A look under the hood at Prompting AI

We are two-thirds of the way through a 3-year project, to build a RAG/LLM for Social Housing.

RAGs or Retrieval-Augmented Generation systems are the brain, the LLM (Large Language Model), is the mouth. They are used, in this instance, to analyse fixed/historical data. They do not do real-time – that would be expensive and susceptible to incorrect responses from poor data.
Our journey has been fascinating, but for us who don't code, or manage the data, we were fairly redundant, for the most part.

Stage 1: We used Machine Learning to organise our data – 580,000,000 pieces of it.

Stage 2: We built a system that uses the same type of AI techniques as ChatGPT/Claude etc., but it only has access to our data. It wouldn't know a dog from a cat.

Stage 3: Now we're on the final leg – training the software. The team got bigger, and I got involved full time.

In building this ‘rule book’ we are creating our competitive ‘moat' – the part of a business/system that can’t be easily replicated. One day, everyone will have a RAG, but for most applications, it will be a supercar without fuel.
The value is not the AI, (although brace yourself people, this is not going to be cheap). The value is in the rules, definitions, and domain logic that train it how to interpret housing data, apply filters correctly, and produce answers people can trust.

It will take us around 12-18 months (we're 7 months in) to train our RAG to answer 95% of ALL possible questions.

At this point it can answer, or we know why it got it wrong, nearly all our questions, but we are looking for 'edge-cases', and they get harder to find. Folks say that you will never get better than 95% - there will always be questions that haven't been asked yet.

Tech people use 'Prompt Injection' – writing prompts designed to break the system, then they fix it. When professional Retrofit/Decarb people start to use 'Locarla Property Insight', we hope to find a lot more edge-cases with their experience.

Working with big data does require a mindset change. Creativity is key. Or ask an LLM.

Let's get prompting.

1. Framing the AI's Role

Role prompting

'You are an expert horticulturist/scientist/quantity surveyor ….' – inconclusive whether this works or not. Probably not.

The research on role prompting is mixed. Some studies suggest it makes no measurable difference to output quality, while others show marginal improvements in specific contexts. Clear and explicit instructions produce better results than anything else.

2. Improving Answer Quality

Self-criticism

'Check your previous response. Is it correct.'

Put your LLM on the spot, say it is important for you to get this right, and you need confirmation that it is.

3. Teaching by Example

Few-shot prompting – 'Here's what good looks like.'

If you want an LLM to write in your style, you need to show it what your style is.

1 (or 0)-shot

> Draft an email to colleagues reminding them not to block open the fire doors.

2-shot

> This is an email I wrote to staff regarding new work policies. Draft an email in my style to colleagues, reminding them to not block open the fire doors.

Few-shot

> Here are 3 of my example emails, and a piece on fire safety I wrote for a blog. Draft an email, in my writing style, to colleagues reminding them to not block open the fire doors.

Two types of prompts

Conversational
You talk to the AI like a person and ask it to help with writing, summarising, or explaining things – think chatbots – Copilot, Gemini etc.

Product/System
The software translates what the user asks into a carefully structured instruction, so the AI gives consistent, trustworthy answers.

Our product is Social Housing data, so we need to train our RAG/LLM to understand what our data is all about, and, probably more importantly, the types of questions our clients will ask. Here's a good example –

“Who owns the most energy-inefficient homes in areas of highest deprivation?”

The user is looking for 'high deprivation'. In our database deprivation data is stored as 'Living Standards' – 0 being the lowest, with 10 the highest living standards areas. The RAG understands that deprivation is the same as living standards so gave properties in the 'higher' range. We needed to write a rule to train our system to recognise that in future 'high deprivation' = 'low living standard'.

Stopping the AI from Making Things Up

Response Governance Policy:

(You can cut and paste these instructions, directly into your prompt.)

1. "If information isn't found, say 'Not found in documents.' Don't guess."
2. "Base your answer ONLY on the uploaded documents."
3. "For each claim state the specific location – document name, specific page/section, and a relevant quote."

Decomposition

Breaking down complex or multi-level prompts into more manageable tasks to get more accurate responses.

"Before you answer this prompt, detail any 'sub-problems' that need to be solved first."

"Which Riverside homes in London should we prioritise for retrofit funding this year?"

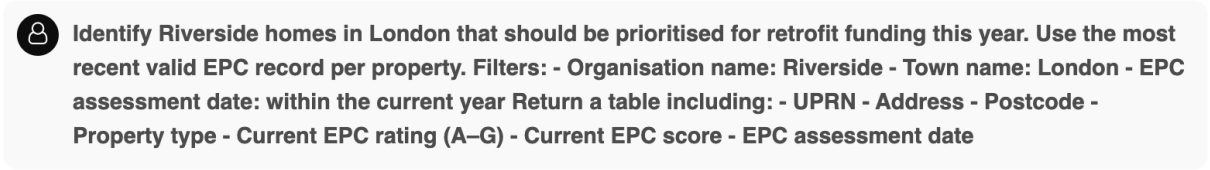
Straight forward question but answering it requires resolving multiple underlying analytical and policy decisions:

• Ownership filtering
• Geography
• EPC status
• Fabric characteristics
• Improvement potential
• Strategic prioritisation logic

The RAG can then decide how to break down the task into stages:

1. Filter Riverside-owned stock in Greater London,
2. Select properties with valid current EPC ratings of D–G,
3. Identify properties with fabric characteristics suitable for cost-effective retrofit (cavity walls, low roof insulation, etc.),
4. Exclude properties with limited improvement potential,
5. Prioritise those in poorer Living Quality deciles or with high EPC uplift potential,
6. Return a ranked or grouped output suitable for funding prioritisation.

First, we can use the Wand tool to improve the prompt structure. Clear instructions improve understanding and accuracy.



We can then view the data full screen or export.

Training Your RAG: The Rule Book Approach

The difference between a RAG that works and one that doesn't, lies in the rules. Think of these rules as the operating manual for how your AI interprets queries, and formats responses. They're the difference between getting back usable intelligence and getting back formatted garbage.

We organise our rules into three categories:

Language & Localisation Standards

These might seem trivial, but inconsistency here creates friction for users. Our system is UK-focused, so we enforce:

- UK spelling: summarise, organise, analyse, colour
- UK terminology (e.g. "postcode" not "ZIP code")
- UK housing phrasing (e.g. "local authority", not "municipality")

Domain Semantics (Housing Terminology Rules)

This is where the real work happens. Our domain ‘Housing’, has specific meanings for common words, and the RAG needs to know them:

"Property" vs "Building"
In social housing, a building can contain multiple properties (flats). The distinction matters for counts, ownership, and EPC assessments. We trained our system to recognise context clues and ask for clarification when ambiguous.

"Owned stock" vs "Managed stock"
Housing associations may manage properties they don't own. This affects which organisations appear in certain queries and what data is available.

EPC validity rules
EPCs expire after 10 years. Our system knows not to use expired certificates for current assessments, and flags when data is stale.

Output Governance (Response Behaviour Rules)

These rules control how the RAG presents information and not just what it finds.

Aggregated metrics vs row-level records
When someone asks to 'see everything', they rarely want 85,000 rows of addresses. They want summary statistics, breakdowns by key dimensions, and the option to drill down. Our listing rule (below) handles this.

One building vs multiple dwellings
Related to domain semantics but governs how counts and totals are presented to avoid confusion.

"If not found, say not found" / no guessing
LLMs are trained to be helpful, which means they'll confidently make things up rather than admit they don't know. We explicitly forbid this. If the data doesn't contain the answer, the system must say so.

When "Show Me Everything" Doesn't Mean "List Everything"

Show me everything you have on L&Q

From a total of 85,949 records on L&Q Group, we're showing 50 results here. The addresses range from 19, Dora Way, London, SW9 7EN to 63b, Stondon Park, London, SE23 1LB.

Address	UPRN	Registered Owner
19, Dora Way, London, SW9 7EN	10000446696	L&Q Group
Flat 12, 36, Hayter Road, London, SW2 5AP	10008787879	L&Q Group
Flat 38, Chatsworth Court, Ederline Avenue, London, SW16 4SE	100020645108	L&Q Group
31, Chaucer Road, Welling, DA16 3NH	100020266941	L&Q Group
19, Waring Road, Sidcup, DA14 6SH	100020263232	L&Q Group
6, Aspen Gardens, Ashford, TW15 1ED	33039448	L&Q Group
12, Hillside Road, Dartford, DA1 3NS	100020220937	L&Q Group

Observed failure mode:

• The RAG returned rows of individual properties (UPRN/address lists).
• This contradicted user intent, which was clearly organisational overview / summary.

The Listing Rule

If a user uses terms like overview, summary, assessment, or analysis, the system must return aggregated organisational-level metrics and must not return individual property-level rows.

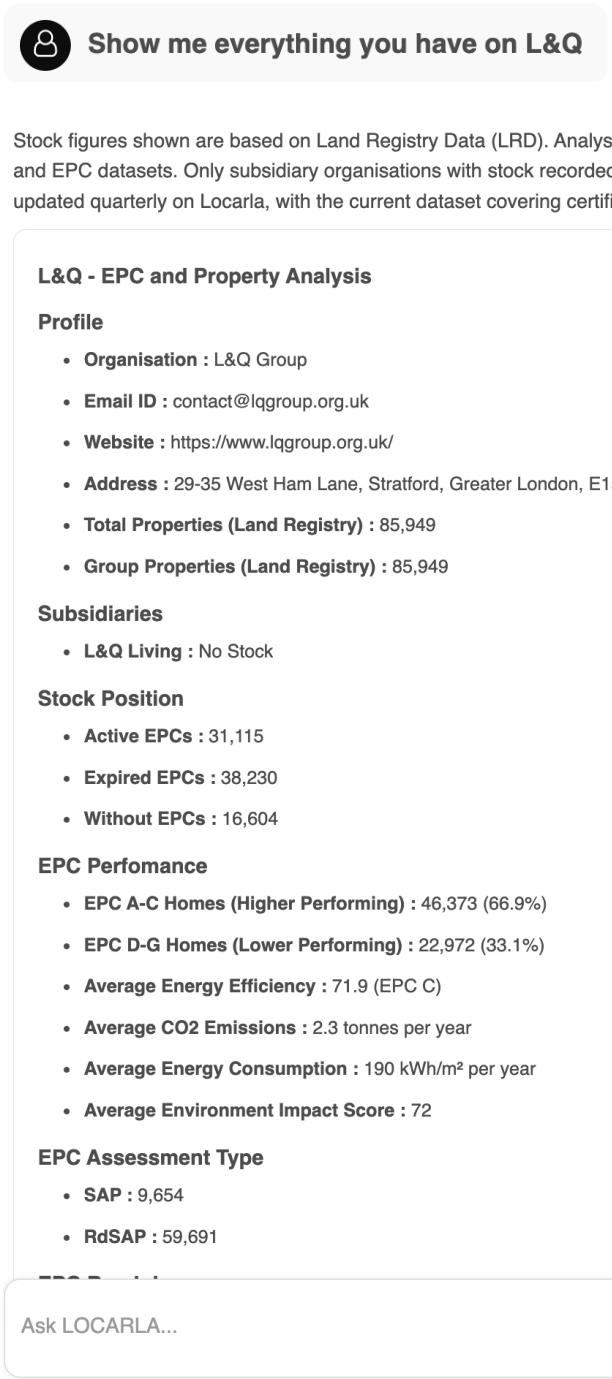
This single rule eliminated a whole class of usability problems. Users wanted to understand the portfolio, not scroll through thousands of addresses. Now when they ask, 'Show me everything you have on L&Q', they get:

- Total properties and breakdown by region
- EPC performance summary (active/expired/missing)
- Stock position by rating band
- Key metrics (average energy efficiency, CO2 emissions, etc.)
- Assessment type breakdown

Not 85,949 rows of addresses.

The lesson: Watch what breaks. When the same type of question consistently produces the wrong format, write a rule. Test it. Add it to the rule book. Each rule you write makes every subsequent query better.

www.locarla.com



Our system improves through closed feedback loops. When the output doesn't match what we know to be true, those discrepancies are recorded and used to refine the data logic, domain rules, and retrieval behaviour. This ensures the platform becomes more accurate and reliable over time, rather than simply producing plausible sounding answers. No one ever said this was going to be easy.

Gaz