PropVisions

The Complete AI-Powered Property Investment Platform

White Paper

Version 1.0 — Confidential Draft for Early Partners and Reviewers

Last updated: August 27, 2025 • www.propvisions.com • hello@propvisions.com

Contents

1 Executive Summary **1**

2 Beta Status and Transparency **1**

3 Product Overview **1**
3.1 Elevator Pitch . . . 1
3.2 What Works Today (MVP) . . . 2
3.3 Planned Features (Highlights) . . . 2

4 Design Principles & Defaults **2**

5 User Personas & Jobs-To-Be-Done **3**
5.1 Professional Investor . . . 3
5.2 Sourcing Agent . . . 3
5.3 Estate Agency / Partner . . . 3

6 System Architecture **3**
6.1 High-Level Diagram . . . 3
6.2 Workflow Orchestration (n8n) . . . 3
6.3 Data Stores . . . 4

7 Canonical Schemas (Selected) **4**
7.1 Core Entities . . . 4

8 Algorithms & Methods **4**
8.1 HTML → JSON Extraction . . . 5
8.2 Vision-Guided Refurb Estimation . . . 5
8.3 Rent Estimation (Hybrid) . . . 5
8.4 EPC Matching (Heuristic + Fallback) . . . 5
8.5 Financials & Mortgage . . . 5
8.6 Uplift & Forecasting . . . 5

9 Prompts & Guardrails **6**
9.1 Strict JSON Output Guard (Global) . . . 6
9.2 HTML-to-JSON . . . 6
9.3 Refurb (Vision) . . . 6
9.4 Rent . . . 6
9.5 EPC Fallback . . . 6

10 Evaluation & Accuracy **7**
10.1 Metrics . . . 7
10.2 Targets (Beta) . . . 7
10.3 Protocol . . . 7

11 Security, Compliance, and Data Governance 7
11.1 GDPR 7
11.2 Credentials & Secrets 7
11.3 PII & Retention 7

12 Performance, Reliability & Cost 8
12.1 Latency Budgets 8
12.2 COGS (Illustrative) 8
12.3 Reliability 8

13 Commercial Strategy 8
13.1 Pricing (Indicative) 8
13.2 GTM 8
13.3 Unit Economics 8

14 Operational Playbooks 9
14.1 Incident Response 9
14.2 Monitoring & Observability 9
14.3 Data Quality Gates 9

15 Mortgage Module (Now & Next) 9
15.1 MVP 9
15.2 Next 9

16 Deal Discovery & Off-Market 9
16.1 Watchers 9
16.2 Off-Market Assistant 9

17 Roadmap (Detailed) 9
17.1 Weeks 1–6 (MVP Polishing) 9
17.2 Launch (v1) 10
17.3 Q2 10
17.4 Q3 10
17.5 Q4 10

18 Risk Register & Mitigations 10

19 Case Study (Illustrative) 11
19.1 Deal Summary 11
19.2 Outputs 11

20 Support, SLAs, and Operations 11
20.1 Beta Support 11
20.2 SLA Targets (Post-Beta) 11

21 Conclusion 11

Appendix A: Supabase SQL DDL (Selected) 11

Appendix B: n8n Workflow (Pseudo-YAML) 13

Appendix C: Report Structure (PDF/Excel) 13

Appendix D: API Stubs (FastAPI) 13

Appendix E: Prompt Library (Copy-Paste Ready) 14

Appendix F: Glossary 14

Executive Summary

PropVisions is an AI-first property investment platform that automates the end-to-end lifecycle of sourcing, underwriting, reporting, and monitoring UK residential deals. The system combines: robust ingestion from URLs/APIs; computer-vision-driven refurbishment estimation from listing photos; rent band forecasting using historical data and listing signals; full-stack financials; EPC enrichment; comparables sanity checks; high-quality PDF/Excel outputs; and ongoing alerts post-purchase (rates, remortgage timing, deal-status changes).

What this document covers. Current MVP and near-term roadmap, reference architecture, canonical schemas, algorithms, operational controls, security & GDPR posture, evaluation methodology, commercialization path, and pilot structure.

Positioning. Unlike calculators or point tools, PropVisions provides: (i) automated upstream data capture, (ii) transparent, AI-assisted analysis with human-in-the-loop overrides, and (iii) post-purchase monitoring—forming a continuous improvement loop.

Outcome. Reduce analysis time from hours to minutes; standardise underwriting; deliver client-ready packs; and scale to hundreds of deals per analyst per week with auditable assumptions.

Beta Status and Transparency

Active beta. Fast progress; imperfect outputs. UI is practical-first. Accuracy is variable by area and listing quality. We publish baseline accuracy metrics and expose confidence bands and editable assumptions.

- **Target accuracy (near-term):** rent band hit-rate $\geq$ 80%; refurb total MAPE $\leq$ 20–25% with 10+ images; EPC match $\geq$ 95% on clean addresses.
- **Levers you can edit:** management%, voids, maintenance, fees, growth assumptions, refurb line items, rent bands.
- **Human judgement remains central:** we provide structure, speed, and signal—not guarantees.

Product Overview

Elevator Pitch

Paste a property link (or run a saved search) and PropVisions outputs a complete investment snapshot—refurb costs by room, rent band with confidence, yield/ROI/cashflow, EPC, comps, and report files—then watches for updates (price drops, back-on-market, remortgage windows) and alerts you.

What Works Today (MVP)

1. **URL ingestion** → scrape & parse: price, address/postcode, layout, features, images, agent details.
2. **Refurb from photos** (vision-guided): per-room items; regional cost indices; editable totals & contingencies.
3. **Rent estimation:** historical rent data + listing features → monthly/annual band with rationale & confidence.
4. **Financials:** stamp duty, legal/survey/insurance, management%, voids, maintenance → yield, ROI, cashflow.
5. **EPC enrichment:** rating & recommended measures; record stored for reporting.
6. **Comps sanity checks:** recent locals; £/sqft bands; outlier flags.
7. **Outputs & alerts:** PDF/Excel; Slack/email notification with links.

Planned Features (Highlights)

- **Unified search & APIs:** Rightmove, Zoopla, OnTheMarket, auction platforms; de-dup, canonicalisation, watchlists.
- **Deal-status watcher:** new/returning, price drops, time-on-market; filters by postcode/price/bed/yield/refurb budget.
- **Mortgage module:** product types (IO/repayment), fees/ERCs, ICR/DSCR, forward curve scenarios, remortgage assistant.
- **Portfolio & sharing:** multi-deal comparisons, read-only client views, white-label branding.
- **Off-market assistants:** GDPR-compliant outreach, probate/distress signals, light CRM for cadence tracking.

Design Principles & Defaults

- **Transparency over black-box:** show bands, confidence, and levers; capture overrides and provenance.
- **Speed with correctness:** deterministic parsers & validations first; LLMs only where they add value.
- **Local realism:** regional cost indices for refurb; postcode priors for rent.
- **Secure by default:** least-privilege, audited access, encrypted storage, no secrets in code.
- **Human-in-the-loop:** every estimate is editable; edits feed calibration.

User Personas & Jobs-To-Be-Done

Professional Investor

JTBD: underwrite consistently; generate investor packs; track portfolio KPIs; spot remortgage opportunities.

Sourcing Agent

JTBD: scan many deals; qualify quickly; package for buyers with credible numbers; automate alerts.

Estate Agency / Partner

JTBD: add value to buyers with analytics; white-label reports; evidence-based pricing; improve conversion.

System Architecture

High-Level Diagram

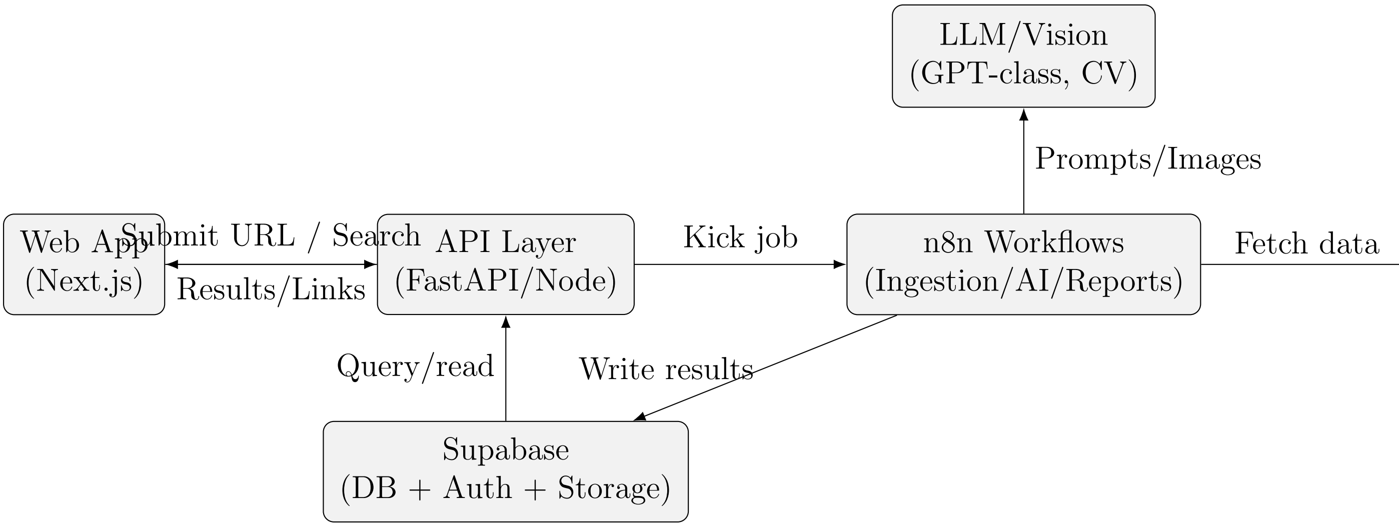

Figure 1: PropVisions high-level architecture.

Workflow Orchestration (n8n)

1. **Scrape Node:** fetch HTML/media with retries, backoff, & robots/T&Cs respect.
2. **Clean & Parse:** deterministic selectors + LLM cleanup; strong schema validation.
3. **Image Loop:** refurb agent per image; classify room, extract conditions; aggregate by room type.
4. **Rent Agent:** regression prior + LLM adjustment (listing text, amenities, finish).

5. **Fees & ROI:** stamp duty, legal, insurance, mgmt, maintenance, voids; yield, ROI, cashflow.

6. **HPI/Uplift:** baseline growth; optional AI uplift vs comps; sensitivity ranges.

7. **EPC Match:** heuristic + LLM fallback; store matched record + recommended measures.

8. **Reporting:** render PDF/Excel; upload artifacts; generate share links.

9. **Notify:** Slack/email with KPIs and download URLs.

Data Stores

Supabase Postgres: canonical tables for properties, analyses, refurb/rent/EPC/comps, runs, and reports.
Supabase Storage: artifact storage (PDF/Excel).
Auth: users, teams, roles; future SSO.

Canonical Schemas (Selected)

Core Entities

Table	Purpose & Key Fields
`properties`	Canonical property identity. *Fields:* id, url, address_full, postcode, source, first_seen, last_seen, status, agent jsonb.
`analyses`	One analysis per run/version. *Fields:* id, property_id, run_timestamp, price, layout jsonb, assumptions jsonb, results jsonb, confidence jsonb.
`refurb_estimates`	Room-/item-level refurb costs. *Fields:* id, analysis_id, room_type, items jsonb, subtotal, contingency, confidence.
`rent_estimates`	Rent predictions and sources. *Fields:* id, analysis_id, monthly_rent, band_low, band_high, method, comps jsonb, confidence, rationale text.
`sales_comps`	Nearby sales comparables. *Fields:* address, distance_m, price, date, sqft, ppsf, weight.
`rental_comps`	Nearby rental comparables. Similar to sales comps, with monthly rent.
`epc_records`	EPC details. *Fields:* rating, uprn, lmk_key, measures jsonb, full_record jsonb.
`reports`	Generated artifacts. *Fields:* type (pdf/xlsx), storage_path, size_bytes, checksum.
`runs`	Orchestration metadata. *Fields:* workflow_id, duration_ms, status, error_log, tokens_used.

Algorithms & Methods

HTML → JSON Extraction

Deterministic tag/attribute rules to collect candidates (price, address, beds/baths, agent, features), then an LLM cleans/standardises. Ambiguity yields warnings; never fabricate. **Validation:** regex & unit checks, currency normalisation, postcode parsing, and required-field gating.

Vision-Guided Refurb Estimation

Input: photos (kitchen, bathroom, bedroom, lounge, hallway, exterior).
Pipeline:

1. **Room classification** (CV cues + prompt heuristics).
2. **Condition extraction** (cabinet age, tiles, damp marks, flooring wear).
3. **Bill-of-quantities map** (replace cabinets, re-tile, rewire room, paint, flooring).
4. **Regional cost index** by postcode; labour/material splits.
5. **Confidence scoring** and **contingency** policy (e.g., 10–15%).

Rent Estimation (Hybrid)

Regression prior on postcode + features (*beds, type, EPC, finish proxies*) blended with LLM adjustment from listing signals:

$$\hat{R} = \alpha R_{\text{prior}} + (1-\alpha) R_{\text{LLM}}, \quad \alpha \in [0.5, 0.8] \text{ by region}$$

Band and confidence reflect historical dispersion and text cues.

EPC Matching (Heuristic + Fallback)

Normalise strings; extract house number & street tokens; filter exact postcode; score candidates with house# match, Jaccard of street tokens, and edit distance; threshold or null. Fallback LLM comparison if borderline.

Financials & Mortgage

Given price P, refurb C_r, other fees F, annual rent R_a, expense ratio e, mgmt μ:

$$\text{NOI} = R_a(1 - e - \mu), \quad \text{Yield} = \frac{\text{NOI}}{P + C_r + F}$$

Mortgage cashflow with amortisation or IO; ICR/DSCR computed against stressed rates; sensitivity tables across rates and voids.

Uplift & Forecasting

Baseline HPI growth and comparable uplift Δ:

$$P' = (P + C_r)(1 + g_{\text{HPI}}) + \Delta$$

Scenarios sweep g_{HPI}, rent, and refurb to produce tornado/range outputs in Excel/PDF.

Prompts & Guardrails

Strict JSON Output Guard (Global)

```
System: Output only VALID JSON. No markdown, no comments. If
   unsure, use null.
User: If any schema key is missing, include it with null. Prefer
   empty arrays to missing arrays.
```

HTML-to-JSON

```
System: You are a strict data extractor. If a field is unknown,
   return null.
User: Given raw HTML and images, return ONLY:
{price_gbp, address, postcode, bedrooms, bathrooms, tenure, agent
   , features[]}
Do not invent values. If ambiguous, set null and add "warnings
   ":[...].
```

Refurb (Vision)

```
System: Estimate refurbishment line-items from photos using UK
   trade assumptions.
User: For each image: classify room; list items {category, qty,
   unit, unit_rate_gbp, subtotal_gbp}.
Adjust for postcode cost index if provided. Return JSON only with
    a "confidence" 0..1.
```

Rent

```
System: Estimate monthly rent (GBP) given features + listing text
    cues.
User: Return:
{ monthly_rent_gbp, low, high, rationale, confidence }
Do not exceed local comparables by >15% without explicit
   justification.
```

EPC Fallback

```
System: Match a single EPC record to title+postcode and return
   exactly one JSON object.
User: Inputs: propertyTitle, postcode, epcRecords[]. If unsure,
   return nulls (no guess).
```

Evaluation & Accuracy

Metrics

- **Rent MAE/MAPE** vs let-agreed.
- **Refurb error** vs invoices or QS estimates (room bundle and total).
- **EPC precision/recall** on labelled set.
- **Deal error** in Year-1 net yield and cashflow.

Targets (Beta)

- Rent MAPE: 10–15% (1–2 bed flats); 15–20% mixed stock.
- Refurb: 20–30% total; 15–25% with >10 images.
- EPC: >95% precision on clean data; >90% otherwise.
- Net yield error: ±(1–1.5)pp for standard BTL scenarios.

Protocol

Hold-out by region & type; backtest against outcomes; publish aggregate, regional, and dispersion stats; monitor monthly drift.

Security, Compliance, and Data Governance

GDPR

Lawful bases (Legitimate Interests/Contract); data minimisation (only needed fields); opt-in flows for outreach; DPA on request; DPIA maintained.

Credentials & Secrets

Env vars/secret managers; never in code or reports; rotated; least-privilege access; redaction in logs.

PII & Retention

Default retention 12 months (configurable); right-to-erasure; access logging; subject access workflows.

Performance, Reliability & Cost

Latency Budgets

- End-to-end target: 30–90s per property (image count dependent).
- HTML parse < 3s; image loop < 40s (10–15 images); EPC/API < 5s; PDF/Excel < 7s.

COGS (Illustrative)

LLM tokens across parse/images/rent/EPC: £0.05–£0.20; storage pennies per artifact; typical run £0.10–£0.40 (model/volume dependent). Caching, heuristics, and smaller models reduce cost at scale.

Reliability

Retries with backoff; circuit breakers on third-party APIs; idempotent writes; run journaling; dead-letter queues; error budgets and SLOs.

Commercial Strategy

Pricing (Indicative)

- **Starter:** £29/mo, 50 runs; £1/run overage.
- **Pro:** £99/mo, 300 runs; team seats; alerts; branded outputs.
- **Team/Agency:** £299+/mo, 1000 runs; SSO; white-label; priority support.
- **Custom Regions:** one-off setup + premium monthly; accuracy guarantees in coverage zones.

GTM

Beta cohort (10–20 users) with discounted lifetime tiers; partnerships with sourcing groups & agents; thought leadership (accuracy dashboards, regional insights); demo tours and case studies.

Unit Economics

Gross margin >80% at scale; levers: prompt efficiency, model choice, deduped runs, batch scoring, regional fine-tunes.

Operational Playbooks

Incident Response

Severity P1 (platform down), P2 (module degraded), P3 (single-integration issue). On-call rotation; status page; user comms templates; post-mortems within 72h.

Monitoring & Observability

Run-level tracing (IDs across scrape/AI/report); latency histograms; token usage; error rates by node; alerting thresholds per integration.

Data Quality Gates

Schema validators; required-field gates; unit/currency checks; postcode validators; numerical bounds; *warnings[]* propagated into UI/report.

Mortgage Module (Now & Next)

MVP

Standard assumptions (rates, LTV, product fee) included in cashflow and ROI; IO vs repayment switch; ICR check.

Next

Product matrices; ERC modelling; fee amortisation; forward curve scenarios; remortgage window alerts based on LTV and rate environment; DSCR/ICR charting in dashboard.

Deal Discovery & Off-Market

Watchers

New listings, price changes, back-on-market; filters across location/price/bed/yield/refurb-budget; digest emails/Slack.

Off-Market Assistant

GDPR-compliant outreach templates; agent cadence trackers; probate/distress signals where available; lightweight CRM to log replies and next actions.

Roadmap (Detailed)

Weeks 1–6 (MVP Polishing)

Week	Objectives
1	Harden scraper; HTML→JSON prompt v2; finalise canonical schemas; seed postcode priors.
2	Vision refurb v2 (better room detection); regional cost indices; contingency policy.
3	Rent hybrid v2; band confidence; export PDF/Excel v1.
4	EPC matcher v2; comps sanity checks; warnings pipeline; Slack/email alert polish.
5	Dashboard polish; scenario inputs; share links; onboarding walkthrough.
6	Stress tests; logging/observability; beta onboarding and support playbooks.

Launch (v1)

Unified search; API connectors (portals/auctions); de-dup & canonicalisation; batch imports; saved searches; error dashboards; accuracy baselines published.

Q2

Watchers and off-market module; smarter comps; regional calibration; push rent-band hit-rate; reduce refurb variance; client-shareable read-only dashboards.

Q3

Mortgage types & forward curves; live scenario dashboard; remortgage assistant; DSCR/ICR visuals; accuracy bands shrink with feedback.

Q4

White-label/embeddable widgets; team workflows; advanced portfolio analytics; acquisition-readiness hardening.

Risk Register & Mitigations

Risk	Description	Mitigation
Data drift	Rents/refurb costs shift regionally.	Monthly backtests; retrain priors; user feedback weighting.
API fragility	Portal/third-party API changes.	Abstraction layer; fallback scrapers; monitoring & alerts.
LLM variability	Output changes across versions.	Strict schemas; retries; guardrails; cache; unit tests on prompts.
Compliance	Outreach & data usage.	GDPR DPIA; opt-in; audit logs; DPA; rate-limiting.

COGS creep	Token costs increase.	Smaller models; heuristics; caching; batch operations.

Case Study (Illustrative)

Deal Summary

Guide price £525k; 2-bed flat; W2 postcode. Refurb: kitchen refresh, bathroom mid-grade, redecoration. Rent estimate £2,650/mo (low £2,450, high £2,850).

Outputs

Year-1 net yield 4.9% (no leverage); leveraged ROI 6.8%; uplift +6.5 %; EPC C; PDF with breakdowns; Slack alert to buyer and agent.

Support, SLAs, and Operations

Beta Support

Email/Slack within 24h; incident comms; weekly build notes; in-product feedback capture; per-user accuracy scorecards.

SLA Targets (Post-Beta)

99.5% uptime; P1 < 4h mitigate; P2 < 1 business day; monthly availability reports; RCA on P1/P2.

Conclusion

PropVisions delivers a transparent, automation-first approach to deal analysis: fast ingestion, structured outputs, editable assumptions, and continuous monitoring. With disciplined execution, the platform aims to be the operating system for UK residential investors, sourcing agents, and partner estate agencies.

Appendix A: Supabase SQL DDL (Selected)

```sql
-- Note: tighten types/constraints per environment.
create table public.properties (
  id uuid primary key default gen_random_uuid(),
  url text not null,
  address_full text,
  postcode text,
  source text,
  agent jsonb,
  first_seen timestamptz default now(),
  last_seen timestamptz default now(),
  status text default 'active'
);

create table public.analyses (
  id uuid primary key default gen_random_uuid(),
  property_id uuid references public.properties(id) on delete
     cascade,
  run_timestamp timestamptz default now(),
  price numeric,
  layout jsonb,
  assumptions jsonb not null default '{}',
  results jsonb not null default '{}',
  confidence jsonb not null default '{}'
);

create table public.refurb_estimates (
  id uuid primary key default gen_random_uuid(),
  analysis_id uuid references public.analyses(id) on delete
     cascade,
  room_type text,
  items jsonb not null,
  subtotal numeric,
  contingency numeric,
  confidence numeric
);

create table public.rent_estimates (
  id uuid primary key default gen_random_uuid(),
  analysis_id uuid references public.analyses(id) on delete
     cascade,
  monthly_rent numeric,
  band_low numeric,
  band_high numeric,
  method text,
  comps jsonb,
  confidence numeric,
  rationale text
);

create table public.epc_records (
  id uuid primary key default gen_random_uuid(),
  analysis_id uuid references public.analyses(id) on delete
     cascade,
  rating text,
  uprn text,
  lmk_key text,
  measures jsonb,
  full_record jsonb
);
```

```
create table public.reports (
  id uuid primary key default gen_random_uuid(),
  analysis_id uuid references public.analyses(id) on delete
     cascade,
  type text check (type in ('pdf','xlsx')),
  storage_path text not null,
  size_bytes bigint,
  checksum text,
  created_at timestamptz default now()
);
```

Appendix B: n8n Workflow (Pseudo-YAML)

```
nodes:
  - Webhook: trigger { url, userId }
  - HTTP: fetch HTML (retries/backoff)
  - Code: sanitize & canonicalize URL
  - Agent: clean HTML -> JSON (price, address, features)
  - Loop: images[]
      - Agent: refurb estimate per image
  - Code: aggregate refurb lines -> totals + contingency
  - Agent: rent estimate (priors + listing signals)
  - Code: fees, mgmt, voids, maintenance, ROI
  - HTTP: HPI fetch -> uplift baseline
  - HTTP: EPC API -> candidate records
  - Agent: EPC match -> rating + record
  - PDF: compose report
  - Excel: compose spreadsheet
  - Supabase: upsert all tables
  - Slack: send summary + links
```

Appendix C: Report Structure (PDF/Excel)

PDF: Cover → KPI summary → Financials → Refurb (room items & totals) → Rent & comps → EPC → Forecasts → Assumptions & Warnings → Disclaimer.
Excel: Tabs: Summary, Inputs, Refurb, Rent, Fees, EPC, Forecasts, Comps, Audit.

Appendix D: API Stubs (FastAPI)

```
from fastapi import FastAPI, HTTPException
from pydantic import BaseModel
import uuid

app = FastAPI()

class RunRequest(BaseModel):
    url: str
    user_id: str

class RunResponse(BaseModel):
    job_id: str
    status: str

@app.post("/analyze", response_model=RunResponse)
def analyze(req: RunRequest):
    job_id = str(uuid.uuid4())
    # enqueue n8n webhook here
    return RunResponse(job_id=job_id, status="queued")
```

Appendix E: Prompt Library (Copy-Paste Ready)

Strict JSON Output Guard

```
System: Output only VALID JSON. No markdown, no comments. If unsure, use null.
If any schema key is missing, add it with null.
```

EPC Match (Final)

```
System: You are a deterministic matcher. Do not guess.
User: Inputs: { "propertyTitle": "...", "postcode": "...", "epcRecords": [...] }
Rules: normalise strings; exact postcode match required; penalise house-number mismatch;
if none cross a confidence threshold, return matched_epc: null, epc_rating: null.
Output schema:
{
  "property_title": "...",
  "postcode": "...",
  "matched_epc": { ... } | null,
  "epc_rating": "A"|"B"|...|null
}
```

Appendix F: Glossary

- **EPC:** Energy Performance Certificate.
- **HPI:** House Price Index.
- **ICR/DSCR:** Interest Coverage / Debt Service Coverage Ratios.
- **MAPE:** Mean Absolute Percentage Error.

- **NOI:** Net Operating Income.