Project Handover Checklist

Starting

☐ As a start, define the exit criteria for the handover. This should be discussed, negotiated and agreed with both parties and make sure higher management knows this. Then write up a checklist of all things needed to achieve the exit criteria and chase it.

Absolutely necessary

☐ Who was working on the project
☐ What they are doing
☐ List of key dates or milestone
☐ Client contacts

Documentation:

☐ Get a short overview of the basic function of the application - what is it meant to achieve. The business case is probably the best document which will already exist.

☐ Then the functional specification. At this point you're not trying to understand any sort of how or technology, just what the app is meant to do. If it's massive, ask them what they key business processes are and focus on those.

☐ Then the high level technical overview. This should include an architecture diagram, required platforms, versions, config and so on. List any questions you have.

☐ Then skim any other useful looking technical documents - certainly a FAQ if there is one, test scripts can be good too as they outline detailed "how to" type scenarios. Maybe it's just me but I find reading technical documents before I've seen the system a waste - it's too academic and they're normally shockingly written. It's certainly an area I'd limit the time I spent on if I didn't feel I was getting a reasonable return for the time I was spending.

Face to face handover items:

☐ Start with a full system demo. Ask questions as they come up, don't let them fob you off with unclear answers - if they can't answer something have it written down and task them with getting the answer.

☐ Now get the code checked out and running on your machines. Do this on at least two machines - one they lead, one you lead. Document the whole process - this is the most important step. If you can't get the code running you're screwed.

☐ Go through the build process. Ensure that you can build the app (including any automated build and unit tests they may have). Note that all unit tests should pass - if they don't or if they say "oh, that one always fails" then they need to fix that before final acceptance.

☐ Go through the install process. Do this at least twice, one they lead, once you lead. Make sure that it's documented.

☐ Now come up with a set of common business functions carried out with the application. Use this to walk the code with them. The code base will be too big to cover the whole thing but make sure you cover a representative sample.

☐ If there is a database or an API do a similar exercise. Come up with some standard data you might need to extract or some basic tasks you might need to carry out using the API and spend some time working through these with them.

☐ Ask them if there's anything they think you should know.

☐ Make sure that any questions you've written down anywhere else are answered.

☐ You may consider it worth going through the bug list (open and closed) - start with the high priority ones and talk through anything particularly worrying looking. Even if they've fixed it it may point at a bit of code which is troublesome.

☐ And finally if the opportunity exists - if there are any outstanding bugs or changes, see if you can pair program a couple.

Minimal criteria for accepting the application

☐ Get the code to compile
☐ Get the code to build (including the database)
☐ Get the application installed

Also try to get the following

☐ Documented anything you picked up on that wasn't covered to your satisfaction
☐ Answered ALL of your questions - a question they won't answer after being asked repeatedly screams of something they're hiding
☐ Get emails and phone of people that worked on the project