Service Agreements

Build, send, sign, amend, and bill against the Code-aligned Service Agreement instrument. Settings, templates, AI populate, government fees, blockers and warnings, the signing portal, change requests, decline, amendments, and three ways to start a draft.

Up to date as of v1.29.0

CONTENTS

- Overview
- Important: read before using
- Three ways to start a draft
- Settings overview
- Settings: Templates
- Settings: Agreement Profile
- Settings: Signing and Signatures
- Settings: Client Translation
- Settings: Billing on Signing
- Settings: Customize Articles
- The Builder, top to bottom
- Builder: Clients and Services (the person-centric model)
- AI Populate, end to end
- Builder: the main client's card
- Builder: family members and other people
- Builder: Matter
- Builder: Scope of Services
- Builder: services (multi-service matters)
- Builder: Professional Fee (derived)
- Builder: Payment Schedule
- Builder: Milestones
- Builder: Government Fees and Disbursements
- Builder: Parties and signers
- Builder: Client Instructions
- Builder: Additional Fees and Change Orders
- Builder: Procedural Fairness Letter surcharge
- Builder: Advance Payments and Trust
- Builder: Refund Policy
- Builder: Communications

• The 40+ legal clauses, what each one does
• The validator: blockers versus warnings
• Reviewing the draft: HTML and PDF
• Sending for signature
• The client signing portal
• Initials and signature capture
• Counter-signing (the RCIC)
• What happens when fully_signed is reached
• Change requests (the client asks for revisions)
• Reopen an unsigned agreement to make changes
• Decline to sign (the client refuses)
• Amendments, in depth
• Resending, rotating tokens, and editing emails
• Cancelling an agreement
• Encryption and the audit ledger
• Start from code (Intake Form or Service Proposal)

Overview

The Service Agreement is the platform's keystone module. It generates a CICC-aligned written agreement covering scope, fees, payment schedule, milestones, client obligations, complaint handling, termination, limitation of liability, the procedural-fairness-letter surcharge clause, and roughly 40 other clauses tied to specific CICC Code of Professional Conduct obligations. It supports multi-party signing (client, optional co-counsel, optional third-party payer, optional sponsor, optional designated person), per-party email rotation, change requests with structured client comments, decline-to-sign with negotiable or terminal outcomes, amendments to fully-signed agreements, and post-signing automations: a Bill in Single Bills and a Transfer Room with a pre-provisioned case-folder tree.

> **Premium**: The Service Agreements module is Premium. Basic tenants can see the sidebar entry; the landing page renders a Premium upsell card with a direct path to subscribe. Once subscribed, every feature in this chapter unlocks.

- Three ways to start a draft: Draft a new agreement (blank), Amendment (modify a fully-signed agreement), Start from code (seed from an Intake Form code or an accepted Service Proposal code).
- Six settings cards under Settings: Templates, Agreement Profile, Signing and Signatures, Client Translation, Billing on Signing, and Customize Articles.
- A live validator surfaces blockers (gate Send) and warnings (do not gate Send) as you edit; both have section anchors so you can jump straight to the offending field.
- Every legal text is reviewable in HTML (live preview) and PDF (printable preview with the same chrome the client will receive) before you click Send.

Important: read before using

> **Important**: Service Agreement templates, clause libraries, signing configuration, and supporting tools provided by Investatech Inc. are software conveniences only. They are not legal advice and are not produced, endorsed, or sanctioned by the College of Immigration and Citizenship Consultants (CICC), Immigration, Refugees and Citizenship Canada (IRCC), or any other government body. Investatech and its officers, directors, employees, and contractors are software providers, not your lawyer, paralegal, or immigration consultant on any matter handled through this platform.

You remain solely responsible for reviewing every Service Agreement you generate, for adapting it to the facts of each matter, for compliance with the CICC Code of Professional Conduct, the Immigration and Refugee Protection Act, the College by-laws, and applicable law, and for the legal effect of every clause used. By using these tools you agree to indemnify and hold harmless Investatech Inc., its officers, directors, employees, and contractors from and against any claim, complaint, loss, damage, regulatory action, or expense arising from your use of any template, default value, or generated agreement.

This same notice appears at the top of Settings, in the rendered HTML preview, and on the signed PDF preface. It is intentional. Read it, then proceed.

Three ways to start a draft

The Service Agreements list page carries three primary actions in its top right: Start from code, Amendment, and Draft a new agreement. Each opens a different entry point into the same Builder.

Draft a new agreement

The maroon primary button. Opens a blank draft seeded with your Agreement Profile defaults (tax rate, currency, payment methods, surcharges, refund policy, communications block, retention block, advance-payments block, and your tenant's Practice Defaults). The new agreement has no client name, no matter, no scope, no fees. You pick a template (or write everything by hand), run AI Populate if you have IRCC letters and identity documents to extract from, and fill the remaining fields. The draft autosaves every 800 milliseconds.

Amendment

The middle button. Opens a picker that lists every fully-signed primary agreement (not amendments themselves; you cannot amend an amendment). You select the parent, then select which clauses to amend; the Builder re-opens with only those clauses (plus parties and signatures, which are always auto-included). Amendments cannot touch parties, definitions, or signatures (those are forbidden), and picking any fee-related clause auto-reveals fees, payment schedule, and milestones together so a fee change is never accompanied by an inconsistent schedule.

Start from code

The leftmost button. Opens a small dialog asking for a reference code in one of two formats. An Intake Form code (IF-YYYY-XXXXXX) seeds the Builder from a submitted Intake Form, pre-filling client identity, family members, matter information, and any AI-extracted document data the intake captured. A Service Proposal code (SP-YYYY-XXXXXX) seeds the Builder from a Service Proposal the prospect accepted, pre-filling everything

from the accepted option: client identity, matter framing, scope, professional fee, payment schedule, indicative timeline, government fees, and the service components the proposal carried. In both cases, the seeded SA carries a back-link to the source so the source's detail page shows a Linked Service Agreement card.

> **Note**: Codes are case-insensitive but the format is strict. A typo lands a generic 'we could not find that code' message rather than revealing whether the code exists. You get 30 lookups per minute per user. Use Copy on the source detail page to avoid retyping.

Settings overview

Six configuration cards live under Settings (the cog icon on the Service Agreements list page top right). All six are owner-only or owner-plus-admin depending on the surface. Most defaults flow into every new agreement you draft, and most can be overridden per agreement before finalize. The exception is Billing on Signing, which is a single global switch that applies to every Service Agreement signed by your firm regardless of the route that created it (Active File Review, the standalone module, or any other surface).

1. Templates: your reusable library of saved scope, milestones, fee combinations, and matter-type framing.
2. Agreement Profile: defaults that flow into every Service Agreement you draft. The Builder pre-fills these on each new agreement; you can override per agreement before finalizing.
3. Signing and Signatures: how many sections need per-clause initials, which signature methods clients and your team can use, and where you save your own signature once for counter-signing.
4. Client Translation: which languages clients can see your Service Agreements translated into. The signing portal renders a translation icon next to each clause; hovering shows the translated text. Premium-gated for languages beyond English and Quebec French.
5.

Billing on Signing: a global toggle. When on, a Bill is auto-created and emailed the moment any Service Agreement reaches fully_signed.

6. Customize Articles: hide existing articles, add your own custom articles, or override the body text of unlocked articles. Seven articles stay locked (parties, definitions, engagement, fees, payment schedule, signatures, entire agreement) and cannot be removed.

> **Note**: New cards land in this same hub as the module grows. Bookmark Settings rather than individual settings pages so the hub keeps surfacing what is new.

Settings: Templates

The Templates page shows two distinct sections. The top section is Your Templates: every template your tenant has published, writable by owner and admin. The bottom section is Global Templates: more than 100 curated templates published by Investatech that span study permits, work permits, visitor visas, sponsorship pathways, refugee streams, citizenship grants, business immigration, and provincial nominee programs. Global templates are read-only for tenants. A maroon note above the bottom section reminds you that the curated library is a guideline; the licensee remains responsible for accuracy on each matter.

What a template carries

- Matter code (for example C13, C18, C41, IMP T13), matter category (Study, Work, Visit, Sponsorship, etc.), and matter type label.
- Scope included items and scope excluded items as separate text lists.
- Typical professional fee range with a minimum, maximum, and recommended midpoint.
- Default payment schedule (pre-tax amounts that the Builder rescales when you change the chosen fee).
- Default milestones tied to the payment schedule.

- Optional indicative timeline (governmental processing context, for client expectations).

Saving a draft as a template

Inside the Builder, the right rail carries a Save as template button. It snapshots the current draft's matter, scope, fee range, schedule, and milestones into a new tenant template. Tenant templates are versioned: editing a published template creates a new version row; older versions are retained because in-flight agreements may already reference them. You never lose an applied-template-version snapshot.

Three apply modes

- Fill empty: only blank fields in the current draft are populated. Safe when you want to preserve your edits.
- Replace: overwrites scope, fee, and schedule with the template's values. Use when you picked the wrong template the first time.
- Add component: appends a Service Component to the current draft. Use for multi-service agreements where one engagement covers, say, a spouse's open work permit alongside the principal's permanent residence application.

> **Note**: Tenant templates can be shared with Investatech for inclusion in the global library: a Share button on the template detail page sends it for superadmin review. Accepted templates land in the global library credited to your tenant; rejected submissions return with a reason.

Settings: Agreement Profile

The Agreement Profile is the largest settings card. It sets defaults that flow into every new Service Agreement you draft. Five sub-cards group the fields.

Tax profile

Default tax rate (the percentage shown on the rendered Fees clause and used to compute the post-tax payable on the schedule), default tax label (GST/HST, GST+QST, TVQ, exempt, etc.), and the default tax-included toggle (when on, the typed professional fee is gross of tax, and the Builder will not double-scale). Per-agreement override is a single number in Client Instructions; the rendered text reads the override when present.

Default official language

English or French. The Builder seeds draft.serviceOfficialLanguage with this value on new agreements; you can override per agreement under Client Instructions. The renderer reads the field and produces an English PDF or a Quebec-French PDF accordingly. The portal still lets the client toggle between English and French at display time, but the legal text was rendered server-side at finalize in the language you chose.

Accepted payment methods

Tick the methods you accept: Stripe (online card and bank), e-Transfer, Cheque, Wire transfer, Cash. Each method ticked unlocks a sub-section asking for the routing details: e-Transfer email; cheque payee name and mailing address (pre-fills from company name and address on first tick); wire beneficiary name, beneficiary address, bank name and address, account number, transit/routing number, institution number, SWIFT/BIC, and intermediary fields. Stripe carries an optional surcharge configuration; Wire carries its own optional surcharge configuration. Both surcharges support either a percentage or a flat amount and cover both professional fees and disbursements.

PFL surcharge

Procedural Fairness Letter surcharge. When IRCC issues a PFL under IRPA section 16 or 40 or on inadmissibility grounds and you have to draft a response, this conditional clause adds a named surcharge to the agreement. Configure the default amount and whether the toggle is on by default. The Builder lets you override the amount per agreement; the rendered clause body includes the licensee-caused-issue carveout that exempts errors, omissions, incompetence, professional misconduct, or dishonest acts by you from triggering the surcharge.

Practice defaults

Twelve tenant-stable fields that seed every new draft via fill-empty merge. The fields cover: use of assistants, use of agents, use of third-party professionals, refund policy, complaints handling channel, file continuity language, communications block (preferred channels and expected response times), licensee public register link, code copy availability, retention period for client files, advance payments and trust handling, and invoicing cadence. Each field has a sensible default; tenants tune them once and forget.

Default early-termination admin fee

Optional. A flat dollar amount the client owes when they terminate the engagement before any work has been done. The rendered termination clause references this amount when set; when blank, the clause renders without an admin fee line.

Settings: Signing and Signatures

This card configures how the signing portal behaves for every Service Agreement your firm sends, plus where you save your own signature once for counter-signing.

Signature mode

Five modes are available. Pick one. The mode decides which clauses require per-clause initials on the public sign view.

- None: the client signs at the bottom only. No per-clause initials. Lightest legal weight.
- Slot-1 only: the client (or designate, if signing in lieu) types initials once; the platform stamps every section heading. Other parties do not initial.
- Material-only: only clauses flagged material in the spine require initials. The default. Covers scope, fees, payment schedule, refund policy, complaint handling, termination, limitation of liability, the PFL surcharge if present, and the sponsor joint retainer if present.
- All clauses: every clause except signatures requires initials. Heaviest legal weight.

- Custom: you pick the exact list of clauses that require initials. The Builder shows a checklist of every clause.

> **Important**: Signature mode carries real legal consequences. The platform does not steer you toward a particular choice. A bold amber legal-advisor warning above the radio group reminds you to consult your legal advisor before changing the firm default.

Allowed signature methods

Three checkboxes: Type, Draw, Upload. Type is always available. Draw lets the signer use a touch screen or mouse to draw a signature directly into the portal. Upload lets the signer upload a PNG or JPEG of an existing signature image. You can disable any combination for either the client side, your team side, or both. The signing portal hides the disabled tabs.

Save your signature for counter-signing

Each RCIC and assistant on the team carries an individual saved-signature row on company_members. Pick a method (Type, Draw, Upload), capture or render it, and save. The platform encrypts the resulting PNG under your tenant DEK and stores it in a private bucket. When you counter-sign an agreement from the dashboard, the platform decrypts your saved signature, paints it into the [[SIGN:rcic]] marker on the rendered HTML and the signed PDF, and finalizes. You do not redraw on each counter-signing; you click Confirm countersign once.

> **Note**: If you try to counter-sign an agreement without a saved signature, the platform redirects you to Settings before letting you continue. Save once and you are good for every future counter-signing.

Settings: Client Translation

Premium tenants can configure the languages their signing clients may see Service Agreements translated into. The signing portal renders a small translation icon next to each clause heading; hovering or tapping shows the translated text in a side panel without leaving the page. The legal text stays in the agreement's official language;

the translation is presented alongside as a comprehension aid, accompanied by a disclaimer that the legal effect lives in the official-language version.

The 10 supported languages

- English (always available; no Premium gate)
- French Quebec / fr-CA (always available; no Premium gate)
- Spanish, Mandarin Chinese, Tagalog, Punjabi, Arabic (right-to-left), Hindi, Urdu (right-to-left), Persian / Farsi (right-to-left)

Per-locale name overrides

Proper nouns (your company name, the RCIC's name, the client's name) are often transliterated badly by machine translators. The platform supports per-locale name overrides on both companies and company_members rows. You enter the canonical spelling for each target language once, and the renderer substitutes an ASCII placeholder around Google Translate so the translator never touches proper nouns. The result is that your business name in Arabic, Persian, or Punjabi reads exactly as you typed it, not as the translator's best guess.

> **Premium**: Translation cache: each translated clause is keyed on (clause id, version, target locale) and cached. The first signer in a given language pays the runtime cost; every later signer of the same agreement in the same language reads from the cache.

Settings: Billing on Signing

> **Note**: Billing on Signing is a single global setting. It applies to every Service Agreement your firm signs, regardless of how the agreement was created (Active File Review, the standalone Service Agreements module, or any other route).

When the toggle is on, the moment a Service Agreement reaches fully_signed, the platform auto-creates a Bill in the Single Bills module and emails it to the client. The Bill carries Stripe Checkout for online card and bank payment plus every offline payment method instructions from your Agreement Profile (e-Transfer email, cheque payee and address, wire routing). The Bill row in Single Bills is owned by the agreement; cancelling the agreement cancels the bill if it has not been paid.

Configurable on the same card

- Bill scope: first installment from the payment schedule, or the full professional fee.
- Include government fees by default: when on, the Bill adds every itemized government fee row to the total. Tenants can override per agreement in the Builder.
- Reminder cadence: how many days after the bill is sent the platform sends a polite reminder. Configurable; default 7 days then 14 days.
- Reference prefix: bills auto-created from Service Agreements use the BILL- prefix; the bill reference also carries back to the agreement reference for cross-look-up.
- Optional accountant CC: a tenant-wide email that gets copied on every bill-sent and every bill-paid notification. Configured on a sibling card.

Amendments and auto-billing

Amendments only auto-bill when the amendment touched a fee-related clause (fees, payment schedule, milestones, PFL surcharge, government fees, Stripe surcharge, wire surcharge, additional fees and change orders, advance payments). An amendment that touched only scope language, communications, or termination wording does not trigger a new bill.

Settings: Customize Articles

> **Important**: Customize Articles changes the legal text on every new Service Agreement your firm produces. Talk to your legal advisor before hiding, overriding, or adding clauses. The platform does not vet your customizations.

Three capabilities on one page: hide non-locked articles, override the body text of unlocked articles, and add your own custom articles at one of three insertion points.

Seven locked articles

- Parties (always rendered; carries the legal identity block of every signer)
- Definitions (introduces defined terms used throughout the agreement)
- Engagement (the core authorization for the licensee to act)
- Fees (the dollar amount and breakdown)
- Payment Schedule (the installment timing)
- Signatures (the signature block)
- Entire Agreement (the merger and integration clause)

Hide an article

For any unlocked article, tick Hide. The article is removed from the rendered HTML, the PDF, and the section list. The validator does not run blockers tied to that article. Hidden articles are tenant-wide: every new agreement omits them. Existing agreements are unaffected.

Override a clause body

For any unlocked article, type your own body text in EN and fr-CA. The renderer reads your text instead of the platform's default body when the agreement is rendered. The clause title stays the same; only the body changes.

Save as draft, preview the rendered HTML, then commit. If the override is empty, the platform falls back to the locked default text.

Add a custom article

Three insertion points are available: after the Fees article, before the Signatures article, or at the end of the agreement. Each custom article has a title and a body in EN and fr-CA, an order index relative to other custom articles at the same insertion point, and an optional toggle to make it conditional on a per-agreement flag in the Builder. Custom articles do not get auto-included in amendments unless the amendment specifically picks them.

> **Important**: Customizations apply to every new agreement created after you save the change. Existing agreements (drafts and signed) are unaffected. Save first, test on a fresh draft before sending real client agreements through.

The Builder, top to bottom

The Builder is the long-form editor where you compose every Service Agreement. The layout is a two-column grid: the main edit surface on the left and a sticky right rail on the right. Autosave runs every 800 milliseconds in the background; you never lose work. A status pill at the top of the right rail reads All changes saved, Unsaved changes, Saving, or Save failed depending on the state of the most recent autosave round trip. A Save now button lives next to the pill as a force-shortcut you almost never need.

Right-rail contents

- Reference number (AGMT-YYYY-XXXXXX) auto-generated on first save. Click to copy. Use it to refer to the agreement in client emails, audit notes, or admin tickets.
- Preview HTML: opens the rendered agreement in a side panel exactly as the client will see it in the signing portal (without the signature pills).

- Preview PDF: opens a printable PDF preview with a DRAFT watermark, identical chrome to what the signed PDF will eventually carry.
- AI Populate: opens the document upload modal. See the dedicated AI Populate section below. After an apply, a short green summary next to the button reports how many items landed.
- AI Review: runs a paid Claude pass over the draft and returns observations grouped by severity. Costs 1 Store point per run; never auto-edits the draft.
- Jump-to-section search box: type a few characters and the rail lists every Builder section whose heading matches. Click a result to scroll the main column to that section.
- Filter dropdown: hides Builder sections you do not need on this agreement. The legal text still renders all clauses on output; the filter is a visual aid for the Builder only.
- Send checklist: every blocker and warning the live validator has raised. Each row is clickable; clicking jumps the main column to the offending field. The Send for signature button stays disabled while any blocker is unresolved.
- Delete draft: only available while the agreement is in draft status. Deletes the row and best-effort cleans up any uploaded documents in storage. Cannot delete after Send.

Main column section order

Sections appear in the same order on every new agreement: Templates (apply or save as), Clients and Services, Matter, Professional Fee, PFL Surcharge, Government Fees and Disbursements, Payment Schedule, Milestones, Additional Fees and Change Orders, Refund Policy, Advance Payments and Trust, Communications, Parties (signers and roles), Client Instructions, and Acknowledgements. Clients and Services is where the people on the matter and the work you are doing for each of them both live; it replaces the separate Client, Family Members, Service Components, and Scope sections of the earlier layout. The right-rail Filter dropdown can hide any of

these per session; hidden sections still render in the final HTML and PDF, so they keep their legal weight even when you collapse them while editing.

Note: Section IDs in the URL hash let you deep-link the Builder to a specific section. The Send checklist uses these IDs to jump-scroll; you can also share a Builder URL with /dashboard/agreements/<id>#sa-section-fees to land a teammate directly on the Fees section.

Builder: Clients and Services (the person-centric model)

The Builder is organized around the PEOPLE on the matter. Every person gets a card, and every service you are performing sits as a line on the card of the person it is for. A spousal sponsorship with two dependent children is four cards — the sponsor, the principal applicant, and one card per child — each carrying the services that belong to that person. This replaces the older layout, which had one Client block, a separate Family Members list, and a separate Service Components list you had to link together by hand.

The main card

The first card is the main client and cannot be deleted. It carries the legal identity that the Parties clause and the Section 1 identification block print verbatim: full name, email, phone, and mailing address are all required, and each one raises a blocker while blank. Date of birth, country of birth, country of citizenship, and preferred name are recommended and are what AI Populate fills from a passport bio page. If the client is a business, tick the business-contact box on the main card and the business name, registration number, and business address fields appear there.

Adding other people

Add a card for anyone else in scope, whether or not they sign. Each card carries a relationship to the client (spouse, common-law partner, child, dependent child, parent, sibling, other), a role on the matter (applicant,

sponsor, employer, other), accompanying status, date of birth, and country of citizenship. An agreement holds up to 20 people.

Adding a card does NOT make that person a signing party. Signing is still governed by the Parties block further down the Builder. A person can be fully described here, appear correctly on the rendered agreement, and never be asked for a signature.

Services on a card

Each card holds up to 10 services. A service carries its own name and category, its own included and excluded scope, its own professional fee, and a representation status. Templates apply per service rather than per agreement, so a work permit for the principal applicant and a visitor record for an accompanying parent can each draw scope and fee guidance from their own template. A service that came from a template shows a small chip naming it, and the fee picker offers that template's typical range as a slider so you can position the fee inside it. Services you type by hand carry no chip, which is how you tell at a glance what came from a template and what you wrote.

Fee treatment is what decides whether a service is charged

Every service is set to one of three treatments. Billed separately means the service contributes its fee to the agreement total. Included, no separate charge means the work is covered by another service and is not billed on its own; a note beside it explains the arrangement on the agreement. Pro bono means the work is done without charge. Only services billed separately count toward the total, and the fee box is disabled on the other two so money cannot be typed into a field that will never charge it.

If you type a fee and the total does not move, that service is almost certainly not set to Billed separately. Switch its treatment and the amount becomes editable and starts counting.

Opening an older draft

A draft created before this change is converted to the person-based layout the first time you open it, and a banner at the top says so. The conversion never changes the money: the fee, the payment schedule, and the milestones come across identical, and a leftover bundle adjustment that was inert under the old layout is switched off rather than silently activated. If the converted layout is not what you want, a Revert button on that banner puts the draft back the way it was. Agreements that are already finalized are never touched — they keep rendering exactly as they were signed.

AI Populate, end to end

AI Populate is the fastest path from a stack of client documents to a near-complete draft. The engine is Claude Sonnet 4.6 running through a strictly typed tool schema so the response is structurally guaranteed; the prompt is prompt-cached so the second and later runs in a five-minute window pay only for user-message tokens. The action is Premium-only and counts against your daily AI assistant pool (50 calls per day on Premium).

What to upload

- Passport bio pages and identity documents (driver's licence, government IDs) for client and family members.
- IRCC correspondence: refusal letters, procedural fairness letters, request for additional documents, port of entry refusals, biometric instructions.
- Language test certificates (CELPIP, IELTS, TEF Canada, TCF Canada, PTE Core) for proof of language fields.
- Intake notes typed on the consultation booking page, written consultation tickets, or your own client intake summary.
- Any prior Service Agreement or co-counselling agreement scope or fee language you want the AI to mimic.

Up to ten documents per call. Each document is tagged with a role: client primary, family member with the family member's full name in the row, sponsor, prior agreement, IRCC letter, language test, ID document, or generic context. The role tag is part of the Claude prompt so the extractor knows which fields a given file should inform.

A short tenant prompt lets you add context the documents do not carry (the matter type if it is ambiguous, a fee anchor, any specific scope language you want to preserve, any client preferences).

What the AI returns

- Recommended template: the best-fit global or tenant template, with the recommended chosen fee in cents and a rationale paragraph explaining why. If the documents do not match any template strongly, this field returns null.
- Recommended service components: for multi-service matters (a principal applicant plus a spouse's open work permit, for example), a list of component rows ready to apply with their own scope and fee. Each component has an applicantKind, representationStatus, and feeTreatment.
- Recommended family members: a list of rows (name, relationship to client, date of birth if visible, accompanying status if visible, matter role if inferable). You add them per row with a button.
- Per-field proposals: every scalar field the AI is allowed to propose. Roughly 32 supported paths cover client name, email, phone, date of birth, address fields, country of birth, country of citizenship, business name and registration number for corporate clients, matter type, matter complexity, matter summary, scope included items, scope excluded items, instructions summary, previous attempts, existing representative, official language preference, and party-side details (co-counsel name, CICC ID, fee share, third-party payer name, designated person name and relationship).
- Observations: free-text notes from the extractor about anything unusual it noticed (a discrepancy between two documents, a missing identifier, an ambiguous matter type, a fee figure that read as pro bono but the documents suggested otherwise).

Reviewing and applying

Each proposal renders as a row in the AI Populate modal with a checkbox pre-ticked, the proposed value, a confidence badge (high, medium, low), and the source filename in parentheses. You untick the rows you do not want. A Fill empty radio versus Replace radio at the top of the modal controls whether ticked proposals overwrite existing fields. Array paths (scope.included, scope.excluded, recommendedFamilyMembers, recommendedServiceComponents) always append on apply; they never replace your existing rows. Enum fields and scalar fields respect the Fill empty / Replace toggle.

Click Apply at the bottom. The modal closes, the Builder pushes the new state into your draft, and the validator immediately re-runs. Anything the AI applied that triggers a blocker stays unresolved until you address it; AI Populate is a productivity assist, not a compliance pass. The recommended template Apply button at the bottom of the recommendation card has its own Fill empty / Replace radio so you can apply the template with the AI's chosen fee in one click.

> **Premium**: The recommended-fee field always returns either an explicit dollar amount or null. The system prompt forbids the AI from guessing. A null fee reads as 'no fee mentioned in documents' on the modal; you type the figure yourself or accept the template midpoint. Phrases like 'pro bono' or 'agreed for $500' are correctly distinguished from ambiguous wording thanks to a positive-and-negative examples list in the prompt and a matter-code-priority rule for the recommended template.

Builder: the main client's card

The legal identity of the main client lives on the FIRST card in the Clients and Services section. The rendered Parties clause and the Section 1 identification block read from that card verbatim. Fields are required even if you applied a template; a template seeds matter and scope, not client identity.

Required fields

-

Full name: as it appears on the client's passport bio page. The renderer uses this exact spelling everywhere the agreement names the client. A blocker fires if blank.

- Email: at least one valid email per agreement. Used to invite the client to the signing portal, send the signed copy, and route the bill. Blocker if blank or malformed.
- Phone: used for client-side contact and on the agreement's Communications block. Not invited as a signature channel.
- Mailing address: street, city, province or state, postal code, and country. The Cheque payment method routes a paper cheque to this address; the agreement's Section 1 identification block prints it verbatim.

Recommended fields

- Date of birth. AI Populate fills this from a passport bio page when one is uploaded. Used by templates that want age-conditional scope language.
- Country of birth and country of citizenship. Citizenship accepts multiple values, so a dual citizen carries two. The Information Card reads these for matter context.
- Preferred name: an alternative spelling or chosen name the client prefers in correspondence. Not used in the legal identity block; used in invitation emails and the Communications clause.
- Internal notes: free text that does NOT appear on the rendered agreement. Useful when an assistant picks up the file later.

Business clients

When the client is a business — a corporation hiring an LMIA-exempt worker, for example — tick the business-contact box on the main card. Business name, registration or incorporation number, and business address appear there. The Section 1 identification block then carries the business as the Client and lists the named

individual signing on its behalf in the same block. Most templates handle business clients automatically; a few exclude them.

Note: Everyone else on the matter gets their own card beside this one, and the services you are performing sit on the card of the person they are for. See Clients and Services for the full model.

Builder: family members and other people

Family members are no longer a separate list. Each related person gets their own card in the Clients and Services section, beside the main client's card. Add one card per person who is in scope for this matter even if they are not signing: spouses, common-law partners, dependent children, sponsors, employers, and any other related individual named on the application. An agreement holds up to 20 people.

Per-card fields

- Full name (required).
- Relationship to the client: parent, spouse, common-law partner, child, dependent child, sibling, or other. Other shows a free-text label box.
- Role on the matter: applicant, sponsor, employer, or other. Distinct from relationship. A spouse can be an applicant on a permanent residence application while a spouse on a pure visit visa is other.
- Date of birth. Used by templates that gate scope on whether a child is under or over 22.
- Accompanying status: accompanying or non-accompanying. Drives some templates' default scope language about dependent processing.
- Country of citizenship, accepting multiple values. Used by AI Populate for context.
-

Disclosure authorization: ticking this generates a named entry in the Disclosure Authorization clause of the rendered agreement, explicitly authorizing that person's information to be shared with the licensee and government bodies.

- Custody arrangement, shown only when the relationship is child or dependent child. Use it for shared-custody contexts where IRCC may ask for consent from both parents.
- Internal notes: not rendered on the agreement.

Services belong to the person

The important change is that the work you are doing for each person sits on that person's own card, up to 10 services each. You no longer add a component elsewhere and link it back to a family member by name — the link is the card itself, so a service can never end up attached to the wrong person or to nobody.

Note: Adding a card does NOT make that person a signing party. To require someone's signature, add them on the Parties block below as a co-applicant, sponsor, or designated person. Cards govern inclusion in the matter; the Parties block governs signing.

Builder: Matter

The Matter section frames what the agreement is about. The renderer reads it on the Scope of Services clause heading, the Engagement clause body, the Fees clause body, the signed PDF metadata, and the Bill description if Billing on Signing is enabled. Get it precise; a vague matter description undercuts the rest of the agreement.

Matter type label

A free-text human-readable label. Examples: Study permit application (initial), Open work permit for spouse of skilled worker (C41), Citizenship adult grant under section 5(1), Spousal sponsorship inland with open work permit, PFL response on misrepresentation finding, BC PNP Skilled Worker application. Aim for the level of detail

a client would recognize on a year-end statement. AI Populate fills this from IRCC letter subject lines and from the matter framing in any prior agreement you upload.

Matter code

An IRCC-style code: C13, C18, C20, C41, A71, A74, A77, IMP T13, etc. AI Populate has a matter-code-priority rule that surfaces the most likely code when the documents contain LMIA decisions, IMP exemption letters, or work-permit category mentions. The code does not render on the agreement body; it lives in metadata (used by audit, by reports, by future analytics). Tenant templates carry a default code that auto-fills when you apply the template.

Matter category

A dropdown with twelve categories: Study, Work, Visit, Permanent Residence (Economic), Permanent Residence (Family), Permanent Residence (Refugee), Citizenship, Provincial Nominee, Business Immigration, Inadmissibility and Litigation, Procedural Fairness Response, Other. Templates filter on this in the picker.

Matter complexity

Simple, Standard, Complex. A dropdown. The fee range on a template often varies by complexity; Standard is the default and matches the template's typical midpoint. Complex is where you would expect to anchor higher in the fee range; Simple anchors lower. The validator emits a soft warning if you set Complex but choose the minimum fee, or Simple but choose the maximum, since these often signal a mismatch between scope and price.

Intended outcome summary

A short paragraph naming what success on this matter looks like. Renders inside the Engagement clause: 'The Licensee is engaged to <intended outcome>.' Keep it to one or two sentences. Examples: 'submit a complete C41 spousal open work permit application supported by genuine relationship evidence and pursue its approval to issuance', 'prepare and submit a response to the procedural fairness letter dated <date> on the section 40(1)(a) misrepresentation allegation related to undisclosed prior refusals'.

Builder: Scope of Services

Scope is the most legally-loaded user-typed surface in the agreement. The Scope of Services clause is one of the eight CICC-flagged material clauses; clients initial it under material-only signature mode and the validator runs forbidden-pattern scans across its contents. Two parallel lists live here: scope.included (everything you will do) and scope.excluded (everything you will NOT do under this fee). Both are required for every Service Agreement and both render in full on the agreement body.

How to write scope.included

- One bullet per discrete deliverable. A short clause and an actionable verb: 'Prepare and submit the IMM 5645 sponsorship undertaking', not 'Help with the sponsorship paperwork'.
- Anchor each bullet to a phase or a specific document where possible: 'Draft the response to the procedural fairness letter dated <date>'.
- Include intake, drafting, document review, government correspondence, and follow-up explicitly. If you intend to file an appeal in case of refusal, that lives under Excluded unless you mean to include it for the same fee.
- Templates seed roughly six to twelve included items; tweak them for the matter's specifics before sending. Quick-add chips appear at the bottom of the editor with template-suggested items you have not yet added.

How to write scope.excluded

- Name every common adjacent service that is NOT covered: appeals, judicial review, federal court litigation, study or work permit extensions outside the named application, family member's separate matters, criminal rehabilitation, complex inadmissibility responses, PR Card renewals on the side, employer compliance audits.
- Explicitly exclude the procedural fairness letter response if the PFL surcharge clause is in your agreement; the surcharge clause adds its own fee for that work.
-

Exclude language test booking, document acquisition (police certificates, civil documents from abroad, sworn translations), photo services, biometric appointments. These are the client's responsibility unless you specifically include them.

Forbidden patterns

The validator runs a regex pass over scope.included, scope.excluded, instructions.summary, advancePayments.trustHandling, refundPolicy.earnedFeeBasis, and a few other tenant freetext surfaces. Any of these strings raises a blocker that gates Send: 'non-refundable retainer' / 'non-refundable deposit' / 'non-refundable fee', 'contingency' / 'no-win-no-fee' / 'success-fee', 'withhold the file' / 'file transfer withheld', 'either party may terminate at any time', 'ICCRC' (the regulator changed to CICC in November 2021), blanket disclosure consent, 'Member in Good Standing' without a verified date, guarantee wording on outcome or processing time. The validator surfaces each match with the exact offending substring and the section anchor so you can jump straight to it.

> **Important**: The contingency-prohibited acknowledgement clause is pre-ticked on every new agreement. The validator emits a warning if it is unticked since CICC Code section 24 disallows contingency fees outright.

Builder: services (multi-service matters)

One Service Agreement can cover multiple distinct applications under a single retainer. Common combinations: a principal applicant's permanent residence application alongside a spouse's open work permit (C41), a sponsor's undertaking alongside the sponsored person's permanent residence application, a study permit alongside a co-op work permit. Each of those is a SERVICE, and each service sits on the card of the person it is for in the Clients and Services section. Up to 10 services per person.

Per-service fields

- Service name and category: a short name (Spouse OWP C41, PR principal application, Sponsor undertaking) and the immigration category it falls under.

- Template: apply a global or tenant template to this service. The service line then shows a chip naming the template it came from, and the fee picker offers that template's typical range as a slider. Services typed by hand carry no chip.
- Representation status: represented by the licensee (default), self-representing but disclosure-authorized, or separate signing client (the applicant signs a separate agreement and consents to information sharing with this one). Separate-signing requires recording an authority basis note; the validator raises a narrow blocker when the basis is missing.
- Fee treatment: Billed separately (contributes its fee to the agreement total), Included, no separate charge (covered by other work; renders an explanatory note in the Fees clause), or Pro bono. The fee box is DISABLED on the latter two, with a line beside it naming the setting to change — money can no longer be typed into a field that will never charge it.
- Professional fee: the dollar amount this service contributes when it is billed separately.
- Included and excluded scope: the same editor as before, but scoped to this service. The renderer prints each service's scope under the person it belongs to.
- Service notes: free text for nuance — a deadline that does not apply to the rest of the matter, a document this service depends on.

The fee is derived, not typed

The agreement's professional fee is the sum of every service marked Billed separately, across every person. Change one service's fee and the total, the payment schedule, and the milestones all follow. This is why the older drift blocker and the Sync-from-total button are gone: the total cannot disagree with the itemization any more, because it IS the itemization. A bundle adjustment is still available as a deliberate plus-or-minus line when you want the retainer to differ from the straight sum.

Note: AI Populate returns recommended services and the people they belong to, and applies them onto the right cards automatically. If someone it names is not on the agreement yet, the apply adds their card first. When the 20-person or 10-service limit would be exceeded, the people who did not fit are reported back to you by name rather than silently dropped.

Builder: Professional Fee (derived)

The Professional Fee block carries the legal headline fee that appears in the Fees clause, and is the source from which the payment schedule, milestones, tax, and the auto-generated Bill all derive. The figure itself is no longer typed here: it is the SUM of every service marked Billed separately across every person's card in the Clients and Services section, plus or minus any bundle adjustment. Change a service's fee up there and this block follows, along with the schedule and the milestones. A short note under the amount says where the number came from.

Billing modes

Fixed is the default: the derived sum is the fee. Hourly locks an hourly rate and a budgeted total, and the Fees clause renders the hourly arrangement with a clause about how additional hours are quoted before they accrue; the validator raises hourly_rate_zero when you switch to Hourly and leave the rate blank. Pro bono renders the pro bono framing language, collapses the schedule and milestones to a single zero row, and skips the auto-bill. The contingency prohibition still applies in every mode; pro bono and contingency are not the same thing.

In Fixed mode the validator raises professional_fee_zero when the derived total is zero, which now means every service is either unpriced or not billed separately. Fix it where the money lives, on the service, rather than here.

Client discount

An optional pre-tax deduction line. Type a dollar amount (clamped to be no greater than the professional fee) and an optional reason. The Fees clause renders a Discount line below the fee with the amount and the reason in parentheses. The schedule, the milestones, and the auto-bill all rescale to the post-discount taxable base. This

is CRA-standard treatment: a discount reduces the pre-tax amount, which reduces the GST/HST you collect proportionally.

Bundle adjustment

A deliberate plus-or-minus line for when the retainer should differ from the straight sum of the services — a $500 reduction for taking two matters together, for example. It carries its own label, which renders verbatim in the Fees clause. It is separate from the client discount: the adjustment reflects how the work was priced, the discount reflects a concession to this client.

Tax overrides

Tax rate override (a single number that overrides the tenant default), tax label override (overrides GST/HST or whatever your default says), and the tax-included toggle (when on, the fee is gross of tax and the Builder does not double-scale). Setting the rate to 0 explicitly means no tax and the clause renders without a tax line. Leaving it blank falls back to the tenant default.

Fee notes

A short free-text field that renders inside the Fees clause as a notes paragraph. Use it for arrangements you cannot encode structurally: how the fee was calculated for an unusual matter, why a payment milestone lies outside your standard cadence, why the bundle adjustment is negative.

Builder: Payment Schedule

The Payment Schedule block is a multi-row editor where each row carries an installment label, an optional due date, and a dollar amount. Amounts are PRE-TAX. The renderer scales them at output time so the rendered clause shows the post-tax payable next to each pre-tax amount when an effective tax rate is set. The Fees clause references the schedule by name in its body.

Per-row fields

- Label: a short human-readable name. Examples: 'On signing', 'After biometrics submitted', 'On file submission', 'On AOR receipt', 'On approval-in-principle', 'On final decision'. Drives the line item description on the Bill if auto-billing.
- Due date (optional): a date picker. When set, the schedule clause renders the date inline. Use it sparingly; date-anchored milestones often misalign with the IRCC processing timeline.
- Amount (pre-tax cents): the dollar amount due at this milestone. Money-input pattern (Rule 41).

schedule_mismatch BLOCKER

The sum of every schedule row's pre-tax amount must equal the taxable base. The taxable base is the professional fee plus the PFL surcharge if enabled, minus the client discount if set. A discrepancy raises the schedule_mismatch BLOCKER with the exact dollar shortfall or excess in the message. The fastest fix is the Rescale schedule button next to the schedule editor: it scales every row proportionally so the sum matches exactly. The button cancels the difference into the last row when proportional rescale would leave fractional cents.

Templates seed the schedule

Applying a template populates the schedule from the template's default_payment_schedule, rescaled to the chosen fee. If the template's typical fee is $4,500 and you choose $5,000, the scale factor is computed from chosen/template-sum so the sum of scaled rows exactly equals $5,000. This is why the Sync from total button feels reliable: it follows the same math.

Late payment penalty

An optional configuration under the schedule that adds a late-payment clause to the Fees and Payment Schedule combined clause. Pick a percentage and a grace-period number of days. The clause body explains the consequence in plain language and refers to the schedule. Leaving both fields blank disables the late-payment clause.

Builder: Milestones

Milestones are a parallel structured list that names the work-completion events tied to each installment in the payment schedule. They are optional but strongly recommended: milestones make the agreement legibly accountable to both parties on what triggers what payment, and they form the basis for the Active File Review fee-earned marker on completed matters.

Per-row fields

- Label: a deliverable-anchored name. Examples: 'Initial intake and document review complete', 'Application submitted to IRCC portal', 'Biometrics instruction received and forwarded', 'AOR received', 'Approval-in-principle received', 'Final decision rendered'.
- Amount (pre-tax cents): the dollar amount this milestone earns. The Builder shows a live drift hint if the milestones do not sum to the taxable base, but unlike the schedule, milestone drift does not raise a blocker by default. It is acceptable for milestones to not sum to the taxable base if your refund policy uses time-based earned-fee accounting rather than milestone-based.
- Linked schedule installment (optional): a dropdown that ties this milestone to a specific payment-schedule row. Used by the refund policy and by the auto-bill flow to release the linked installment when the milestone is marked complete.

Milestones in the rendered agreement

The Milestones block renders as its own clause titled 'Milestones' or 'Jalons' depending on language. Each row appears as a bullet with the label and the dollar amount. Rows without a dollar amount render as deliverable-only milestones. Rows linked to a schedule installment add a small parenthetical referring to the installment by label. The clause body opens with a sentence explaining that milestones are the events that mark portions of the work complete and trigger the corresponding installment under the Payment Schedule.

Builder: Government Fees and Disbursements

The Government Fees and Disbursements section is a curated catalogue of every IRCC, CBSA, IRB, and provincial or territorial fee tenants are likely to pay on behalf of clients. The catalogue carries roughly 300 active fee items spanning processing fees, biometric collection, right-of-permanent-residence fees, work-permit issuance fees, study-permit issuance fees, citizenship grant fees, sponsorship undertaking fees, every province's PNP application fees, and inadmissibility-related fees. The catalogue updates when IRCC publishes a new fee or amends an existing one; tenants do not maintain the catalogue.

The breadcrumb picker

Click Add fee. A modal opens with a category tree on the left (Federal -> IRCC -> Permanent residence -> Economic, Federal -> IRCC -> Work permits, Federal -> CBSA -> Inadmissibility, Provincial/Territorial -> Ontario -> PNP, etc.) and a search box on top. The breadcrumb at the top shows your current path; the Home icon jumps back to the root. Click any leaf-level fee to add it to your agreement. The search box runs a word-AND match across fee name and authority: type 'biometric' to find every biometric-related fee, type 'PNP Ontario' to narrow to ON entries. The modal preserves your last-viewed path within the Builder session so you do not re-navigate the tree for every add.

Per-row fields

- Snapshot name: the catalogue fee name as it stood when you added the row. Immutable on the agreement (a future catalogue rename does not retroactively change finalized agreements).
- Snapshot amount (cents): the catalogue dollar amount at add time. Immutable for the same reason. Government fees change. The agreement carries the amount at the time of signing.
- Snapshot authority: IRCC, CBSA, IRB, provincial (Ontario, BC, etc.), or territorial. The renderer groups fees by authority on the rendered clause.
- Snapshot per-unit: per application, per principal, per dependent under 22, per dependent 22 or over, per family. Drives how the quantity multiplier reads ('5 dependents under 22 at $150 each').

- Quantity: how many units. Defaults to 1. Tenant manages this; the Builder does not auto-calculate from Family Members. A row for IRCC right-of-permanent-residence at $515 per principal applicant with quantity 2 means $1,030 total (principal + spouse).
- Override checkbox + override reason + override amount: untick to keep the catalogue snapshot value; tick to override. The Override reason is required when you override and renders inline on the clause. Use the override sparingly: an honest deviation on a specific matter, not a substitute for asking Investatech to update the catalogue.

Custom / provincial / other override

The picker's Use custom / provincial / other escape hatch lets you add a one-off row that is not in the catalogue. Type the fee name (Quebec MIFI processing fee, an ad-hoc translation service, a courier disbursement), the dollar amount, and an optional reason. The row renders identically to a catalogue row but stays labeled 'Custom' in the snapshot authority. Use it for Quebec program fees (MIFI does not publish their fees on the federal portal, so they live as custom rows), translation services, courier services, or one-off disbursements that are part of the matter but not part of the IRCC catalogue.

Work-permit exemption banner

Some work-permit categories are exempt from the standard government processing fee. The catalogue flags those categories; when you browse into work-permit territory, the modal renders a yellow exemption banner reminding you which IMP / LMIA-exempt codes do not pay the IRCC processing fee. The banner also appears per-row when you select an exempted category. Use it as a sanity check; the renderer does not auto-zero the fee, you remove the row yourself.

gov_fees_empty warning

When the Government Fees list is empty AND the matter type is not pure-advisory, the validator emits a soft warning (gov_fees_empty) reminding you that most retainer matters involve at least one IRCC, CBSA, or

provincial fee. The warning does NOT gate Send (matters do exist where you take no fee in advance, for instance pre-engagement consultations). It is a nudge, not a blocker.

> **Note**: Report an inaccuracy: every row carries a small flag icon. Click it to open the Report fee inaccuracy modal. The report lands in the admin console for triage; an Investatech operator amends the catalogue and republishes. The Service Agreements module subscribes to the catalogue at version-id level, so a republish does not retroactively change finalized agreements.

Builder: Parties and signers

The Parties block decides who signs the agreement. Every agreement has at least two signing parties: the Client (or a designate) and the RCIC (the named lead RCIC on the agreement). Optional additional parties are co-counsel, third-party payer, sponsor, and designated person. Each party row gets a portal token at Send and an invitation email; each party signs from their own personal link.

Client signer kind

- Client signs (default): the named client from the Client section is slot 1. Their email is the contact email captured there. They initial on material clauses (when signature mode requires it) and sign at the bottom.
- Designated signatory: a designated person signs in lieu of the client. The Designated Signatory clause renders on the agreement; the designate's name and relationship to the client are required. The designate receives the portal invitation; the client does not. Use when the client is a minor, incapacitated, or genuinely unable to sign (overseas with no portal access, for example).
- Designated representative (non-signing): a designated person is named in the agreement and given communication authority but does NOT sign. The Designated Representative clause renders. The client still signs slot 1. Use for situations where the client wants a family member to receive copies of every communication without becoming a signatory.

Co-counsel

Tick the Co-counsel toggle. Five fields appear: co-counsel name, co-counsel email, co-counsel CICC registration number, fee structure (Percentage or Fixed amount), fee share, and responsibilities (a multi-row list, at least one item required at Send). The Co-counsel clause renders on the agreement naming the second RCIC; the fee-share clause renders the percentage or the fixed dollar amount; the responsibilities list renders verbatim. The co-counsel receives a portal token and an invitation email at Send. Co-counsel must be a CICC member with a valid registration number; the platform does not verify the number, but you should.

Third-party payer

Tick the Third-party payer toggle when the person paying for the work is not the client (a parent paying for a minor's study permit; an employer paying for an LMIA-exempt worker's permit application; a spouse paying for the other spouse's matter). Three fields appear: third-party payer name, email, and relationship to the client (parent, employer, spouse, friend, other). The Third-Party Payer clause renders on the agreement explaining that the payer is not the client and does not direct the matter, that the licensee's duties run to the client only, and that the payer's role is limited to funding. The payer signs (portal token + invitation) to acknowledge that limit. Code section 24(3)(p) requires this disclosure.

Sponsor

Tick the Sponsor toggle when a sponsorship matter requires the sponsor to be on the agreement (family-class sponsorship, refugee sponsorship). Fields: sponsor name, sponsor email, sponsor address, and an Is also a client checkbox. When Is also a client is on, the sponsor is treated as a co-Client and the Parties clause renders an 'and -- The Co-Client (Sponsor):' block; the Sponsor Joint Retainer clause body (also always rendered when a sponsor is present) explicitly names the joint-Client status. When Is also a client is off, the Sponsor Joint Retainer clause body explicitly disclaims joint-Client status ('The Sponsor is NOT a Client of the Licensee under this Agreement'), but the sponsor still signs the agreement to acknowledge the four Code-required disclosures: client status, instruction priority, information sharing, and financial undertaking.

Required signers always sign

Once a party row exists, that party MUST sign before the agreement reaches fully_signed. There are no optional or acknowledgement-only parties. The Builder will let you toggle a party off if you change your mind before Send (the form data is wiped on toggle-off; toggling back on starts from empty). Once you Send, you cannot remove a party; only re-rotate their token (Re-send), update their email (Edit email), or cancel the entire agreement.

Builder: Client Instructions

The Client Instructions block carries everything the agreement should know about how the client has briefed the matter. It is the home of the per-agreement overrides for tax rate, official language, and complexity, and the place to capture client-supplied context that does not fit into Scope.

Fields

- Matter summary: a one-paragraph narrative of why the client came to you and what they want done. Renders inside the Engagement clause body. Distinct from the Intended outcome summary on the Matter section: outcome is what success looks like; summary is the story.
- Previous attempts: a multi-row list. Each row is a free-text description of a prior application, refusal, withdrawal, or attempted permit (with date if known). Renders inside the Engagement clause as a disclosure block. AI Populate fills this from IRCC refusal letters with the refusal date and reason.
- Existing representative: a checkbox plus a free-text note. When ticked, the rendered Engagement clause includes a paragraph naming the prior representative and confirming that the client has terminated that retainer. Use this for cases where the client has fired a previous consultant or lawyer; CICC Code section 24(3)(o) requires the disclosure.
-

Official language (Code 24(3)(s)): a dropdown that overrides the tenant default for THIS agreement. English or French. Drives the rendered PDF language and the official-language version of the legal text. Required by Code section 24(3)(s) to be confirmed with the client before drafting.

- Tax rate override and tax-included override: single fields that override the tenant defaults from Agreement Profile.
- Complexity override: dropdown that overrides the matter complexity set on the Matter section. Use when you want the validator to apply different complexity-based heuristics on this agreement.
- Instructions notes: a free-text field for any additional context the renderer does not consume. Internal use; appears as a sidebar note on the Builder so assistants picking up the file later have the context. Not rendered on the agreement.

Builder: Additional Fees and Change Orders

An optional block that lists itemized additional fees or change-order entries the agreement should disclose at signing. Use it for matter-specific add-ons that do not fit a standard surcharge clause: a translation-of-foreign-civil-documents fee you have already agreed to pass through, an emergency same-day-courier disbursement, an out-of-Canada travel surcharge for in-person tribunal appearance, a sub-contracted forensic-document-review fee.

Per-row fields

- Label: a short name. Required.
- Amount (cents): the dollar amount. Money-input pattern.
- Description (optional): a sentence or two explaining what triggers the fee and how it is billed.
-

Bill at signing? (checkbox): when on, the row is included in the auto-bill amount on Bill on Signing. When off, the row renders on the agreement but the client is billed separately later when the trigger event occurs.

The Additional Fees and Change Orders clause renders below the Professional Fee + Schedule combined clause on the agreement body. The clause body opens with a one-paragraph framing explaining that the items below are itemized add-ons, are not part of the headline professional fee, and become payable when the corresponding triggers occur (or at signing for ticked rows).

Builder: Procedural Fairness Letter surcharge

A conditional clause that adds a named surcharge when the matter ends up requiring a response to a Procedural Fairness Letter. The toggle and default amount come from Agreement Profile; you can override per agreement in this block. When the toggle is on and the amount is positive, the PFL Surcharge clause renders on the agreement; when off or zero, the clause does not render.

When the PFL surcharge applies

The clause body covers PFLs issued under IRPA section 16 (truthfulness / authenticity), section 40 (misrepresentation), or on inadmissibility grounds (criminal, security, health, financial). When IRCC issues a PFL on the matter you are retained on, the surcharge becomes payable for the work of drafting and submitting the response. The clause body includes the licensee-caused-issue carveout: the surcharge does NOT apply when the PFL was triggered by the licensee's error, omission, incompetence, professional misconduct, or dishonest act. This carveout protects the client from being billed for your mistake.

Validator behavior

When the toggle is on but the amount is zero or blank, the validator emits the pfl_amount_missing BLOCKER. Fix by either typing a non-zero amount or turning the toggle off. The PFL surcharge is part of the taxable base when enabled, so the payment schedule recompute includes it; turning the toggle on after you have already set a schedule may require a Rescale schedule pass.

Builder: Advance Payments and Trust

An always-present block that drives the Advance Payments and Trust Handling clause. The clause is REQUIRED on every CICC-aligned agreement because CICC Code sections 24 and 26 govern how licensees handle client funds before they are earned. The Builder defaults to required=true with a generic trust-handling boilerplate that you can edit per agreement.

Fields

- Required toggle: defaults to on. Almost every agreement that takes advance payments needs the clause. The exception is the pure-no-advance-payment matter (rare); turning the toggle off renders the agreement without the clause but the validator emits a warning.
- Trust handling text: a multi-paragraph editor with a sensible default seeded from Agreement Profile -> Practice Defaults. The boilerplate covers: how advance payments are held (in trust until earned), how earnings transfer from trust to general (per milestone, per invoice, per time entry), and what happens to unspent advance payments on termination (refundable per the Refund Policy clause). Forbidden-pattern scan applies (the validator emits a blocker if 'non-refundable retainer' creeps in).
- Non-refundable fee disclosure rules: by default the platform expects every advance payment to be refundable subject to the Refund Policy. If you have a specific lawful non-refundable component (an administrative one-time setup fee, for instance), record it here with a clear basis. The validator's forbidden-pattern scan distinguishes 'non-refundable administrative fee' (acceptable when well-defined) from 'non-refundable retainer' (always a blocker).

Builder: Refund Policy

Required on every agreement. The Refund Policy clause names how the licensee earns the fee, what happens to unearned funds on termination, and the timeline for any refund. CICC Code section 24(3)(j) requires the disclosure; the validator emits a blocker if the block is empty or carries forbidden wording.

Earned-fee basis

A free-text field that anchors how the fee is earned across the lifespan of the matter. Three common patterns: time-based (the fee is earned in proportion to billable hours worked), milestone-based (the fee is earned in proportion to the milestone events completed, with each milestone's dollar amount earning on completion), hybrid (a mix, with explicit rules for what triggers which earning event). Type the pattern you use; the renderer prints it verbatim inside the Refund Policy clause body. Forbidden-pattern scan applies.

Refund timeline

How many days after a termination event the licensee will issue a refund of unearned funds. Default 30 days. Maps to the rendered clause body sentence: 'Any unearned advance payments will be refunded to the Client within X days of termination.' Code section 24(3)(j) requires this disclosure to be specific; 'reasonably promptly' is not enough.

Deduction rules

An optional free-text field naming any lawful deduction the licensee will make from a refund (the early-termination admin fee from Agreement Profile, an out-of-pocket disbursement already incurred, a specific reasonable fee for the work-to-date). The platform does not auto-deduct; this is documentation. The rendered clause body lists the rules verbatim. Use plain language; do not name a percentage if you can name a dollar amount.

Builder: Communications

Drives the rendered Communications clause AND the Electronic Communication clause (a separate clause covering AI, technology, and data-security risks of digital communication). Required by CICC Code section

24(3)(r). The block seeds from Agreement Profile -> Practice Defaults and you override per agreement when the client has unusual needs (a strict no-email channel for residents of a sensitive jurisdiction, for example).

Preferred channels

A multi-checkbox group: Email, WhatsApp, SMS, Phone, In-person, Video conference (Google Meet, Microsoft Teams), Postal mail. Tick every channel the client and the licensee have agreed to use. The rendered Communications clause body enumerates these channels and frames them as the primary means of contact for routine matters.

Expected response times

A free-text field for the response-time framing. Examples: 'Routine messages are answered within two business days; urgent matters within one business day.' Renders verbatim in the clause body. Avoid promises you cannot keep on a sustained basis; the clause sets the client expectation that governs every complaint about response cadence.

After-hours and language of correspondence

Two short fields: After-hours posture (when the licensee is unavailable, and the protocol for genuine emergencies) and Language of correspondence (which language the licensee uses for written correspondence; usually matches the matter's official language but a bilingual client may prefer English for technical correspondence while signing the French agreement). Both render in the Communications clause body.

Electronic Communication clause (v2)

A separate sibling clause titled 'Electronic Communication, Technology, AI, and Data-Security Risks'. Renders by default on every agreement. The body is a 19-item numbered list covering: email is not a confidential channel, attachments can be lost or altered in transit, AI-assisted tools may be used by the licensee for drafting and document review (with redaction of sensitive identifiers where reasonable), cloud-based storage providers have data-residency considerations, the client should never share government-issued identifier numbers (UCI, SIN,

passport) over unsecured channels, and several other technology-related disclosures. Override the body via Customize Articles if your practice has a different standard; the platform-default body is suitable for most practices.

The 40+ legal clauses, what each one does

Every Service Agreement renders against a 44-clause Code-aligned spine. The spine is built so that each clause maps to one or more obligations under CICC Code section 24 (Service Agreements) plus the General By-laws and the IRPA-related obligations. Below is the full list, grouped by where they sit in the agreement, with what each clause does and which ones are unconditional versus conditional.

Identification and engagement (always)

- Parties: legal identity block for every signer. Locked.
- Definitions: introduces defined terms. Locked.
- Engagement: the core authorization for the licensee to act on the matter; references the Client Instructions intended outcome. Locked.
- Scope of Services: lists scope.included and scope.excluded verbatim. Material. Initialled on material-only mode.
- Exclusions: a sibling clause framing what is NOT under the licensee's care. Reinforces scope.excluded. Material.

Fees and money (always or conditional)

- Fees: the headline professional fee, the breakdown (components, discount, tax), and the billing model (fixed, hourly, pro bono). Locked. Material.
- Payment Schedule: the structured installments. Locked. Material.
-

Milestones: the deliverable-anchored events tied to schedule installments. Conditional (renders when milestones list is non-empty).

- Refund Policy: how the fee is earned, refund timeline, deduction rules. Material.
- Government Fees and Disbursements: structured catalogue list + custom rows. Conditional (renders when the gov-fees list is non-empty).
- PFL Surcharge: the conditional surcharge when an IRCC PFL triggers extra work. Conditional.
- Stripe Credit-Card Surcharge: the per-transaction surcharge when Stripe is an accepted payment method and a surcharge is configured. Conditional.
- Wire Transfer Fee: the per-transaction surcharge when Wire is accepted and a surcharge is configured. Conditional.
- Additional Fees and Change Orders: itemized add-ons from the Builder. Conditional.
- Advance Payments and Trust Handling: how advance payments are held and earned. Almost always rendered.
- Chargebacks and Payment Reversals: the client's responsibility if they dispute a Stripe charge after work has begun. Material. Always rendered when Stripe is an accepted method.

Obligations and conduct (always or conditional)

- Client Obligations: the duties of the client (truthful information, timely document provision, prompt response to requests, payment per the schedule, prompt notice of material change). Always rendered. Heavy on Code section 24 alignment.
- Client Delay and Non-Response: what happens when the client fails to meet their duties (the matter goes on hold, additional fees may accrue, the licensee may terminate). Sub-clause within Client Obligations. Always rendered.

- Client Self-Action: the client agrees not to take action on the matter (calling IRCC, submitting documents, contacting border services) without first checking with the licensee. Always rendered.
- Disclosure Authorization: the client's consent to share information with the licensee and named third parties (family members, sponsor, third-party payer). Always rendered. Sponsor and family members named here when they ticked the disclosure-authorization checkbox.
- Sponsor Joint Retainer: governs the sponsor's role and the four Code-required disclosures. Conditional (renders when a sponsor exists).
- Third-Party Payer: governs the payer's limited role per Code section 24(3)(p). Conditional.
- Co-counsel: governs the second RCIC's responsibilities and the fee share. Conditional.
- Designated Signatory: governs the case where someone signs in lieu of the client. Conditional.
- Designated Representative: governs the case where someone has communication authority without signing. Conditional.
- Communications: preferred channels, response times, after-hours, language of correspondence. Always rendered.
- Electronic Communication, Technology, AI, and Data-Security Risks: the 19-item disclosure block. Always rendered.
- Complaint Handling: how the client raises concerns with the licensee first; how they file a complaint with the CICC if unresolved; links to the College's website and a one-sentence framing of the College's role. Always rendered. Material.
- File Retention: how long the licensee keeps client files (default seven years post-closure) and the file-transfer obligation on termination. Always rendered.

- File Continuity: in case of the licensee's death, incapacity, or licence revocation, who takes over the file. Always rendered.

Termination and liability (always)

- Termination: when each party may terminate, the licensee's narrowed grounds (good reason + reasonable notice + no serious prejudice), the 30-day completion duty, and the 10-business-day file-transfer obligation EVEN IF UNPAID. Always rendered. Material.
- Limitation of Liability: caps on the licensee's monetary liability, exclusions for criminal acts and gross negligence, the statute of limitations notice. Always rendered. Material.
- Indemnification: the client's promise to indemnify the licensee for losses caused by misrepresentation, falsified documents, or omitted material facts. Always rendered.
- Governing Law: which province's law governs the agreement and which courts have exclusive jurisdiction over disputes. Always rendered.
- Severability: the standard severability clause. Always rendered.
- Entire Agreement: the merger and integration clause. Locked. Always rendered.
- No Guarantee of Outcome: the licensee does not guarantee outcome or processing time. Always rendered. Material.

Closing (always)

- Acknowledgements: the client acknowledges they have read, understood, and had the opportunity to ask questions about every clause. Always rendered.
- Contingency Prohibited: the client and licensee acknowledge that contingency fees, success fees, and no-win-no-fee arrangements are not permitted under the CICC Code. Pre-ticked. Always rendered.

- Code Copy Availability: the licensee makes a copy of the Code available to the client on request. Always rendered. References the College's website where the Code is published.
- Public Register: the client may verify the licensee's status on the College's public register. Always rendered. References the College's website.
- Signatures: the signature block with one slot per party. Locked.

The validator: blockers versus warnings

The validator runs after every keystroke in the Builder, on every autosave, and at Send time. It surfaces two classes of finding: BLOCKERS gate the Send for signature button (you cannot send until you fix them) and WARNINGS render in amber on the right rail (you can send through them, but you should read them first). The Send checklist on the right rail shows both classes; clicking a row jumps the main column to the offending field via the section anchor.

Common BLOCKERS and how to fix them

- client_name_missing: type the client's full name in the Client section.
- client_email_missing or client_email_invalid: a valid email is required so the platform can invite the client to sign.
- client_address_missing: a mailing address is required for the rendered Section 1 identification block.
- matter_type_missing: type a human-readable matter label in the Matter section.
- scope_missing or scope_too_short: scope.included must have at least one substantive bullet. The validator counts characters; a one-word entry fails the substantive threshold.
- scope_excluded_missing: scope.excluded must also have at least one entry. Empty exclusions are a regulatory risk.

- professional_fee_zero: in Fixed mode, the professional fee must be greater than zero. Switch to Pro bono mode if the matter is genuinely pro bono.
- hourly_rate_zero: in Hourly mode, the hourly rate must be greater than zero.
- schedule_mismatch: the sum of payment-schedule pre-tax amounts does not equal the taxable base. The message tells you the exact dollar shortfall or excess. Click Rescale schedule to fix automatically.
- component_total_drift: priced Service Components plus the bundle adjustment do not sum to the professional fee. Click Sync from total to rescale every priced component proportionally.
- pfl_amount_missing: the PFL surcharge toggle is on but the amount is blank or zero. Type a positive amount or turn the toggle off.
- gov_fee_name_missing: a Government Fees row has no snapshot name. Re-select the catalogue fee, or fill in a Custom name.
- gov_fee_override_reason_missing: a row is marked override but no reason was typed. Type the reason or untick the override checkbox.
- component_separate_signer_unrecorded: a multi-service component is marked Separate signing client but no authority basis note was recorded. Add a note explaining how that component's applicant has authorized information sharing with this engagement.
- co_counsel_responsibilities_missing: the Co-counsel toggle is on but the responsibilities list is empty. Add at least one row describing what the second RCIC will handle.
- co_counsel_fee_share_missing: a fee structure is selected but no percentage or fixed amount is set.
- third_party_payer_relationship_missing: the Third-party payer toggle is on but the relationship to the client is blank.

- designated_signer_relationship_missing: the Designated Signatory or Designated Representative kind is selected but the relationship to the client is blank.
- party_email_duplicate: two parties share the same email. Every signer needs a unique email so portal tokens do not collide.
- refund_policy_basis_missing: the Refund Policy earned-fee basis is empty.
- broad_termination_wording (forbidden-pattern): the agreement contains 'either party may terminate at any time' or similar. The CICC Code narrows the licensee's grounds; this language sneaks in via tenant freetext and the validator catches it.
- contingency_wording (forbidden-pattern): 'contingency', 'no-win-no-fee', 'success fee' anywhere in tenant freetext.
- non_refundable_retainer_wording (forbidden-pattern): 'non-refundable retainer', 'non-refundable deposit', 'non-refundable fee' anywhere.
- guarantee_wording (forbidden-pattern): 'guarantee outcome' or 'guarantee processing time' or 'guaranteed approval' anywhere.
- withhold_the_file (forbidden-pattern): 'withhold the file' or similar; the Code mandates file transfer within 10 business days even if unpaid.
- iccrc_reference (forbidden-pattern): 'ICCRC' anywhere; the regulator changed to CICC in November 2021.
- amendment_no_clauses_selected: an Amendment was opened but no clauses were picked. Pick at least one fee-related or non-locked clause.
- amendment_forbidden_clause: an Amendment includes parties, definitions, or signatures. Those are auto-included; remove them from the pick list.

- open_counter_offer_blocks_send: a Service Proposal counter-offer is open on the related proposal and the Send is paused. Resolve the counter-offer first.

Common WARNINGS and what they nudge you toward

- gov_fees_empty: the Government Fees list is empty. Most retainer matters carry at least one disbursement. Add the IRCC processing fee, biometric fee, or the relevant provincial PNP fee if applicable. Send is NOT blocked.
- scope_too_short_warning: scope.included has fewer than three bullets. A thin scope clause invites misalignment later.
- amendment_reason_empty: an Amendment was opened without a reason note in the metadata. The reason is internal but documents the negotiating context.
- complexity_unset: Matter complexity is left at the default and no override was set. Confirm complexity matches the fee.
- milestones_total_drift: milestones do not sum to the taxable base. Acceptable for time-based fee earning; review nonetheless.
- contingency_acknowledgement_unticked: the contingency-prohibited acknowledgement was unticked. The CICC Code prohibits contingency outright; the warning nudges you to leave it ticked.
- referral_fee_mention: 'referral fee' was found in tenant freetext. Warning only; narrow lawful uses exist but the validator surfaces the match for review.

> **Note**: The Send checklist on the right rail counts the unresolved blockers and warnings. The Send for signature button is disabled while any blocker remains. The validator runs server-side on Send too (defense-in-depth); a client-side validator bug that lets a blocker slip would still be caught on the route handler.

Reviewing the draft: HTML and PDF

Before Send, review the rendered agreement in TWO formats. The right rail carries a Preview HTML button and a Preview PDF button; both render the same draft data through the same renderer (so they cannot drift in content), but they differ in chrome and use case.

HTML preview

Opens a side panel showing the rendered agreement EXACTLY as the client will see it in the signing portal, minus the signature pills. Every clause heading, every body paragraph, every list bullet, every substitution (client name, RCIC name, fee amount, schedule rows, government fees grouped by authority, parties block) is rendered with the same template the production portal uses. The HTML preview is fast to load and the right format for catching wording issues: a typo in a custom clause body, a mis-spelled client name, an inflation of the fee amount, a placeholder token that did not resolve. Scroll through it once end-to-end before Send.

PDF preview

Opens a printable PDF in a new tab with a DRAFT watermark on every page. The PDF chrome is identical to what the signed PDF will eventually carry: tenant logo at the top of page 1, agreement reference and effective date in the top-right of every page, page numbers ('Page X of Y') at the bottom, and the signature block formatted with the right slot count for the parties you have configured. The PDF preview is the right format for catching layout issues: a clause body that runs past the page bottom and pushes the signature block to a lonely last page, a tenant logo that renders mis-sized, a tax line that does not align with the schedule rows.

> **Note**: The DRAFT watermark on the PDF preview is removed only on the final rendered PDF after fully_signed. There is no way to download an un-watermarked draft PDF before signing. This is intentional: a draft PDF sent to a client out-of-band would bypass the audit trail and the portal-token mechanism.

What stays in the rendered output

- Every selected clause from the spine, in the order defined by the renderer (parties first, definitions, engagement, scope and exclusions, fees and money clauses, obligations and conduct, termination and liability, closing).
- Every conditional clause that has its trigger met (PFL surcharge if enabled with a positive amount, Co-counsel if toggled, Sponsor Joint Retainer if sponsor exists, etc.).
- Every Customize Articles override AND addition you have published at Settings level (custom clauses splice in at their configured insertion point; overridden bodies replace the default).
- The legal preface on amendments (three paragraphs: parent identification, scope-limit, conflict clause).
- The AI translation disclaimer block on French-rendered agreements (a one-paragraph notice acknowledging that the French text was reviewed by a human RCIC who reads French).

Sending for signature

When every blocker is resolved, the Send for signature button at the top of the right rail goes live. Clicking it runs the server-side validator one more time, materializes a service_agreement_parties row for every signing party, mints a per-party portal token, persists the rendered HTML and substitution map as a snapshot (so the legal text the client signs is byte-immutable even if the underlying clause library changes later), dispatches an invitation email to every signer, and transitions the agreement's status from draft to awaiting_signature. The Builder reloads in lifecycle view (read-only, with party rows visible as Pending Pending Pending until they sign).

Per-party portal tokens

Each token is a 32-byte CSPRNG value, hashed with SHA-256, and stored as the hash in service_agreement_parties.portal_token_hash. The plaintext token rides in the invitation email's CTA URL only; the platform never persists it. When a signer clicks the link, the portal hashes the URL token and compares against the stored

hash. Tokens are HMAC-bound to a domain prefix (sa-portal-token-v1:) so a leaked token cannot validate against any other module's portal. Tokens rotate on Re-send and on Token rotation triggered by a change-request re-send.

Invitation email contents

From: '<RCIC name>, RCIC via RCIC App' on the RCIC App branded transporter. Reply-To: the company's primary email (so the client's reply lands in your inbox, not Investatech's). Subject: 'Sign your Service Agreement, reference AGMT-YYYY-XXXXXX' (or its French counterpart on French-language agreements). Body: a brief framing paragraph in EN+FR, a Sign your agreement button anchored at the per-party portal URL, the reference number, and a one-sentence reminder that the email is the canonical invitation.

> **Note**: If a signing party's email is wrong, do NOT send a new agreement. The lifecycle view carries an Edit email button next to each pending party row. Click it, type the corrected email, and the platform rotates that party's token and re-sends a fresh invitation. The original invitation's link no longer validates.

The client signing portal

Each party's invitation link lands them on the public Service Agreement portal at /agreement/<token>. The portal is a public surface (no login required) gated by the token in the URL. The page has four panels: the portal header, the rendered agreement body, the per-party action bar at the bottom, and an Acknowledgements block that the signer ticks before submitting.

Portal header

The tenant logo on the left, the agreement reference in the middle, and a language picker on the right. The language picker on the portal lets the signer switch between English and Quebec French (the legal text was rendered server-side at finalize in the agreement's official language, so switching does not change the legal text; it changes the portal chrome and reveals or hides the side-by-side translation panel when one is configured under Client Translation). On Premium tenants with additional Client Translation languages configured, a translation

icon appears next to each clause heading; clicking it opens the translated text in a side panel for the signer's comprehension.

Rendered agreement body

The full agreement text rendered as HTML, scrollable. Each section heading carries an anchor for in-page navigation; the side panel on desktop shows a sticky outline mirroring the section headings. Per-clause initial pills appear inline next to the clause heading on every clause that the signature mode marks as requiring initials. The pills read 'Initial here' before the signer interacts; after the InitialsCaptureModal closes with a saved value, every pill on every required clause fills automatically with the signer's typed initials in their chosen cursive font.

Bottom action bar

The action bar lives at the bottom of the page. It carries the Sign at the bottom signature pill (the slot for THIS party's signature), and four secondary actions: Request changes (opens the structured change-request form), Decline to sign (opens the free-text decline reason form), Save and continue later (no-ops the action and refreshes the page; the signer's already-filled pills persist via the session cookie), and Print (browser print dialog on the rendered HTML, useful when a signer wants a paper read-through before signing). On non-RCIC parties, an additional 'Not ready to sign?' link at the very bottom collapses Request changes + Decline into a less-prominent fall-back.

Acknowledgements gate

Just above the bottom action bar, a checklist of acknowledgements: 'I have read and understood the agreement', 'I had the opportunity to ask questions', 'I confirm the personal information above is accurate', 'I confirm I am the named signer'. Every checkbox must be ticked before Submit will accept. The contingency-prohibited acknowledgement is pre-ticked by default; the signer can untick if they want to flag concern, which triggers a warning when they Submit and asks them to either re-tick or contact the licensee.

Initials and signature capture

Two related modals capture the visual marks that bind each signer to the agreement. The InitialsCaptureModal handles per-clause initials (when signature mode requires them). The SignatureCaptureModal handles the bottom-of-document signature. Both follow a setup-once UX: the first interaction (a click on any initial pill, or a click on the bottom signature pill) opens the relevant modal; saving inside the modal fills every related pill on the page automatically. The signer does not redraw or retype on each pill.

InitialsCaptureModal

Type-only. No Draw or Upload tabs. Two-to-four character initials do not benefit from drawing or upload, and a typed initial in a chosen cursive font is legally sufficient under PIPEDA and the Code's e-signature framework. The modal carries a text input (filtered to A-Z, max 10 characters), a cursive font picker (six fonts to choose from), and a live preview that renders the typed initials in the chosen font at the size that will appear on the agreement. Save closes the modal; the platform stamps every required pill across the agreement body with the saved initials and font.

SignatureCaptureModal

Up to three tabs depending on the firm-configured Allowed signature methods: Type, Draw, Upload.

- Type tab: a text input plus a cursive font picker (twenty fonts to choose from, more than the Initials modal). The signer types their full name or any other glyph they want to use as a signature. A live preview shows the result. The Type tab is the most accessible and works on every device.
- Draw tab: a canvas that accepts touch and mouse input. The signer draws their signature freehand. A Clear button resets the canvas. A live preview shows what the rendered output will look like. Best on touch screens and tablets with a stylus.
- Upload tab: a file picker accepting PNG and JPEG. The signer uploads an image of an existing signature (a scan or photograph of a paper signature). The platform crops and normalizes the upload to a fixed aspect ratio (600x144) and re-encodes as PNG.

How the captured signature ends up on the signed PDF

On submit, the captured value is rasterized to a fixed-size PNG (600x144). For typed signatures, the platform renders the text in the chosen font onto a canvas at the right size and exports the canvas as PNG. For drawn signatures, the canvas IS the PNG. For uploaded signatures, the original image is crop-normalized and re-encoded as PNG. The resulting PNG is encrypted under the tenant DEK and uploaded to a private signatures bucket. The signed PDF render then decrypts each party's saved PNG, paints it into the corresponding [[SIGN:slot]] marker on the rendered text, and re-encrypts the merged PDF under the same DEK.

> **Note**: Signers do not need an account. Every signature flow is anonymous-with-token: the URL token is the capability that proves they are the named party. The session cookie issued by the portal at first interaction binds them to a single party slot for the rest of the session; reloading the page or coming back later from the same browser preserves their already-captured pills.

Counter-signing (the RCIC)

The lead RCIC counter-signs from the dashboard, NOT from a portal token. The lifecycle view on /dashboard/agreements/<id> renders a Confirm countersign button as soon as the agreement enters partially_signed (any non-RCIC party has signed). The button stays visible until the RCIC has counter-signed. If you click before any other party has signed, the platform accepts the counter-sign anyway (RCIC can sign at any time per the parallel-sending design); the agreement just stays in partially_signed until the remaining parties sign.

Pre-flight saved-signature gate

When you click Confirm countersign, the platform first checks that you have a saved signature on your company_members row (set under Settings -> Signing and Signatures). If you do not, the platform redirects you to Settings before letting you continue. Once saved, the platform decrypts your PNG, paints it into the [[SIGN:rcic]] marker on the rendered HTML and the signed PDF, and proceeds.

Inline lifecycle preview

Above the Confirm countersign button, the lifecycle view renders an inline LifecycleSnapshot preview: the full agreement body with every signed party's signature painted into their slot and every initial pill stamped with their typed value. The preview makes it easy to spot a missing or misaligned signature before you commit. Clicking the RCIC's signature pill at the bottom of the preview auto-fills the slot with your saved signature; the Confirm countersign button at the bottom of the preview submits.

> **Note**: Counter-signing is the same act as a client signing: the platform appends a signature_event row to the audit ledger with event_type=countersigned, stamps the time, captures the IP, and progresses the agreement's status. If you are the last signer, the agreement immediately transitions to fully_signed and the post-signing automations fire.

What happens when fully_signed is reached

The single SQL-guarded transition from partially_signed to fully_signed (or from awaiting_signature to fully_signed, when there is only one signing party and they sign directly) runs through one function: tryFinalizeSignedAgreement. The function is idempotent: two concurrent signers on the same agreement cannot trigger duplicate finalizations. The function performs the following work, in order.

1. Render the final HTML and the final PDF. Every [[SIGN:slot]] marker is replaced with the corresponding party's saved signature PNG. Every [[INITIAL:clause_id]] marker is stamped with the corresponding signer's typed initials in their chosen cursive font, rendered at the right size.
2. Append the audit certificate to the last page of the rendered PDF. The certificate carries one row per signer with: signer name, signer email, redacted IP (/24 CIDR), token hash prefix (first 8 hex characters), UTC timestamp of the signature event, signature method (typed, drawn, uploaded), and the agreement reference. The audit certificate is a single page added before the merged-PDF encryption step.

3. Apply owner-password encryption to the merged PDF via @cantoo/pdf-lib. A random 32-byte hex owner password is generated per save and discarded after the encrypt step (the platform never stores it; the goal is to lock down the PDF's permissions, not to require a password to open). Permissions: print + copy + accessibility ON; modify + annotate + fill forms + assembly OFF.

4. Encrypt the merged PDF bytes under the tenant DEK and upload to the agreement-pdfs Supabase bucket at the path agreement-pdfs/<companyId>/<agreementId>/signed-<timestamp>.pdf.enc. The bucket is private; service-role-only reads. The agreement row stamps signed_pdf_path and signed_pdf_encrypted=true.

5. Email the signed PDF as an attachment to the named client (the in-memory PDF buffer, plaintext for the email transport; only the bucket copy is encrypted at rest). From: '<RCIC name>, RCIC via RCIC App'. Subject: 'Your signed Service Agreement, reference AGMT-YYYY-XXXXXX' or French equivalent. CC: every configured sa_signed_cc_emails address from the tenant-wide CC list, plus the tenant accountant if configured, plus the company_email as a record.

6. Trigger the Bill on Signing automation when the global toggle is on: auto-create a Bill in the Single Bills module, email it to the client, route through Stripe Connect, and configure reminder cadence per the Bill on Signing settings.

7. Trigger the Transfer Room activation when the matter is eligible (the SA family qualifies for a Transfer Room): auto-provision the case-folder tree in your connected Drive, materialize the Transfer Room with default tenant-side and client-side notification slots, send the Transfer Room activation email.

8. Trigger Active File Review's advanceAfrOnSaSigned hook when this SA was generated FROM an AFR (the SA row carries source_kind='active_file_review' and source_id pointing to the AFR row). The hook advances the AFR's status from sa_generated to sa_signed.

9.

Stamp the audit ledger with the final signature_event row (event_type=fully_signed) and update the agreement row's audit_cert_appended_at timestamp.

> **Important**: If the render-or-upload step fails, tryFinalizeSignedAgreement rolls the agreement status back to partially_signed for retry. The signature events are NOT rolled back (they are immutable per the audit-ledger trigger); the agreement just remains in partially_signed until a successful retry. Tenants whose first attempt failed see an Amber 'Finalization failed; try again' banner on the lifecycle view with a manual Retry button.

Change requests (the client asks for revisions)

On the public sign view, instead of signing, the client may click Request changes. A structured form opens: the client picks one or more clauses they have concerns about, types a comment per clause, and submits. The agreement transitions to changes_requested. Every other party's signature is preserved; their pending invitations stay live for as long as the request is pending. A partial unique index allows only ONE pending change request per agreement at a time; if a second client tries to file a request while the first is pending, the platform returns a clean 409 'another party has a request pending'.

What the tenant sees

The lifecycle view mounts a ChangeRequestHistoryCard at the top of the agreement detail page. The card lists every change request on the agreement (pending and past) with the requester's name, date, clauses requested, and full text of the comments. The pending request renders in a blue banner with three action buttons: Keep original, Reopen to draft, Amend after negotiation.

Three decisions on a pending request

- Keep original: rejects the request. The agreement returns to awaiting_signature or partially_signed (whichever state it was in before the request). The platform emits a 'we received your concerns but the agreement stands'

email to the requester (with your typed response framing). Every other party's existing invitation continues to work. Use when the client's concerns do not move you legally.

- Reopen to draft: returns the agreement to the draft status. Every existing signature is invalidated (the signed_at and signed_typed_name and signature_event_id stamps clear on every party row; encrypted signature images stay in storage referenced by the audit ledger, which keeps the legal record). The Builder reopens; you edit, re-finalize, and the platform mints fresh tokens and re-invites every party. Use when the client's concerns require structural changes to the legal text.
- Amend after negotiation: a tenant-only action that opens a separate flow. The platform issues a Personal-Invites meeting link tied to this party (15, 30, or 60 minutes; you pick the duration) so you can have a conversation. After the meeting, you come back and re-send the agreement as-is (no edits) or you reopen-to-draft and edit. The amend-after-negotiation status keeps the request visible while you negotiate; it does not invalidate signatures until you choose to.

> **Note**: The change-request history is always surfaced. Even after you Keep original or after the requester has come around and signed, the past requests stay visible on the lifecycle view as part of the audit story. They are part of the negotiating record.

Reopen an unsigned agreement to make changes

When a Service Agreement is in **Awaiting signature** or **Partially signed** status and you discover something needs to change — a wrong address, a missed fee component, a clause that needs to be revised — you no longer need to cancel and start over. Open the agreement detail page, look in the right-rail action card, and click **Reopen to make changes**. This is a sibling action to Cancel agreement and appears in the same place. A short prompt asks you to confirm, then a second prompt asks you to type a brief summary (up to 500 characters) of

what's changing. The summary is required — it will be shown verbatim to the client in the next invitation email so they know what to look for when they re-review.

Once you confirm, the system flips the agreement back to **Draft** status, clears every party's signature (including your own RCIC pre-signature if you had already countersigned), records a **changes_reopened_to_draft** event in the audit ledger with the summary attached for the record, and lands you in the Builder so you can edit immediately. Make the corrections you need, click Re-finalize when you're done, and the system sends fresh invitations to every party. The email subject changes from **Service Agreement ready to sign** to **Updated Service Agreement ready to sign**, and a maroon callout above the Review and sign button renders your summary along with a note that any signatures from the previous version have been cleared.

Client portal links rotate on re-finalize, so the link from the previous invitation no longer opens the signing surface. If the client clicks the old link by mistake (they're easy to miss in a busy inbox), they don't see a 404 — they see a friendly page that reads **This link has been replaced** along with the agreement reference, telling them the consultant updated the agreement and a new signing link is in their email. They open the new email, click the new link, and re-review the updated document with the change summary still visible in the maroon callout.

> **Reopen vs. Amendment.** Reopen-for-changes is only available before the agreement is fully signed. Once every party has signed and the document is in Fully signed status, you use the Amendment flow instead (see the Amendments in depth section). The rule of thumb: if any party other than you (the RCIC) has signed, the right tool is Amendment — it preserves the original signed instrument and creates a new -A1 amendment as its own auditable document. If only you have pre-signed, or nobody has signed yet, Reopen is the right tool — it edits the same agreement in place and is the faster path for routine corrections like a wrong address or a typo.

Decline to sign (the client refuses)

Decline to sign is a different and stronger signal than a change request. A client who declines is rejecting the agreement, not asking for revisions. The decline form opens from the bottom action bar's Decline to sign link: a free-text reason field plus a dropdown picking whether the decline is intended as negotiable (the client wants to discuss further) or terminal (the client is done; do not contact). On submit, the agreement transitions to declined_negotiable or declined_terminal depending on the picker; every other party's pending invitation stays live but the declining party's slot is permanently locked.

declined_negotiable

The client signaled they may still come around if you can address their concerns. The lifecycle view shows the decline in an orange banner with three tenant actions: Invite to meeting (issues a Personal-Invites meeting link tied to this party for 15, 30, or 60 minutes; the platform stamps meeting_invited_at and on booking redemption stamps meeting_accepted_at), Re-send as-is (rotates the declining party's token and re-sends a fresh invitation; useful when the meeting clarified things and the client agrees to sign the original), or Amend in builder (reopens the agreement as a draft, edits enabled, every signature invalidated). End decline by tenant is the fourth option (a soft close that moves the status to terminal without further attempts).

declined_terminal

The client signaled they are done. The lifecycle view shows the decline in a charcoal banner. The only tenant action available is Cancel agreement (closes the agreement out cleanly; the audit ledger preserves every signature event, every change request, every decline reason in plaintext for legal defensibility). Re-send and Amend are NOT available; respect the client's signal.

> **Note**: Personal-Invites meetings on declined_negotiable are internal-only services (is_internal_only=true on the underlying services row), so they do NOT appear on your public booking page and the lazy-seeded services live behind the scenes. The Negotiation meeting service is created once per tenant on first use;

> subsequent declines reuse it. The meeting durations available are 15, 30, and 60 minutes; pick the one that fits your sense of how long the conversation needs to be.

Amendments, in depth

Amendments are how you modify a fully_signed agreement without cancel-and-redraft. The flow respects the legal weight of the parent: the amendment is a separate legal instrument that names the parent by reference and signed date, the parent agreement remains intact and fully signed, and the amendment carries forward all locked clauses (parties, definitions, engagement, fees, payment schedule, signatures, entire agreement) automatically alongside the clauses you specifically picked to amend.

Step 1: pick the parent

The Amendment button on the dashboard opens a picker that lists every fully_signed primary agreement (amendments themselves cannot be amended; one amendment supersedes a prior amendment if you need another change later). Search by client name, agreement reference, matter type, or signed date. The picker carries a Recent agreements section at the top.

Step 2: pick the clauses

A grouped accordion lists every clause from the parent agreement organized by section. A clause-title search box at the top lets you find a clause by typing. Tick the clauses you want to amend. Three clause IDs are forbidden and the picker disables them: parties, definitions, signatures. Picking any fee-related clause auto-reveals every fee-related clause (fees, payment schedule, milestones, PFL surcharge, government fees, Stripe surcharge, wire surcharge, additional fees and change orders, advance payments, chargebacks, refund policy) together so a fee change is never accompanied by an inconsistent schedule.

Step 3: the Builder reopens scoped

The Builder opens with only the picked clauses visible. The right rail's filter dropdown is locked to the picked list; other Builder sections are hidden. You edit, autosave runs, the validator runs scoped to the amendment (it does

NOT re-validate unchanged clauses from the parent), and you finalize. The send flow re-uses the standard signing pipeline (per-party portal tokens, invitation emails, lifecycle view), but the invitation email subject and body carry an amendment context callout: 'You are signing an Amendment to Service Agreement AGMT-YYYY-XXXXXX, originally signed <date>'.

The legal preface on the amendment document

Three short paragraphs render at the top of the amendment, before the parties block. Paragraph 1 identifies the parent agreement by reference and signed date and confirms that this amendment is intended to be read alongside the parent. Paragraph 2 limits the scope of the amendment to the picked clauses (clauses not amended remain in force as signed in the parent). Paragraph 3 is a conflict clause: in case of inconsistency between the amendment and the parent, the amendment governs the amended clauses; the parent governs all others. Both EN and FR pairs are rendered alongside the rest of the bilingual chrome.

What never appears on the amendment

The amendment's internal reference (AGMT-YYYY-XXXXXX-A1 or -A2) is INTERNAL ONLY. The visible amendment document references ONLY the parent by parent's reference and signed date. The dashboard list and the audit ledger use the -A<n> suffix to track the amendment row uniquely; the legal instrument identifies what it amends by the parent, not by its own internal sequence number. The {{reference}} substitution in clause bodies resolves to the parent's reference on amendments to keep the legal instrument coherent.

Auto-billing on fee-related amendments

Amendments only auto-bill when the amendment touched a fee-related clause AND Billing on Signing is enabled at the tenant level. An amendment that touched only scope language, communications, or termination wording does not trigger a new bill. The amendment's auto-bill scope is the same as the parent's auto-bill scope (first installment vs full professional fee); the bill row in Single Bills references the amendment id as its source.

> **Note**: The lifecycle view on /dashboard/agreements nests amendments under their parent: parent agreement at the top, then its amendments indented below in chronological order. The group sort uses max(parent.updated_at, latest_child.updated_at) so a recent amendment floats the whole group up.

Resending, rotating tokens, and editing emails

Once an agreement has been Sent, the lifecycle view shows each party row with action buttons next to it. The most-used buttons are Re-send and Edit email. Both follow strict rules so the per-party portal-token system stays consistent.

Re-send

The Re-send button on a party row rotates ONLY that party's token and re-sends ONLY that party's invitation. Other parties' links stay live and continue to validate against their existing tokens. Re-send is the right action when a client says they did not receive the original email, when an email lands in spam and the client cannot recover it, or when you want to nudge an unsigned party after a few days. The platform stamps a token_rotated event on the audit ledger every time you Re-send. The Re-send button is hidden for parties who have already signed (their work is done) and for parties whose row has no email recorded (rare; would happen on a malformed setup).

Edit email

The Edit email button opens a small modal where you type the corrected email. On save, the platform updates the party row, rotates the token (the old link no longer validates), and sends a fresh invitation to the new address. The previously-emailed invitation cannot be used; it returns 410 Gone if a recipient tries to click it. Use Edit email when you typed an email wrong at Send time or when the client's email address has changed since Send. The signed PDF, when finalized, picks up the corrected email automatically because the renderer reads from the latest party row at render time.

Re-send after a change request is resolved

When you accept a change request via Reopen to draft, the entire agreement returns to draft status and every signature is invalidated. After you re-finalize the amended draft, the platform mints fresh tokens for every party (not just the requester) and re-invites everyone. This is the safest path: every signer sees the updated text and re-signs fresh, so no party can claim they signed before they saw the amended language.

> **Note**: When a Re-send fails to deliver (SMTP soft-fail), the Builder surfaces an inline 'Re-send failed; check the email address' error and the audit ledger records the failure. The platform never silently swallows email failures on the Service Agreements module.

Cancelling an agreement

Cancellation is available while an agreement is in draft, awaiting_signature, partially_signed, changes_requested, declined_negotiable, or declined_terminal status. After fully_signed, you do NOT cancel — you amend. The Cancel button on the lifecycle view opens a confirmation modal that asks for a cancellation reason (required) and a confirm-by-typing-the-reference safety prompt for non-draft cancellations. The audit ledger preserves the plaintext cancellation reason for legal defensibility; even if your client later disputes the cancellation, the record shows the reason you stated at the time.

What cancellation does

- Transitions the agreement to cancelled. The status is terminal; the agreement cannot be re-opened.
- Invalidates every pending portal token. A signer who clicks an outstanding invitation link sees a friendly 'this agreement has been cancelled' page.
- Cancels the auto-bill if one has been created from Bill on Signing AND it has not yet been paid. A paid bill is not refunded by the cancellation itself; you handle the refund separately under the Refund Policy clause.
-

Emails every party with a cancellation notification, framed for the party that is receiving it (different copy for a signed party, an unsigned party, and the declining party).

- Audit-ledger row with event_type=cancelled, the reason text, your user id, and the timestamp.

Hard delete on drafts only

A separate Delete draft button appears in the Builder right rail ONLY while the agreement is in draft status. Delete draft is destructive: it removes the agreement row, cascades to its uploaded documents in storage, and leaves no trace. Use it for genuinely-discarded drafts (you started a test agreement, you started a draft for the wrong client). Once the agreement has been Sent, you cannot Delete; only Cancel. There is no operator-handled escape valve to hard-delete a non-draft agreement.

Encryption and the audit ledger

Service Agreements is the most legally-sensitive module on the platform. The encryption posture matches that. Every PII column, every uploaded document byte, and every signed PDF is encrypted at rest under your tenant's data-encryption key. The audit ledger is UPDATE-immutable at the database trigger level so that historical signature events cannot be tampered with after the fact.

Two-layer encryption (DEK + KEK)

Each tenant carries a unique data-encryption key (DEK), generated lazily at first encryption-aware write. The DEK is wrapped under a master key-encryption key (KEK) that lives in Vercel environment variables, not in the database. Database backups, restored exports, and even an attacker with full database access cannot decrypt your data without also obtaining the master KEK. The DEK is the same for every encrypted column on every encrypted row for your tenant; the master KEK is the same for every tenant on the platform, rotatable independently.

What is encrypted

-

Client name, client email, client phone, client address, client business name (when applicable), client business address.

- Family member rows (every field that names a person).
- Party rows (signers' names, emails, phones, addresses).
- The complete draft_data JSONB (matter framing, scope text, fee numbers, instructions, every tenant-typed field).
- Uploaded documents (the AI Populate input documents) — bytes encrypted in the agreement-uploads bucket with .enc-suffixed paths; filenames encrypted at the column level.
- The signed PDF (the final encrypted-at-rest copy in the agreement-pdfs bucket; the in-memory copy for the email attachment is plaintext and discarded after send).
- Saved signature PNGs (per-member typed-or-drawn-or-uploaded signatures used for counter-signing).

Blind-index for searchable email

Searching for an agreement by client email exact-match would not work on encrypted column values. The platform addresses this with a blind-index column: client_email_hash is HMAC-SHA256 of the lowercased email under a separate search-index key (different from the master KEK). The index allows sub-millisecond exact-match lookups without decryption. It does NOT support substring or prefix queries; for fuzzy search, you scan the encrypted dataset linearly which is acceptable for the volumes the platform handles.

Audit ledger (INSERT-only via trigger)

service_agreement_signature_events is the canonical ledger for everything that happens to an agreement: sent, verified, signed, countersigned, change_request_filed, change_request_resolved, declined_to_sign, decline_ended_by_tenant, meeting_invited, meeting_accepted, resent_after_negotiation, token_rotated, cancelled, expired. The table carries a Postgres trigger (prevent_sase_update) that raises on every UPDATE attempt,

even from the service role. DELETE is allowed but only as a cascade from the parent agreement deletion (which is itself only available on drafts). The fields stored: agreement_id, event_type, party_role, party_name, party_email, ip_address, user_agent, token_hash_prefix, occurred_at. PII in the audit ledger (party_name, party_email, ip_address, user_agent) stays in PLAINTEXT for legal defensibility; the privacy story is the same.

Audit certificate appended to the signed PDF

On finalize, a single audit certificate page is appended to the rendered PDF as its last page. The certificate carries: agreement reference, agreement signed date, one row per signer with name, email, redacted IP (/24 CIDR), token hash prefix (first 8 hex characters), UTC timestamp of the signature event, signature method. The certificate is the legal record traveling with the PDF; the ledger row in the database is the canonical evidence. The two should always agree byte-for-byte (the certificate is generated from the ledger at render time).

> **Note**: If a master KEK is ever rotated for security reasons, every tenant's DEK gets re-wrapped under the new KEK in a controlled migration. Encrypted column values do NOT need to be re-encrypted (they were encrypted with the DEK, which stays valid). The two-layer design exists precisely so a KEK rotation does not touch tenant ciphertext.

Start from code (Intake Form or Service Proposal)

Start from code is the third primary action on the Service Agreements list page. The button opens a small modal that asks for one reference code in one of two formats: IF-YYYY-XXXXXX (an Intake Form code) or SP-YYYY-XXXXXX (an accepted Service Proposal code). On submit, the platform looks up the source, validates that it belongs to your tenant, validates that it has not already been converted to a Service Agreement (each source can spawn ONE SA; idempotency lock), and opens a fresh draft in the Builder pre-filled with everything the source carries.

From an Intake Form (IF-YYYY-XXXXXX)

- Client identity: full name, email, phone, address, date of birth, country of birth, country of citizenship — every field the intake captured.
- Family members: every family-member row the intake collected, including relationship, DOB, and any matter-role hints.
- Matter framing: matter category, matter type label, and an inferred intended outcome summary based on the intake's category branch.
- AI-extracted document data: when the intake was Premium-tier and the visitor uploaded documents, the extracted passport / ID / language-test fields flow into the seeded draft.
- Instructions notes: any text the visitor typed in the intake's free-text fields about urgency, file status, or previous attempts.

From a Service Proposal (SP-YYYY-XXXXXX)

- Client identity envelope: the full Service Proposal identity (client + family + matter defaults).
- Accepted option: every field from the prospect-accepted option. Scope.included, scope.excluded, professional fee, payment schedule (every row), indicative timeline, government fees (every itemized row, override flags preserved), service components (every row with applicantKind, representationStatus, feeTreatment), notes.
- Applied template version: the SA's applied_package_version_id is stamped from the proposal's choice, so the agreement carries the template provenance for audit even though you started from the proposal.
- Recipient: the prospect who accepted the proposal becomes the named Client on the SA (you can still edit before Send).

Back-link to the source

The seeded SA carries source_kind ('intake_form' or 'service_proposal') and source_id (the UUID of the source row). The source's detail page renders a 'Linked Service Agreement' emerald card pointing at the SA's lifecycle URL. The link is one-way (SA references source, source references back via the back-link card). The source's status advances to 'converted_to_service_agreement' on the conversion stamp so the source list can filter on completed conversions versus still-open intake submissions or proposals.

Per-user rate limit

Code lookups are rate-limited at 30 per minute per user. Generous for interactive UX (typing a code and clicking Look up); caps runaway loops at 1,800 lookups per hour. A typo lands a generic 'we could not find that code' message rather than revealing whether the code exists in another tenant or has already been converted. Use Copy on the source detail page to avoid retyping when you can.

> **Note**: Start from code is the bridge between top-of-funnel data capture (intake forms, sales proposals) and bottom-of-funnel legal commitment (the signed Service Agreement). When you Convert a proposal or intake into an SA, you are NOT discarding the source; the source stays in its own module's list, its own audit ledger, its own dashboard surface. The SA is a sibling record that points back. This preserves the negotiation record even after the engagement starts.