FREE RESOURCE

Software Pilot Team Checklist

Catch workflow problems before the entire company depends on the system.

Bradley Hankins Consulting

AI WORKFLOW CONSULTING FOR SMALL BUSINESSES

The Seven Traits of a Strong Pilot Tester

Look for people who have most of these, not necessarily all seven:

1. **Does the work daily.** Someone who touches this process every day will notice friction a manager reviewing a demo never will.
2. **Complains constructively.** You want the person who says "this doesn't handle X" — not the person who says nothing and works around it quietly.
3. **Represents an edge case, not just the average case.** The highest-volume user and the most complicated-use-case user are both worth including.
4. **Willing to document what breaks.** Testing is only useful if the friction gets written down, not just remembered and mentioned in passing.
5. **Has influence with peers.** Whoever the team already listens to will shape how the rollout is received — better they're a believer with real experience than a skeptic with none.
6. **Not the most senior person in the room by default.** Seniority doesn't guarantee they still do the hands-on work the system needs to support.
7. **Comfortable saying "this isn't ready."** You need at least one person on the pilot who won't soften bad news to avoid conflict.

Who to Include

A pilot team of 3–6 people usually works better than a team of 1–2 (too narrow a view) or 10+ (too slow to coordinate). Aim to cover:

- The role that will use the system most often day to day
- The role that receives the *handoff* after the primary task is done (production, billing, customer service — whoever inherits the output)
- Someone who manages an exception-heavy account or process, not just a typical one
- A manager or owner who can make the final go/delay/revise call
- IT or whoever owns the tool relationship, if that's a separate person

Testing-Scenario Worksheet

Run each pilot tester through these scenario types and log the result:

Scenario type	Example	Pass / Fail	Notes
Standard case	A typical, everyday transaction start to finish		
High-volume case	Your busiest customer or heaviest use pattern		
Mid-process change	Customer/user changes their mind partway through		
Data entry error	Typo, missing field, wrong format submitted		
Concurrent edit	Two people touch the same record at once		
Handoff point	Work passes from one role/department to another		
System downtime	What happens if the tool is briefly unavailable		
Cancellation/reversal	Something needs to be undone after the fact		

Training Issue vs. System Issue

When something goes wrong in the pilot, don't assume it's the software. Ask:

- **Could a clear one-page instruction have prevented this?** If yes, it's likely a training gap, not a system flaw.
- **Did two testers hit the exact same wall independently?** If yes, that's a strong signal it's the system, not the person.
- **Is the workaround something a new hire could figure out without help?** If not, the process needs to be redesigned, not just explained better.
- **Did it happen once, or does it happen every time this scenario occurs?** One-off friction is worth noting; repeatable friction is worth fixing before launch.

Exception-Testing Prompts

Push the pilot past the obvious cases. Have testers specifically try:

- Submitting the same request twice
- Entering information in the wrong order the form expects
- Testing what happens when required information genuinely isn't available yet

- Having someone outside the pilot team (a real customer, if safe to do so) interact with the system unprompted
- Testing on a slow connection or an older device, if customer-facing
- Asking "what does the next person in line see when I do this?" — not just what the tester sees

Launch-Readiness Questions

Before rolling out beyond the pilot, the team should be able to answer yes to all of these:

- Can every pilot tester complete the core task without asking for help?
- Is there a written answer for what to do when the system is down or wrong?
- Does everyone downstream of the pilot team know the rollout is coming and what changes for them?
- Have at least two real edge cases (not just the standard case) been tested successfully?
- Is there a single owner who can approve fixes if something breaks in week one?
- Is there a defined date to check back in and confirm it's still working as expected?

Final Decision: Go / Delay / Revise

Go — All launch-readiness questions are a clear yes, and no exception-testing scenario surfaced a repeatable failure.

Delay — Testing surfaced fixable issues, but the team needs more time before they're resolved. Set a specific re-test date, not an open-ended "later."

Revise — The pilot revealed a fundamental mismatch between the tool and how work actually happens. Before spending more time on this rollout, revisit whether this is the right tool or the right process for the problem you're solving.

> *This checklist is provided by Bradley Hankins Consulting. Have a rollout that keeps stalling, or a workflow that breaks the moment volume increases?* ***Request a free 30-minute workflow assessment.***