CASE STUDY

Building a Local AI Lead Generator on an Existing CRM Backbone

A law firm in Toronto · Real Estate & Related Practices

Suganth Srichandramohan · Aruvi Consultancy Services Inc. · 2026

Figure 0. Engagement visual: structure over spectacle.

1. Executive summary

A law firm in Toronto engaged Aruvi Consultancy Services to advance practical AI adoption inside the firm. I had expected to start with a broad readiness assessment. After the first meeting, the work became a focused delivery: a **Lead Generator** module that sits on the firm's existing CRM foundation, runs on the client network, and helps staff target wills and estates outreach and cross-sell opportunities without sending client lists to the public cloud.

I did not treat a frontier model as the default engine for every filter. The solution uses **pandas** for fast, deterministic filtering of the full client set, and a **local Ollama model** only when natural-language interpretation is truly needed. Common phrasing ("purchase in the last two years," "over age 40," "with email") runs as **fast filters** in milliseconds. I think of that as the Corolla that completes the business process, rather than leasing a slower, more expensive Ferrari of frontier AI for every click.

The MVP was designed, built, hardened, and deployed to the firm's shared environment with live data and operating guides. The CRM backbone, already started by the firm's tech lead and team, became the platform on

which this module (and later local AI capabilities) can expand.

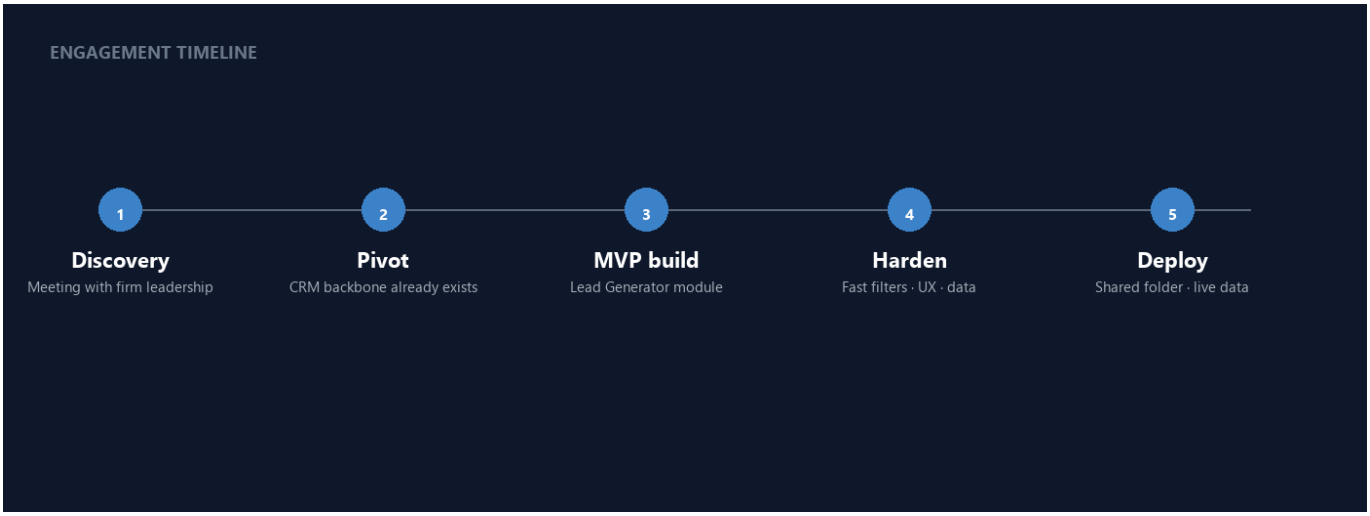

Figure 1. Engagement arc from discovery through deployment.

2. Client context

The client is a law firm in Toronto with strength in real estate and related client work. Like many professional services firms, growth multiplies relationship data: purchasers, sellers, refinance clients, referral sources, and life-stage moments that should trigger follow-up, especially wills and estates after property transactions.

The firm was not starting from zero on technology. Under the managing partner and the tech lead, the practice already had an HTML CRM backbone tied to a working spreadsheet: people, file history, parties, and the beginnings of a relationship layer. That was the book of record. That fact reshaped the engagement from “diagnose everything” to “accelerate the right next module.”

3. Initial meeting and the strategic pivot

3.1 What Aruvi expected to find

The initial posture was classic consulting entry: meet leadership, assess AI readiness, map risks, and produce a broad opportunity landscape. That approach is often correct when a firm has ambition but no platform, no data discipline, and no internal technical ownership.

3.2 What the meeting actually revealed

In the working session with the managing partner and the tech lead, it became clear the firm was already on a path to AI adoption. There was a game plan. The CRM base layer was real. The surprise was productive: the highest-leverage help was not a multi-month assessment tour. It was to **boost a module in the CRM**: a lead generator using a local LLM, so staff could target campaigns and cross-sell without sending client lists to the public cloud.

The vehicle was half-built. The consulting job was to pick the next foothold that staff could use, that respected confidentiality, and that strengthened the backbone rather than bypassing it.

Figure 2. Initial assumption versus the path taken after the first meeting.

4. The problem to solve

Even with a CRM foundation, relationship intelligence often stays locked in columns. Staff know intuitively that recent purchasers without wills are good outreach targets, or that clients in their forties may also need a conversation about aging parents. Executing that at scale requires either painful spreadsheet filters or tools that are too heavy, too cloud-dependent, or too slow.

- Target lists for wills and estates without re-inventing the CRM
- Cross-sell rationales that feel specific to each client (hyper-personalization), not generic blasts
- Keep all client data on the firm network. No bulk export to frontier APIs
- Ship something usable quickly, then expand
- Avoid architecture optimized for a consultant’s demo reel rather than the firm’s long-term cost and risk profile

5. Approach and principles

The engagement followed a simple discipline: align to how the firm wants to operate, ship a viable module on real data, and refuse complexity that does not buy process outcomes.

5.1 Principles that governed design

- **Build on the CRM, do not bypass it.** Person and matter history remain the system of record.
- **Local by default.** Ollama on the machine; pandas on the full set; no cloud round-trips for client tables.
- **Fast path first.** Common business language becomes filters without waiting on a large model.
- **LLM as interpreter, not analyst of the whole book.** Natural language becomes structured filters; ranking stays deterministic.
- **Stand-alone value, expandable surface.** The Lead Generator pays for itself now; the same local stack can answer other queries later.
- **Deliberate architecture over cool factor.** Prefer a maintainable Corolla for routine process over a Ferrari that impresses peers and invoices monthly.
- **Humans stay in the loop.** Recommendations are decision support. A lawyer still reviews before anyone is contacted.
- **Privacy and CASL shape the lists.** Campaigns stay inside the firm’s existing client relationship and consent rules, not only inside the software.

“Some consulting practices optimize for the portfolio piece, the flashiest stack to resell or list on a résumé. This engagement optimized for deliberate architecture that saves the firm over years: local models where structure is enough, frontier capacity only when the business truly needs it.”

6. Technology stack: first intent, then the real design

6.1 Initial technical posture

The first MVP draft assumed a green field: a new structured data layer (SQLite), ingestion of client documents, and a local RAG stack. That is a reasonable default when a firm has ambition and no book of record. After reviewing the firm’s materials, that draft was the wrong plan. They already had an HTML backbone calling a spreadsheet. Version 0.2 dropped the rewrite. The job became a focused module on architecture the firm already owned.

6.2 Revised stack after discovering the CRM backbone

Once the firm’s CRM layer was understood (HTML backbone plus an Excel working model with People, File_History, File_Parties, and related relationship tabs), the stack was re-centered on integration and locality:

Layer	Choice	Why
Existing backbone	HTML CRM + working Excel workbook on the firm network	Already the book of record; Lead Generator sits alongside it, same-folder deploy
Data access	pandas + openpyxl	Deterministic joins, filters, and summaries on thousands of rows in milliseconds
Application UI	Streamlit	Fast internal tool delivery; staff open a browser; no separate web ops for MVP
Natural language	Ollama + qwen2.5:14b (local)	Interpret unusual queries without sending client tables to frontier APIs
Common queries	Fast filter engine (rules + patterns + scenario JSON)	Purchase/date/age/email phrases run instantly; avoids multi-minute model waits
Scenario library	campaign_scenarios.json	Firm-owned few-shot examples staff can extend without retraining weights
Launch	Open Lead Generator.bat	Day-to-day double-click; no PowerShell for end users after setup
Fallback	Keyword heuristic if Ollama is offline	Staff are not blocked when the local model is not running
Documentation	User guide, install/load guide, scenario catalogue, CRM roadmap	The firm can run, extend, and plan the next layers without a rewrite

6.3 Why pandas

Pandas was chosen because the hard problem was never “can a model invent a list of people?” The hard problem was “can we apply precise business filters to the full book reliably?” Matter type, closing window, age band, presence of email, absence of wills work. These are set operations. Spreadsheets already do them;

pandas does them programmatically at scale, with reproducible logic, on the same machine as the file.

6.4 Why Ollama (and when not to use it)

Ollama hosts a local large language model so staff can type plain English. The model's job is narrow: translate intent into a structured filter object. It is not fed the client table. Early versions that sent every query through the 14B model were too slow for daily use. The architecture was therefore hardened: **common wording uses fast filters; the LLM is the exception path.** That single decision is the difference between a demo that impresses once and a tool people reopen.

Figure 3. Fast path versus local LLM path: process first, model second.

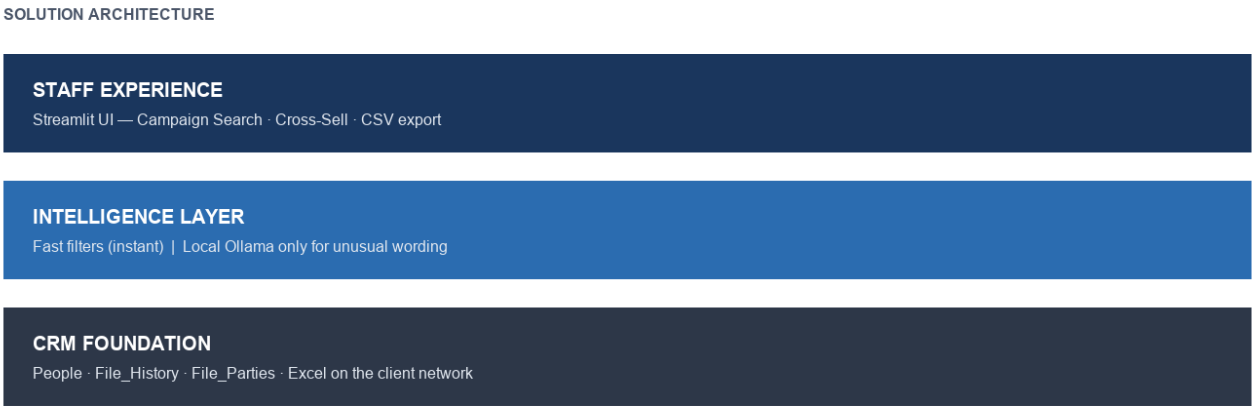

Figure 4. Three-layer solution architecture on the CRM foundation.

7. Engagement journey: every major step and why

Step 1: Discovery meeting with the managing partner and the tech lead

Establish goals, constraints, and existing technology ownership. Outcome: recognition that AI adoption was already underway; the consulting offer shifted from broad assessment to module delivery.

Step 2: Inspect the CRM working spreadsheet

Map People, File_History, File_Parties, and supporting tabs. Confirm header rows, ID conventions, and gaps (for example names missing, PersonID mismatches, empty DOB/email). Data quality became part of the product story, not an afterthought.

Step 3: Repair and enrich test data for a trustworthy demo

Populate FirstName/LastName and ContactType samples; align File_Parties PersonIDs to People so joins do not yield empty names; add Date of Birth distributions for age-based scenarios; later seed email addresses for ~30% of people so email-blast filters can be tested realistically.

Step 4: Define product scope: AI Search + Filtered Search

Two modes: natural-language campaign targeting (AI Search), and rules-based opportunities with personalized rationales (Filtered Search: recent purchase, refinance, age band, co-owner, city). Scope protected schedule and clarity.

Step 5: Implement data_loader and client summary

Load Excel with header=1; join parties to history to people; derive FullName, Services, FileCount, Age, City from property address when needed, and flags such as HasPurchase / HasWills.

Step 6: Implement analysis layer

Query translation (heuristic + optional Ollama), structured filter application, and cross-sell rules engine with client-specific rationale text.

Step 7: Ship Streamlit UI (MVP)

Sidebar global filters, tabs for both modes, CSV export, local-only messaging, reload controls, and a Recent Queries list so useful AI Search prompts can be rerun. AI-assisted development (including Grok Build) accelerated implementation under the privacy constraint.

Step 8: Scenario library and few-shot guidance

A firm-editable JSON file encodes about a dozen campaign patterns already supported by the CRM fields: recent purchasers and sellers, refinance, age bands (including the mid-forties plus aging-parents conversation), condos, co-owners, multi-file clients, neighbourhood clusters, and referral-source enablement. Staff can add phrases or a new campaign without retraining model weights.

Step 9: Performance hardening: fast filters

After observing multi-minute waits on simple queries, common language was moved to instant pattern matching so the client never pays LLM latency for “purchase last 2 years over 40.”

Step 10: Productization for the firm

Launcher, shared-folder layout with the existing CRM, email filter for blasts, and the documentation pack: end-user how-to, install/load guide, scenario catalogue, and a 3 / 6 / 12 / 24-month CRM roadmap.

Step 11: Client network placement and live data

Files placed with the CRM on the shared folder; Python venv and dependencies installed; Ollama and model pulled on machines that run natural-language search; debugging against live firm data; validation that double-click launch and browser UI work on the client environment.

Step 12: Working session and knowledge transfer

Demo of common scenarios, review of CRM Roadmap and Scenarios documents, next-step agreement on further help, and access requirements for sustained ownership by the firm's team.

8. What was delivered

8.1 Lead Generator capabilities

- **AI Search.** plain English → filters → matching clients; export CSV for call teams
- **Filtered Search.** wills after purchase/sale/refinance; investor/corporate flags; personalized rationales
- **Recent Queries.** last AI Search prompts kept on the tab so staff can rerun a useful search
- **Global filters.** time range, matter type, city/province, contact type without any LLM
- **Email-ready lists.** fast filter for people with email on file for blast planning
- **Scenario library.** about a dozen firm-editable campaign patterns, plus JSON staff can extend
- **Offline fallback.** keyword path if the local model is not running
- **Operational packaging.** launcher, same folder as the CRM, install and user documentation

8.2 Hyper-personalization in practice

Recommendations are not generic. A mid-forties client with recent real estate activity can surface both a personal wills message and a family conversation about parents often in their sixties or seventies. That pattern (hyper-personalization) aligns with deliberate AI project framing (including CPMAI study applied in practice): name the pattern, bound the use case, measure usefulness.

9. Outcomes

Outcome	Detail
Usable MVP on real CRM structure	AI Search + Filtered Search operating against the firm's HTML + Excel data model
Privacy posture	No bulk client rows sent to cloud frontier APIs for filtering
Speed for daily work	Common queries milliseconds; LLM reserved for unusual phrasing
Deployability	Same-folder placement with CRM; bat launch; documented Ollama install
Expandability	Local stack and scenario JSON ready for additional query types
Strategic alignment	Module strengthens CRM backbone instead of creating a side AI island

10. The Corolla, not the Ferrari

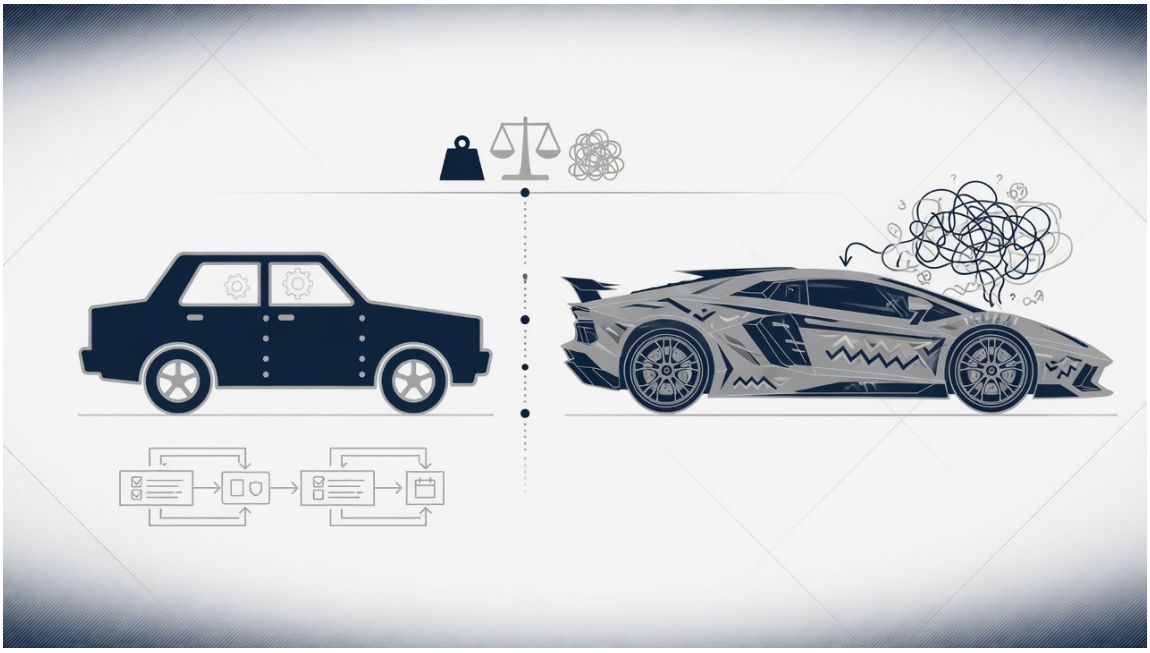

Figure 5. Conceptual metaphor: fit-for-purpose reliability versus over-specified performance for routine process.

Frontier models are extraordinary tools. They are not always the right engine for a law firm's routine campaign filter. When business logic is structured (matter type, closing date, age, email presence), the correct system is a fast, local, inspectable filter path. When language is ambiguous, a local model can interpret intent. When the problem truly requires frontier capability, that can still be procured, but as an exception, not a habit that burns budget and data risk on every click.

There is no need to lease a Ferrari for trips the Corolla already covers in the business process. Some consultants and firms still prioritize the cool factor, the stack that photographs well for resale narratives and personal brand. This engagement prioritized **deliberate architecture**: lower long-run cost, lower leakage risk, higher staff adoption, and a backbone that compounds.

11. From Lead Generator to firm-wide local AI backbone

The CRM is not only a marketing database. With discipline, it becomes the operating system of the firm: identity, matters, consent, fees, and eventually reception and billing handoffs. The roadmap left with the firm is layered on purpose (about 3, 6, 12, and 24 months): trust the data, then a staff UI, then invoices and a front-desk pilot, then one backbone. The same local intelligence pattern (structured data, fast rules, optional local LLM interpretation) can extend to:

- Additional campaign domains beyond wills (with new scenarios in JSON, not new cloud contracts)
- Staff UI replacing spreadsheet typing while preserving the same IDs and joins
- Invoice drafting and aging views fed from matter data
- Front-desk validation flows (phone + DOB/address) against the same person records
- Internal Q&A over firm procedures where answers must stay local

Each expansion reuses the same philosophy: structure first, local compute second, frontier last. That is how a single module becomes a platform without rewriting the firm every quarter.

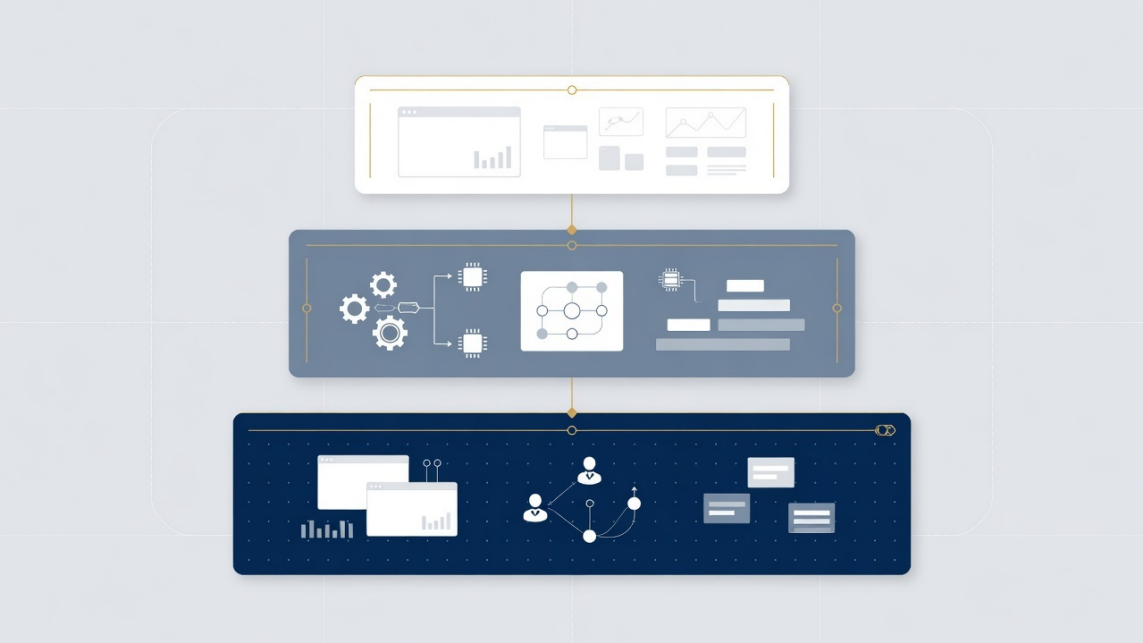

Figure 6. Layered foundation thinking: build up from CRM truth, not down from hype.

12. Lessons for other professional services firms

- Meet technical owners early; you may discover a backbone that changes the entire proposal.
- Ship a module that staff open next week, not only a roadmap they admire.
- Measure latency as a product requirement. AI that waits two minutes for a two-second filter will die in operations.
- Encode firm language in scenario libraries so “training” is editable, not mystical.
- Be willing to look less flashy if the architecture is more honest about cost, privacy, and maintenance.

13. What this leaves us with

This case study documents a completed arc: from an initial meeting that reset scope, through MVP build and hardening, to placement on the client network with live data, debugging, and operational guides. The Lead Generator is a stand-alone success and a foundation stone. The CRM backbone, and the local intelligence pattern around it, is the long game. If a similar first module would help your firm, I am glad to talk it through.

Suganth Srichandramohan

Founder, Aruvi Consultancy Services Inc.

Suganth@AruviConsultancyServices.com

This document is intended for download from the related public case study on AI consulting work with a law firm in Toronto. Client operational details are described at a level appropriate for public sharing. The client is not named at the firm’s request.