing, launch and support after launch.

Want a second set of eyes?

Send your completed checklist to support@loowittechnology.com or call (503) 714-5859. We'll tell you plainly whether the project is ready to quote, and whether we're the right fit. More resources at www.loowittechnology.com/resources.

About the author

Nick Andrew builds custom software, Azure setups, and AI automation, and trains teams to use AI, working with clients worldwide.

For your personal or internal team use. Please don't redistribute. © 2026 Loowit Technology Group, LLC · www.loowittechnology.com · support@loowittechnology.com · (503) 714-5859