WHITE PAPER

Owning your systems after the build

A system you cannot understand, change or move without your partner is not fully yours, however much you paid for it. This paper sets out what any organization should insist on from any technology partner, us included, and gives you a checklist to use before you sign.

01 What ownership actually means

02 Your own accounts and licenses

03 Admin access that does not depend on anyone

04 Documentation in plain English

05 A record of every release

06 A handover you can test

07 No lock-in, by design

08 The partner checklist

For
Anyone choosing a partner or signing a technology contract

Time to read
About 25 minutes

Use it
Before you sign, and at each renewal

A NOTE BEFORE YOU START

The day the project ends is the day ownership is tested.

Most conversations with a technology partner happen before the build: scope, timeline, cost, who does what. Very few happen about what the organization will hold when the build is over. That is the conversation that decides whether you own your system or rent your understanding of it.

I have seen good organizations, with good partners, end a project unable to answer simple questions. Who is the administrator? Where is the account in our name? Why does this field calculate the way it does? What changed last month? Nobody did anything wrong. The questions were simply never asked at the start, so the answers were never written into the work.

This paper is a list of those questions, and what a good answer sounds like. It is written to be used with any partner, any platform, and any size of project. Use it with us too. A partner who welcomes these questions is one you can work with for years. A partner who resists them has told you something useful before you sign.

None of this is about distrust. The best partnerships I know are the ones where the client could leave at any time and chooses not to.

Sunil Dua
Executive Director and Principal Consultant, Kloudserv

The short version

- Every account, license and domain in the organization's name.
- At least two of your own people with full admin access.
- Documentation in plain English, owned by you, kept current.
- A record of every change: what, why, who, when.
- A handover you test by doing, not by reading.
- Standard tools where possible, and a clear way to part.

When to use it

Before you sign, when it costs nothing to ask. Then at each renewal, and before any partner change.

Worth remembering. Ownership is not a clause you add at the end. It is a set of habits a project either has from the first week or struggles to add later.

01 · WHAT OWNERSHIP ACTUALLY MEANS

Owning a system means you can do five things without asking anyone

Legal ownership is usually clear. The contract says the work is yours, the data is yours, and you pay for the licenses. Practical ownership is different, and it is the kind that matters on a Tuesday afternoon when something has gone wrong and your partner is on another project.

A simple test: could your organization do each of these things, on its own, within a working day?

01

Get in

Log in as a full administrator, with your own named account, without asking anyone to reset a password or grant access.

02

Understand it

Answer "how does renewal pricing work?" or "where does this gift total come from?" by reading something you hold, not by sending an email.

03

See what changed

Know what was changed in the last three months, why, and who approved it.

04

Bring in someone else

Hand the system to a new staff member, a new partner or an independent reviewer, and have them productive within weeks, not months.

05

Take your information with you

Export your records, files and history in a standard form if you ever decide to move, without needing a tool or a favor from the partner.

> If the answer to any of these is "we would have to ask our partner", that is the gap. The rest of this paper is about closing it before it opens.

02 · YOUR OWN ACCOUNTS AND LICENSES

Everything in your name, from the first day

It is common, and usually well meant, for a partner to set up a tool under their own account to save time. A form builder, an email service, an integration platform, a test environment, a domain. Months later, part of how your members renew or your donors give runs through an account you do not control, on a card you do not see, under terms you never read.

The rule is simple: every account, license, subscription, domain and environment that your system depends on is opened in the organization's name, with the organization as billing contact, and with a shared organizational email (not one person's) as the owner. The partner is invited in as a user.

Ask before signing

- Which accounts and tools will this project use?
- Will any of them be held by you rather than us? Why?
- Who is the billing contact for each license?

A good answer sounds like

"Everything is opened in your name in week one. We will give you a list of every account, who owns it and when it renews, and keep it current."

03 · ADMIN ACCESS

Two of your own people, with the keys

At least two people inside the organization should hold full administrator access to every critical system, each with their own named login. Not a shared "admin" account whose password lives in someone's notebook, and not access that only the partner can grant. Two, because one is a single point of failure of its own; that risk is the subject of our paper *The one-person risk*.

Integrations and scheduled jobs deserve the same care. They often run under the login of whoever set them up. When that person leaves, or the partner's account is removed, the integration quietly stops. Ask for them to run under a dedicated integration account the organization owns.

Holding admin access does not mean your staff should make changes they are not trained for. It means the organization can always get in, always grant and remove access, and always see what is there.

04 · DOCUMENTATION IN PLAIN ENGLISH

Written for the people who run it, not the people who built it

Technical documentation has its place, and a good partner will produce it. But the document that decides whether you own your system is a different one: a plain-English account of how the system supports your work, readable by a membership manager, a development director or a new executive.

It should answer the questions your staff and board actually ask. How does a member move from applicant to active to lapsed? What happens when a recurring gift fails? Which fields feed the board report, and what does each one mean? Why does a chapter member renew differently from a national one? Each answer should explain the rule and the reason, because the reason is what lets someone decide whether the rule still applies.

WHAT TO ASK FOR	WHY IT MATTERS
A system overview	One or two pages: what each system does, how they connect, and where each kind of record lives.
Process pages	One page per key process (renewals, gift entry, event registration, grant reporting), with the rules and exceptions.
A data dictionary	Every field that matters, what it means, and where its value comes from. This is what keeps board numbers consistent.
Automation and integration list	Every rule, flow, job and connection: what triggers it, what it does, who to ask.
Admin guide	How to add a user, reset access, run the routine tasks and spot the usual problems.

Three conditions that make documentation real

It is yours. It lives in a place the organization owns, in a format you can open and edit without the partner's tools. **It is written during the build, not after.** Documentation left to the final week is always thin, because the final week is always busy. Ask to see it grow at each stage. **It is kept current.** Every change after go-live updates the relevant page. If your support arrangement does not include that, the documentation will drift out of date within a year.

A good answer sounds like: "You will get a plain-English guide to your system, written as we go, stored in your workspace. You can see it at every review, and we update it whenever we change something."

05 · A RECORD OF EVERY RELEASE

If you cannot see what changed, you cannot own what you have

Systems change constantly after go-live. A new membership type, a tweak to the renewal email, a field added for a funder, a fix to the event sync. Each change is small. Over two or three years they add up to a system that nobody fully understands, because the reasons for each change were in a conversation, a ticket or an email that nobody can find.

A release record fixes this. It is a plain list, kept by the partner and owned by the organization, with one line for every change that reaches the live system. It is not a technical change log. It is written so that your operations lead can read it.

DATE	WHAT CHANGED	WHY	APPROVED BY
Example	Renewal notice now goes 60 days before expiry instead of 30	Membership team asked for more lead time	Director of Membership
Example	New "Restricted: capital campaign" fund code added to gift entry	Board approved the campaign	Finance Director
Example	Fixed duplicate contacts created by event registration	Reported by events team	Operations lead

Why the "approved by" column matters

The release record is also how an organization keeps decisions where they belong. Every change to the live system should have someone on your side who said yes. That protects you from changes nobody asked for, and it protects a good partner from being blamed for changes they were asked to make.

Ask for testing before release, too

Changes should be built and checked in a separate test environment before they reach the live system, and someone on your side should see them working first. For small changes this can take minutes. It still matters. The habit is what keeps a quick fix from turning into a lost weekend for your membership team.

A practical sign of a healthy partnership: at any review meeting, you can open the release record, read the last month of changes, and recognize every one of them.

06 · A HANDOVER YOU CAN TEST

Do not accept a handover by reading. Accept it by doing.

Many handovers are a meeting and a folder. The partner walks through the system, shares the documents and the project closes. Everyone nods. A month later, the first real question arrives and the organization discovers what it did not understand.

A handover you can test has acceptance criteria agreed at the start of the project, and is signed off only when your own people have done the work themselves while the partner watches. It might take a day or two of focused time. It is the cheapest insurance in the whole project.

A handover test, in practice

YOUR STAFF SHOULD BE ABLE TO	DONE
Add a new user, set their access, and remove a user who has left	☐
Run the main cycle themselves: a renewal batch, a gift batch, an event registration, a grant report	☐
Produce the board report and explain where each number comes from	☐
Find the documentation for a given process and follow it without help	☐
See which integrations ran overnight, and tell whether any failed	☐
Export a full set of records in a standard format	☐
Say what changed in the last release, and who approved it	☐

Ongoing support is not a substitute

Many organizations keep a partner on after go-live, and that is often wise. But support should be a choice, not a dependency. The test of a good handover is that you could take support in house, or move it to someone else, and the system would keep running while you decided. If the only way to keep the system working is to keep the same partner, ownership has not really transferred.

A good answer sounds like: "We will agree the handover tests with you in the first month. The project is not finished until your team has passed them."

07 · NO LOCK-IN, BY DESIGN

The freedom to leave is what makes it safe to stay

Lock-in is rarely deliberate. It builds up through choices that each made sense at the time: a custom component where a standard feature would have done, a partner's own add-on product installed to save effort, configuration that only makes sense with knowledge the partner holds. None of these is wrong in itself. Together, without anyone noticing, they make it very hard to change course.

You cannot remove every dependency, and you should not try. The aim is to know which ones you have, to choose them on purpose, and to make sure each has a way out.

Standard before custom

Ask the partner to use the platform's standard features wherever they meet the need, and to explain in writing each time they do not. Custom work is sometimes right. It should never be the default.

Know what is installed

If the partner installs any product or component of their own, you should know what it does, what it would take to remove it, and whether you can keep using it if the relationship ends.

Your code and configuration are yours

Anything built for you, including custom code, belongs to you and is kept in a place you can reach, with the documentation needed for another qualified team to maintain it.

Your information is yours

You can export all of your records, files and history at any time, in a standard format, without a fee to the partner or a special tool.

Agree how you would part, while you are still on good terms

It feels odd to discuss the end of a relationship before it begins. It is also the only moment when both sides can discuss it calmly. A short exit section in the agreement should cover the notice period, what the partner hands over (accounts, access, documentation, code, the release record), a reasonable amount of time to help a successor, and confirmation that their access is removed when it ends.

A partner confident in their work will agree to this easily. Most good ones already have it in their standard terms. If yours does not, ask; it is a reasonable request, and the reply tells you a great deal.

08 · COPY THIS

The partner checklist

Use it before you sign, with any partner, on any platform. Ask each question, note the answer, and tick only when the commitment is written into the proposal or the agreement.

QUESTION TO ASK	A GOOD ANSWER INCLUDES	AGREED
ACCOUNTS AND LICENSES		
Will every account, license, domain and environment be in our name?	Yes, from week one, with a shared organizational email as owner	☐
Will you give us a list of every account, its owner and its renewal date?	Yes, kept current through the project and support	☐
Is any part of our system running on an account you hold?	No, or a named exception with a date to move it	☐
ADMIN ACCESS		
Will at least two of our staff have full admin access, with their own logins?	Yes, set up before go-live	☐
Will integrations and scheduled jobs run under an account we own?	Yes, a dedicated integration account, not a person's login	☐
How will your team's access be granted, reviewed and removed?	Named accounts, reviewed with us, removed at the end	☐
DOCUMENTATION		
Will we get plain-English documentation written for our staff?	Yes: overview, process pages, data dictionary, admin guide	☐
When is it written, and where does it live?	During the build, in our workspace, in a format we can edit	☐
Who keeps it current after go-live?	Updated with every change, as part of support	☐
CHANGES AND RELEASES		
Will you keep a release record we can read?	Yes: what, why, who approved, when, in plain English	☐
Are changes tested before they reach the live system?	Yes, in a separate environment, seen by us first	☐
Who on our side approves a change to the live system?	A named person per area, agreed at the start	☐

THE PARTNER CHECKLIST, CONTINUED

QUESTION TO ASK	A GOOD ANSWER INCLUDES	AGREED
HANDOVER		
What will our team be able to do on our own at handover?	A written list of tasks, agreed early in the project	☐
How will we test the handover?	Our staff do each task themselves while you watch	☐
Is sign-off tied to passing those tests?	Yes, the project closes when the tests pass	☐
LOCK-IN AND EXIT		
Will you use standard platform features before custom work?	Yes, and explain in writing each time you do not	☐
Will you install any product or component of your own?	Named up front, with what it does and how to remove it	☐
Who owns code and configuration built for us, and where is it kept?	We do, kept where we can reach it, documented	☐
Can we export all our records, files and history at any time?	Yes, in a standard format, without a fee to you	☐
What happens if we end the relationship?	A notice period, a full handover, help for a successor, access removed	☐
PEOPLE		
Who on your side knows our system, and what if they leave?	More than one person, with our documentation as the shared source	☐
Will you teach our staff, not only do the work for them?	Yes, training is part of each stage, not only the end	☐

How to read the results

You are not looking for a perfect score. You are looking for a partner who answers each question plainly and is willing to write the answer down. A "not yet, but we can" is fine. Vagueness on accounts, admin access or exit is the thing to stop and discuss before you sign.

Already signed? Use it at your next review

Most of these can be put right in an existing relationship. Start with accounts and admin access: they are the quickest to fix and protect you most. Then ask for a release record from now on, and agree a date to bring the documentation up to current.

WHAT TO DO NEXT

Two steps, this month

01 Run the five-part test on what you have today

Can you get in, understand it, see what changed, bring in someone else, and take your information with you? Ask your operations lead to answer each in one line. The gaps tell you where to start.

02 Bring the checklist to your next partner conversation

Whether it is a new project, a renewal or a routine review, share the checklist in advance and walk through it together. Framed as "this is how we work with every partner", it is an easy conversation, and a useful one.

Owning your systems does not mean doing everything yourself. It means that whoever you work with, the organization keeps the keys, the understanding and the choice. That is good for you, and it is good for every partner who wants to earn the next piece of work on merit.

Hold us to it

This is the standard we set for our own work: everything in your name, documented in plain English, so the organization owns its system. If you work with us, use this checklist on us. If you work with someone else, use it on them.

IF YOU ARE PLANNING A CHANGE

The Digital Readiness Blueprint

A vendor-neutral look, over four to six weeks, at the systems you already run: what you own, what depends on whom, and what to fix first. You keep the written plan, and you can take it to any partner.