ContractIQ — Product Requirements Document

Date: June 24, 2026 **Author:** Product Team **Status:** Draft **Version:** 1.0 **Contract Types in Scope:** Non-Disclosure Agreements (NDA), Master Service Agreements (MSA)

Table of Contents

---

1. Problem

What problem is this solving?

Business professionals — founders, operations managers, and procurement leads — routinely sign NDAs and MSAs without fully understanding what they are agreeing to. Without in-house legal teams, reviewing a single contract takes an average of 90–120 minutes, requires legal expertise most SMBs don't have, and frequently results in missed obligations, unfavourable terms, or costly disputes. Existing tools either require legal training or produce generic summaries that don't surface the specific clauses that matter most to each business.

ContractIQ solves this by automatically extracting the key terms from any NDA or MSA, telling the user exactly where each term lives in the document, how confident the extraction is, and allowing them to ask follow-up questions about the contract in plain English — all without needing a lawyer on call.

Why is this problem worth solving?

Quantified pain:

- The global legal tech market was valued at $25.9B in 2023 and is growing at 9.1% CAGR (Grand View Research, 2023 — assumed).
- SMBs spend an average of $1,500–$3,000 for a single lawyer-reviewed contract (assumption based on US market legal billing rates of $250–$500/hr).
- 43% of SMBs report having experienced a commercial dispute stemming from contract misunderstanding (Law Society of England & Wales, 2023 — assumed).
- The Contract Lifecycle Management (CLM) software market alone is projected to reach $4.1B by 2027 (assumed, Markets and Markets).

Market gap:

Existing tools (DocuSign CLM, Ironclad, Kira Systems) are designed for enterprise legal teams with $50k–$500k annual contracts. ChatGPT and generic AI assistants lack structured extraction, page-level attribution, confidence scoring, and contract-type-specific term libraries. There is no affordable, focused tool purpose-built for SMBs reviewing NDA and MSA contracts with a streamlined upload-extract-chat workflow.

MOAT:

ContractIQ's defensibility comes from four sources: 1. **Contract-type specificity:** The prompt library and key-term schema are purpose-built for NDA and MSA structures. Generic LLMs extract "something" from contracts; ContractIQ extracts the 20–30 terms that actually matter for each contract type with page-level attribution. 2. **Feedback-driven improvement:** Every user correction to an extracted term is logged (opt-in, anonymised) and used to improve prompt quality over time — creating a proprietary improvement loop. 3. **Confidence scoring with transparency:** Showing a confidence score per extracted term teaches users what to scrutinise, building trust and reducing the "black box" problem that keeps lawyers in the loop. 4. **Chat-grounded-in-document:** The Q&A feature strictly answers from the uploaded contract text, not from general knowledge — reducing hallucination risk that undermines trust in generic AI chat tools.

Why Agentic AI?

Dimension	Detail
What unstructured data is involved?	Free-form legal prose across NDAs and MSAs with no standardised structure. A "Governing Law" clause may appear in one contract as a numbered section on page 2 and in another as an inline sentence buried in Section 11.4
Why rule-based systems fail	Regex and keyword matching cannot handle clause variant diversity. A confidentiality obligation may be phrased in 50+ structurally different ways across different law firms and geographies. Rule-based parsers produce >30% miss rate on real-world contracts
Why LLMs are necessary	GPT-4o can read legal prose in context, reason about what a clause means, identify the value of a term (e.g. "36 months" as the notice period), and attribute it to a page number — all in a single inference pass
Why not just ChatGPT?	ChatGPT gives unstructured summaries with no page reference, no confidence score, no schema, and no ability to add custom terms to the extraction schema. ContractIQ wraps OpenAI in a structured workflow that produces machine-readable, auditable, user-correctable output

2. User

Who are you solving this problem for?

Primary persona — The Time-Pressed Founder / Ops Lead

- **Industry:** SaaS, agency, professional services, fintech, e-commerce
- **Role:** Founder, COO, Procurement Manager, Legal Operations Manager
- **Company size:** 5–250 employees; no in-house legal counsel
- **Behaviour:** Signs 5–15 NDAs or MSAs per month; relies on Google searches or expensive ad-hoc legal consultations to understand contract terms
- **Pain:** Spends 90–120 minutes per contract review; frequently misses key obligations (auto-renewal clauses, indemnification limits, IP assignment); pays $250–$500/hr for a lawyer to review something that feels routine

Secondary persona — The Freelancer / Consultant

- **Industry:** Design, marketing, software development, consulting
- **Role:** Individual contributor signing client contracts
- **Behaviour:** Receives 1–4 MSAs per month from larger clients; often signs without reading carefully because the power imbalance discourages pushback
- **Pain:** Cannot afford legal review; unsure which clauses are non-standard or risky; no tool gives page-level references with confidence scores

---

3. Core Metrics, Prioritisation & Roadmap

How will you know the problem is solved? (Core Metrics)

North Star Metric: Average time from contract upload to completed key-term review - **Baseline:** 90 minutes (manual review, no tool) - **Target:** ≤ 15 minutes end-to-end within ContractIQ - **Tracked via:** Session logs (upload timestamp → user marks review complete or last interaction timestamp)

Primary Metrics:

Metric	Baseline	Target	How tracked
Key-term extraction accuracy (F1)	0% (no tool baseline)	≥ 88% F1 on NDA/MSA test set	Offline eval suite against ground-truth labelled contracts
Confidence score calibration	—	Predicted confidence within ±10% of actual accuracy	Calibration curve computed on eval set
Time to first extracted key-term display	—	≤ 30 seconds P95 for contracts ≤ 20 pages	Server-side timing logs

Secondary Metrics:

Metric	Baseline	Target	How tracked
30-day user retention	—	≥ 45%	Supabase session analytics
NPS (Net Promoter Score)	—	≥ 40	In-app feedback survey at session end
Contracts processed per active user per month	—	≥ 4	Dashboard analytics query on contracts table
AI extraction correction rate	—	≤ 12% of terms manually corrected by users	corrections_count / total_extracted_terms per session
Cost per contract analysis (OpenAI tokens)	—	≤ $0.25 per analysis (20-page contract)	Billing logs from OpenAI usage dashboard

Prioritisation

Breaking the Agentic Workflow into Components

1. **Component A — User Authentication & Session Management** (Supabase Auth)
2. **Component B — PDF Upload & Text Extraction** (Supabase Storage + pdf-parse)
3. **Component C — Key Term Extraction via OpenAI** (GPT-4o structured output)
4. **Component D — Custom Term Addition** (user input → appended to extraction prompt)
5. **Component E — Results Display (PDF Viewer + Key Terms Panel)** (PDF.js + React)
6. **Component F — Contract Chat (Q&A)** (GPT-4o RAG-style with full contract context)
7. **Component G — Dashboard & History** (Supabase queries on contracts + key_terms tables)
8. **Component H — Feedback Collection** (Supabase write to user_feedback table)

Risk Assessment per Component

Component	Check	Result	Explanation
B — PDF Text Extraction	Is ML necessary?	FAIL	Rule-based PDF parsing (pdf-parse) is sufficient for text-layer PDFs; ML (OCR) only needed for scanned PDFs, which are out of scope
B — PDF Text Extraction	Do we have data?	PASS	pdf-parse handles standard PDF text layers without training data
B — PDF Text Extraction	Can it meet accuracy requirements?	PARTIAL	≥ 95% text fidelity for text-layer PDFs; scanned PDFs will fail gracefully
B — PDF Text Extraction	What about bias?	PASS	Pure text extraction; no model bias risk

Component	Check	Result	Explanation
B — PDF Text Extraction	**Architecture note**	—	Text is extracted **once at upload**, stored in `contracts.contract_text` with `[PAGE N]` markers. The processing pipeline and chat route both read from the DB — neither downloads the PDF again. Supabase Storage is used only for the inline PDF viewer (non-blocking: if Storage is unavailable, only the viewer is hidden; the AI pipeline is unaffected)
C — Key Term Extraction	Is ML necessary?	PASS	Clause variant diversity across law firms and geographies makes rule-based extraction untenable
C — Key Term Extraction	Do we have data to train?	PARTIAL	No proprietary labelled dataset at launch; relying on CUAD dataset + few-shot prompting. Fine-tuning planned for v2
C — Key Term Extraction	Can it be solved by AI?	PASS	GPT-4o demonstrated strong NDA/MSA extraction in internal testing (assumed)
C — Key Term Extraction	Can it meet accuracy requirements?	PARTIAL	Targeting ≥ 88% F1; current zero-shot is ~76% F1 (assumed baseline); few-shot prompt expected to close the gap
C — Key Term Extraction	Can it scale?	PASS	OpenAI API scales horizontally; rate limits manageable at projected early usage
C — Key Term Extraction	How fast can we get feedback?	PASS	User corrections are logged immediately; weekly review cycle planned
C — Key Term Extraction	What are the laws?	PARTIAL	GDPR compliance required; no PII used in prompts beyond the contract itself; legal review of data processing needed before EU launch
C — Key Term Extraction	What about bias?	PARTIAL	Model may perform worse on non-US/non-UK contract conventions; in-scope for MVP is English-language contracts only

Component	Check	Result	Explanation
C — Key Term Extraction	How transparent/explainable?	PASS	Source sentence shown per term; confidence score shown; user can view the raw text
C — Key Term Extraction	How easy to judge good vs bad?	PASS	Ground truth is the contract text itself; user corrections provide direct signal
F — Contract Chat	Is ML necessary?	PASS	Natural language Q&A over unstructured legal text requires LLM understanding
F — Contract Chat	Can it meet accuracy requirements?	PARTIAL	Grounding via full contract context reduces hallucination; risk remains for ambiguous or multi-part clauses
F — Contract Chat	What about bias?	PARTIAL	Model may be overconfident on unfamiliar contract types; confidence is not surfaced in chat (only in extraction) — mitigation: add “Based on the document…” prefix to all responses
F — Contract Chat	How transparent/explainable?	PASS	Mandatory page citation on every response; system prompt forbids using general knowledge

Overall Risk Summary

Component	Risk Level	Mitigation
B — PDF Text Extraction	Low	Reject scanned PDFs gracefully with clear user message; support text-layer PDFs only at MVP
C — Key Term Extraction	Medium	Few-shot prompting + offline eval suite + user correction feedback loop; confidence scoring gives user control
D — Custom Term Addition	Low	User-defined terms are injected into a well-tested prompt schema; limited to 5 terms at MVP to manage context length
E — Results Display	Low	PDF.js is a mature, battle-tested library; lazy page loading mitigates large file rendering issues
F — Contract Chat	Medium-High	System prompt strictly limits responses to document text; page citation enforced; “I cannot find this in the document” fallback added

Component	Risk Level	Mitigation
G — Dashboard & History	Low	Simple Supabase queries with indexed user_id; no AI involvement
H — Feedback Collection	Low	Simple form-to-database write; no AI involvement

Prioritised Stories (MVP Scope)

Story ID	Title	Priority	Points	Status
US-001	User authentication (sign up / sign in / sign out)	P0	3	To Do
US-002	PDF upload + text extraction	P0	5	To Do
US-004	Confidence score display per term	P0	3	To Do
US-005	Custom key term addition before processing	P0	4	To Do
US-003	Page number attribution per key term	P0	3	To Do
US-011-partial	Key terms panel with value display	P0	5	To Do
US-006	Inline PDF viewer in results page	P1	5	To Do
US-007	Chat with contract	P1	8	To Do
US-012	Persistent chat history per contract	P1	3	To Do
US-008	Dashboard with contract history	P1	5	To Do
US-009	Inline key term editing	P1	3	To Do
US-010	Feedback rating submission	P2	2	Backlog
US-011	Export key terms to CSV / PDF	P2	4	Backlog

Roadmap

Release	Features Included	Duration
v0.1 — Foundation (MEP)	Supabase project setup + all DB tables; Landing page (static, no dynamic content); Email/password auth; Redirect to Dashboard on login; Empty dashboard state	Weeks 1–2
v0.2 — Core Review Flow	PDF upload screen with contract type selector; pdf-parse text extraction; OpenAI key term extraction (standard terms, NDA + MSA prompt); Key terms panel (name, value, page, confidence); Confidence warning on low-score terms; Results stored in Supabase	Weeks 3–5
v0.3 — Enriched Experience	Pre-processing preview of key terms to be fetched; Custom key term addition (up to 5 terms); Inline PDF viewer (PDF.js); Click-to-navigate from key term panel to page in PDF viewer; Source sentence expandable tooltip	Weeks 6–8
v0.4 — Chat & History	Contract chat interface (full contract context passed to GPT-4o); Persistent chat session storage in Supabase; Dashboard populated with contract history (sortable list); Inline key term editing with edit badge; Error states for upload failures and OpenAI timeouts	Weeks 9–11
v1.0 — Launch	Feedback submission (thumbs up/down + comment); End-to-end performance optimisation (target: ≤ 30s P95); Security audit (RLS policies, signed URL expiry, API key management); WCAG 2.1 AA review; Rate limiting on OpenAI calls; Onboarding tooltips for first-time users	Weeks 12–14
v1.1 — Post-Launch Iteration	Export key terms to CSV; Export results summary to PDF; Batch contract upload (up to 5 contracts); Dashboard analytics (charts: contracts by month, term correction rate)	Weeks 15–18

Release	Features Included	Duration
v1.2 — Growth	Scanned PDF support via OCR (AWS Textract or equivalent); Contract comparison view (side-by-side key terms across 2 contracts); Email notifications on processing completion; Multi-user workspace (team plans)	Weeks 19–24

Dependencies:

- OpenAI API access and approved usage terms (required before v0.2 ships)
- Supabase project provisioned with Pro plan for production (required before v1.0)
- Legal review of terms of service and data processing agreement (required before v1.0 public launch)
- GDPR DPA with OpenAI confirmed before EU user onboarding

External Dependencies:

Dependency	Risk	Mitigation
OpenAI API availability	OpenAI outages directly block contract processing	Implement retry with exponential backoff (3 attempts); surface human-readable error to user with “Try again in a few minutes” CTA
OpenAI pricing changes	Token cost increases could push per-analysis cost above $0.25 threshold	Monitor usage monthly; maintain cost alerting at 80% of budget threshold; evaluate Claude or Gemini as fallback if cost doubles
Supabase free tier limits	500 MB DB + 1 GB storage on free tier; will breach at ~200 contracts	Migrate to Supabase Pro ($25/month) before beta launch; alert at 70% storage usage
PDF.js rendering compatibility	Complex PDFs with unusual fonts or layouts may not render correctly	Test against 50 real-world NDAs and MSAs during beta; provide a “download PDF” fallback link if rendering fails
Browser file API limits	Large PDFs (near 10 MB) may cause browser memory issues on low-end devices	Enforce 10 MB server-side limit; recommend Chrome/Firefox on desktop; warn mobile users

Internal Risks:

Risk	Likelihood	Impact	Mitigation
OpenAI extraction accuracy below 88% F1 target	Medium	High	Few-shot prompt tuning + offline eval suite; lower launch threshold to 82% F1 with transparency to users (confidence scores visible)

Risk	Likelihood	Impact	Mitigation
Users uploading non-NDA/MSA contracts	High	Low	Soft warning if contract type detection doesn't match user's selection; graceful degradation (still extracts, just may miss domain-specific terms)
Chat hallucination (AI answers from general knowledge, not document)	Medium	High	System prompt enforces document-only answers; automated test: feed a question about a topic not in the document, expect "I cannot find this" response
Supabase RLS misconfiguration exposing user data	Low	Critical	Pre-launch security review: attempt cross-user data access from test accounts; RLS unit tests in CI pipeline

---

4. MVP Features

User Flows

Flow 1 — New Visitor → Sign Up → Dashboard

```
Landing Page → Click "Sign Up" → Supabase Auth (Email/Password)
→ Email Verification → Redirect to Dashboard
```

1. User lands on the ContractIQ marketing page; sees product value prop, a short demo GIF, and two CTAs: "Sign In" and "Get Started Free"
2. Clicking "Get Started Free" opens the Supabase Auth sign-up modal (email + password)
3. On successful registration, user is redirected to the Dashboard
4. Dashboard greets user with an empty state: "No contracts reviewed yet — upload your first contract to begin"

Flow 2 — Returning User → Dashboard

```
Sign In → Supabase Auth → Dashboard
```

1. User sees a summary card: total contracts processed, contracts by type (NDA / MSA), last 5 contracts reviewed with status and date
2. Quick-action button: "Review a Contract" is prominently placed

Flow 3 — Core Flow: Contract Review

```
Click "Review Contract" → Choose Contract Type (NDA / MSA) → Upload PDF
→ PDF Text Extraction → Key Term Preview → Add Custom Terms (optional)
→ Click "Process Contract" → OpenAI Extraction → Results Page
→ Contract Preview + Key Term Panel + Chat
```

Detailed steps:

1. **Upload screen:** User selects contract type (NDA or MSA) from a dropdown, then drags and drops or file-picks a PDF (max 20 pages / 10 MB for MVP)
2. **Pre-processing preview:** While the PDF text is being extracted, the UI shows a preview card listing the standard key terms ContractIQ will look for based on the selected contract type. For NDAs this includes: Parties, Effective Date, Confidentiality Obligations, Permitted Disclosures, Term & Duration, Governing Law, Jurisdiction, IP Ownership, Non-Solicitation, Breach & Remedy. For MSAs: Parties, Service Scope, Payment Terms, Invoice Schedule, Late Payment Penalty, Liability Cap, Indemnification, IP Ownership, Termination Clause, Governing Law, Dispute Resolution, Notice Period
3. **Custom term addition:** A clearly visible "+ Add Key Term" button allows the user to type a custom term they want extracted (e.g. "Non-compete radius") before processing begins. Added terms appear in the preview list with a "Custom" badge
4. **Process trigger:** User clicks "Process Contract" — a progress indicator appears (step 1: extracting text, step 2: analysing with AI, step 3: compiling results)
5. **Results page:** Displayed in a two-panel layout:
    - **Left panel:** Interactive PDF viewer (scrollable, zoomable) with page numbers and highlighted spans for extracted terms
    - **Right panel:** Key terms list, each showing Term Name | Extracted Value | Page Number | Confidence Score (colour-coded: green ≥ 80%, amber 50–79%, red < 50%)
6. **Manual correction:** User can click any extracted term to edit its value inline — correction is saved to Supabase and flagged for feedback loop
7. **Chat button:** A floating "Chat with Contract" button (or sidebar tab) opens the chat interface within the same view. All chat messages are saved to Supabase and linked to the contract session
8. **Hallucination safeguard:** If confidence score < 50%, the term value is shown with a ⚠️ flag and a tooltip: "Low confidence — we recommend verifying this in the document directly." The PDF viewer auto-highlights the nearest matching page span
9. **Explainability:** Each extracted term has an expandable "Why?" section showing the verbatim sentence from the contract that the AI used to extract the value

Flow 4 — Chat with Contract

```
Results Page → Click "Chat" Tab → Type Question → OpenAI Response (grounded in contract text)
→ Conversation logged to Supabase
```

1. User types a question such as "What happens if I breach the NDA?" or "Is there an auto-renewal clause?"

2. The backend passes the full extracted contract text + conversation history to OpenAI with a system prompt instructing it to answer only from the provided document text
3. The response appears in a chat UI (user messages right-aligned, AI responses left-aligned)
4. Each AI response includes a “Source: Page X” citation linking to the relevant page in the PDF viewer
5. Conversation is saved in `chat_messages` table, linked to `chat_sessions`, linked to the contract

Functional Requirements

User Stories:

ID	User Story	Acceptance Criteria	Priority
US-001	As a founder, I want to sign up with my email and password so that my contracts and chat history are saved privately	Auth flow completes within 10 seconds; user is redirected to Dashboard on success; invalid credentials return a clear error message	P0
US-002	As a user, I want to upload a PDF contract and see the key terms extracted automatically so that I don’t have to read the whole document line by line	PDF upload accepts files up to 10 MB; extraction completes within 30 seconds P95 for ≤ 20 pages; key terms panel shows ≥ 80% of standard NDA/MSA terms with values	P0
US-003	As a user, I want to see which page each key term was found on so that I can verify the extraction myself	Each extracted term displays a page number; clicking the page number scrolls the PDF viewer to that page	P0
US-004	As a user, I want to see a confidence score for each extracted term so that I know which terms I should verify manually	Each term shows a confidence score (0–100%); scores < 50% show a warning icon and tooltip	P0
US-005	As a user, I want to add a custom key term before processing so that I can get values for clauses specific to my situation	Custom terms appear in the pre-processing preview; processed results include custom term extraction with the same structure (value, page, confidence)	P0
US-006	As a user, I want to see a preview of the PDF within the app so that I don’t have to switch between windows while reviewing	PDF viewer renders all pages; user can scroll, zoom in/out; highlighted term references are clickable	P1

ID	User Story	Acceptance Criteria	Priority
US-007	As a user, I want to chat with my contract in plain English so that I can ask specific questions without searching manually	Chat responds within 15 seconds; responses are grounded in the uploaded document text; each response cites a page number	P1
US-008	As a user, I want my dashboard to show all the contracts I've reviewed so that I have a record of my review history	Dashboard displays contract name, type, date uploaded, and review status; clicking any row opens the results page for that contract	P1
US-009	As a user, I want to edit an incorrectly extracted term so that the record is accurate	Inline edit saves to Supabase within 2 seconds; edited terms display an "Edited" badge; original AI value is stored separately for feedback purposes	P1
US-010	As a user, I want to submit feedback on the AI's accuracy so that the product improves over time	A thumbs up / thumbs down rating and optional text comment is available on the results page; feedback is saved to `user_feedback` table	P2
US-011	As a user, I want to export the key terms as a CSV or PDF report so that I can share the review summary with my team	Export button generates a formatted file within 5 seconds and downloads to the browser	P2
US-012	As a user, I want the chat history for each contract to persist so that I can revisit my questions later	Chat messages are stored in Supabase; reopening a contract's results page loads the previous chat session	P1

Functional Requirements Table:

ID	Requirement	Priority	Notes
FR-01	Users must be able to sign up, sign in, and sign out using Supabase Auth (email/password)	P0	Auth state persists via Supabase session tokens stored in browser
FR-02	The system must accept PDF uploads up to 10 MB and 20 pages; reject files outside these limits with a clear error	P0	Only NDA and MSA documents in scope for MVP

ID	Requirement	Priority	Notes
FR-03	The system must extract text from the uploaded PDF at upload time, store it in `contracts.contract_text`, and reuse it for AI extraction and chat — without re-downloading the PDF	P0	Text extraction (with `[PAGE N]` markers) happens server-side during upload; the extracted text is stored in the DB so that the processing and chat pipelines never depend on Supabase Storage being available
FR-04	The key terms panel must display: Term Name, Extracted Value, Page Number, Confidence Score (%) for each term	P0	Page numbers are 1-indexed; confidence score is returned from prompt and validated
FR-05	Users must be able to add at least 5 custom key terms before processing; these must appear in results with the same data structure as standard terms	P0	Custom terms stored in `custom_key_terms` table with `is_manual = true` flag
FR-06	The results page must always display contract content — either as an interactive PDF viewer (when Storage is available) or as a paginated text viewer fallback (when Storage is unavailable)	P1	Primary: PDF.js viewer using a 1-hour signed URL from Supabase Storage. Fallback: text viewer that parses `[PAGE N]` markers from `contracts.contract_text`, renders each page as a labelled section, and supports the same page-navigation behavior as the PDF viewer. Both viewers must respond to `targetPage` prop changes from key-term click events
FR-07	Clicking a page reference on the key terms panel must scroll the PDF viewer to the corresponding page	P1	Smooth scroll with visual highlight on referenced paragraph
FR-08	The chat interface must send the user's question + full contract text to OpenAI and display the grounded response	P1	System prompt enforces document-only answers; responses include page citation
FR-09	All chat messages must be saved to Supabase in real-time with role (user / assistant) and timestamp	P1	Linked to chat_sessions → contracts → users
FR-10	The Dashboard must show: total contracts reviewed, breakdown by type (NDA / MSA), a sortable list of all previous contracts	P1	Sortable by date, name, type; clickable rows open results
FR-11	Terms with confidence < 50% must show a visual warning and recommend manual verification	P0	⚠️ icon + tooltip; do NOT hide the term — show it with the warning
FR-12	The user must be able to submit a thumbs-up / thumbs-down feedback per contract review with an optional text comment	P2	Stored in `user_feedback` table with user_id, contract_id, rating, comment, timestamp

ID	Requirement	Priority	Notes
FR-13	The system must store all contracts, key terms, sessions, messages, and feedback in a single Supabase project with row-level security	P0	Each table has a `user_id` foreign key; RLS policies ensure users only see their own data
FR-14	The complete database setup — all tables, indexes, triggers, RLS policies, the Supabase Storage bucket, and Storage RLS policies — must be expressible as a single paste-and-run SQL file	P0	Use `INSERT INTO storage.buckets` to create the bucket and `CREATE POLICY ON storage.objects` for Storage RLS; file path pattern is `contracts/{user_id}/{contract_id}/{filename}.pdf`; Storage policies must restrict INSERT/SELECT/DELETE to `auth.uid()::text = (storage.foldername(name))[1]`

Agent Capabilities & System Behaviour

Component	Input	Output	Autonomy Level	Human-in-Loop Trigger
PDF Text Extractor	Uploaded PDF file (binary)	Raw text string with `[PAGE N]` markers stored in `contracts.contract_text`	Fully autonomous	If extracted text < 100 words (likely scanned/image PDF), show error: "Scanned PDFs are not supported yet"; extraction happens once at upload — all downstream features (processing, chat) read from the stored text, not the file
Key Term Extractor	Contract text + contract type + custom term list	JSON: `[{ term_name, value, page_number, confidence_score, source_sentence }]`	Autonomous + review	If confidence < 50% on any term, flag with ⚠️ and recommend manual check
Confidence Evaluator	Extracted term + source sentence	Confidence score 0–100	Autonomous	Always shown to user; no threshold blocks display
Contract Chat Agent	User question + full contract text + conversation history	Plain-English answer with page citation	Suggests, human decides	Always: AI cannot take actions on the contract; it only answers questions
Feedback Logger	User rating + optional comment + contract_id	Stored feedback record	Fully autonomous	—

5. Constraints

Performance constraints:

- P95 end-to-end extraction latency (upload → results displayed) ≤ 30 seconds for contracts up to 20 pages.
- Time to first extracted key-term display ≤ 30 seconds P95 for contracts ≤ 20 pages.
- Chat response latency ≤ 15 seconds P95.
- Each OpenAI call ≤ 20 seconds P95.
- Inline edit saves to Supabase within 2 seconds; export file generation within 5 seconds.
- Auth flow completes within 10 seconds.

Upload & document constraints:

- PDF uploads limited to **10 MB** and **20 pages**; files outside these limits are rejected with a clear error.
- Contract length ≤ 15,000 tokens for MVP; longer contracts rejected with a clear message.
- Only **text-layer PDFs** supported; scanned/image PDFs fail gracefully ("Scanned PDFs are not supported yet"). Trigger: extracted text < 100 words.
- Only **NDA and MSA** English-language contracts (US/UK law) in scope for MVP.
- Maximum **5 custom key terms** per analysis at MVP to manage context length.

Cost constraints:

- Cost per contract analysis ≤ $0.25 per 20-page contract (extraction target ≤ $0.20).

Scalability constraints:

- Must handle 100 concurrent contract analyses without degradation during beta.
- Architecture must support horizontal scaling to 1,000 concurrent users post-launch.

Reliability & security constraints:

- 99.5% uptime SLA. OpenAI API errors must be caught and surfaced with a human-readable message and retry option — no silent failures.
- PDFs optionally stored in Supabase Storage with signed URLs (expiry: 1 hour). Storage upload is non-blocking — failure only hides the PDF viewer; the AI pipeline continues using stored text.
- All data encrypted at rest (AES-256) and in transit (TLS 1.3). Supabase RLS enforced on all tables.

Usability & compliance constraints:

- Usable by a non-lawyer with no onboarding training; all legal jargon tooltipped or explained in plain English. WCAG 2.1 AA compliance.
- Data retention: uploaded PDFs stored for 90 days post last-access, then auto-deleted. Users can manually delete a contract and all associated data at any time.
- GDPR-ready: user data deletion on request; DPA available; no contract content used to train third-party models; OpenAI API configured with `user` parameter and no training opt-in.

6. Technical Requirements

Architecture Overview

The system is built as a single-tenant web application with a React frontend, a lightweight serverless backend (Supabase Edge Functions or a hosted Node.js API), and Supabase as the single backend-as-a-service platform for auth, database, and file storage.

Component layers:

- **Frontend (React SPA):** Handles all user interactions — auth, upload, PDF rendering, key-term panel, chat interface, dashboard. Communicates with Supabase directly for auth and data reads; calls the backend API for OpenAI-heavy operations.
- **Backend API (Node.js / Supabase Edge Functions):** Orchestrates PDF text extraction, OpenAI prompt calls, structured output parsing, and writes results to Supabase. This layer is kept thin — no business logic lives here beyond orchestration.
- **OpenAI API (GPT-4o):** Handles key term extraction (structured JSON output), confidence scoring, and chat Q&A. Called exclusively from the backend — the OpenAI API key is never exposed to the client.
- **Supabase (single project):** Provides Auth, PostgreSQL database (all tables below), Storage (PDF files), and Realtime subscriptions (for chat message streaming).
- **PDF.js:** Client-side PDF rendering library; renders the uploaded PDF inline in the browser using the signed URL returned by Supabase Storage.

Model Requirements

Criteria	Requirement	Rationale
Model	GPT-4o via OpenAI API	Best-in-class reasoning on long legal text; JSON mode support for structured output
Context window	≥ 128k tokens	A 20-page contract ≈ 10,000–15,000 tokens; headroom needed for prompt + history
Response format	JSON mode enabled (`response_format: { type: "json_object" }`)	Eliminates unparseable free-text responses
Max tokens per call	2,000 output tokens for extraction; 1,000 for chat	Extraction output is bounded; chat answers should be concise
Temperature	0.1 for extraction; 0.4 for chat	Low temperature = deterministic structured extraction; slight warmth for natural chat responses
Latency	≤ 20 seconds per OpenAI call P95	Combined with UI/UX loader, total experience target is ≤ 30 seconds

Criteria	Requirement	Rationale
Cost per analysis	≤ $0.20 per 20-page contract (extraction only)	At GPT-4o pricing of $0.005/1k input + $0.015/1k output: ~15,000 input tokens + 1,500 output tokens ≈ $0.097 per analysis

Model Selection & Cost Trade-offs

Item	What we use	Why	Trade-off
LLM for inference	GPT-4o (OpenAI API)	Best performance on legal text understanding; JSON mode; 128k context	Higher cost than GPT-3.5 or Claude Haiku; API dependency
PDF text extraction	pdf-parse (Node.js) / PDF.js	Open-source, no data egress to third parties, handles most text-layer PDFs	Does not handle scanned/image PDFs (out of scope for MVP)
Auth & DB	Supabase	Managed Postgres + Auth + Storage in one platform; generous free tier; RLS support	Vendor lock-in; limited to Postgres dialect
Frontend	React + Tailwind CSS	Team velocity; large ecosystem; PDF.js compatibility	Requires client-side state management for real-time chat
File storage	Supabase Storage	Co-located with DB; signed URLs (1-hour expiry); no extra vendor	Bucket (`contracts`) and RLS policies must be created via SQL before first upload — use `INSERT INTO storage.buckets` and `CREATE POLICY ON storage.objects` ; file path: `contracts/{user_id}/{contract_id}/{filename}.pdf` ; Storage upload is non-blocking (failure only hides PDF viewer)
PDF rendering	PDF.js (client-side)	No server load for rendering; handles page navigation and zoom	Heavy initial load for large PDFs; workaround: lazy-load pages
Hosting	Vercel (frontend) + Supabase Edge Functions (backend)	Zero-config deployments; scales automatically	Cold start latency on Edge Functions (~300ms first call); acceptable given 30s extraction budget

7. Grounding Strategy

ContractIQ's core trust guarantee is that every AI output is **grounded in the uploaded contract text**, never in the model's general legal knowledge.

- **Single source of truth:** PDF text is extracted **once at upload** with `[PAGE N]` markers and stored in `contracts.contract_text`. Both the extraction pipeline and the chat route read from this stored text — the model never sees anything except the user's own document.
- **Extraction grounding:** Every extracted term carries a `source_sentence` — the verbatim sentence from the contract that the value was drawn from — plus a 1-indexed `page_number`. This makes each output traceable back to the exact location in the document (surfaced via the expandable "Why?" section).
- **Chat grounding (RAG-style):** The chat route passes the **full contract text** as context alongside conversation history. The system prompt instructs: *"Answer only from the document text provided. If the answer is not in the document, say so."* Every chat response must include a mandatory `[Page X]` citation.
- **Full-context strategy:** For contracts ≤ 15,000 tokens, the entire document is passed on every turn (no chunking / vector retrieval needed at MVP). This guarantees no relevant clause is missed due to retrieval error. A chunked RAG strategy is deferred until contract length limits are raised post-v1.0.
- **Conversation memory:** The full conversation history (up to 200 messages, ascending) is passed on every turn, enabling memory-style questions ("what did you say earlier about X?"). A query-classification layer (`contract` / `history` / `both`) adjusts the system prompt and context inclusion without an extra API call.
- **"Not found" as a valid answer:** When information is absent from the document, "I cannot find this in the document" is the correct, expected response — not a failure.

8. Prompt Strategy

Task	Technique	Output Format	Rationale
Key term extraction	Few-shot (3 labelled NDA examples, 3 MSA examples in system prompt)	JSON array: `[{ "term_name": string, "value": string, "page_number": int, "confidence_score": float, "source_sentence": string }]`	Consistent structured output across clause variant diversity; few-shot examples ground the model on the expected schema
Confidence scoring	Embedded in extraction prompt — model self-reports a 0.0–1.0 score alongside each term	Float field within the JSON term object	Avoids a second inference call; model reasons about its own certainty while extracting

Task	Technique	Output Format	Rationale
Custom term extraction	Zero-shot with term name injected into the extraction prompt as an additional target	Same JSON schema as standard terms	Custom terms are appended to the standard term list passed to the model
Contract chat (Q&A)	Retrieval-Augmented: full contract text passed as context + conversation history (last 10 turns); system prompt: “Answer only from the document text provided. If the answer is not in the document, say so.”	Free text with a mandatory page citation tag: `[Page X]`	Grounds responses strictly in the uploaded document; prevents hallucination from general legal knowledge
Error recovery	If JSON parse fails, a retry prompt is sent: “Your previous response was not valid JSON. Return only the JSON array, no explanation.”	JSON array	Single automatic retry before surfacing an error to the user

Prompt improvement plan: - Maintain a versioned prompt library (v1.0, v1.1…) in a shared document accessible to the product team - A/B test extraction prompts monthly on a 50-contract offline eval set - Log every user edit to a `term_corrections` view; trigger a prompt review if correction rate exceeds 12% of terms in any 7-day window

---

9. Hallucination Guardrails

ContractIQ treats hallucination — the model asserting something not in the contract — as the top trust risk. Layered guardrails address it at extraction, chat, and UI level.

Extraction-layer guardrails: - **Confidence scoring on every term:** The model self-reports a 0–100% confidence per extracted term. Scores are colour-coded (green ≥ 80%, amber 50–79%, red < 50%). - **Low-confidence flagging:** Terms with confidence < 50% show a ⚠️ warning and a non-dismissible tooltip: *“Low confidence — we recommend verifying this in the document directly.”* The term is **never hidden** — it is shown with the warning so the user retains control. - **Source-sentence requirement:** Every term must carry the verbatim `source_sentence` it was drawn from; a term with no supporting sentence is treated as unreliable. - **Deterministic settings:** Extraction runs at temperature 0.1 with JSON mode enforced to minimise fabrication and unparseable output. - **Calibration monitoring:** Monthly calibration evaluation checks that predicted confidence matches observed accuracy; a UI calibration warning is shown if eval reveals ≥ 15% miscalibration.

Chat-layer guardrails: - **Document-only system prompt:** The model is instructed to answer strictly from the provided contract text and to reply “I cannot find this in the document” when the answer is absent. - **Mandatory page citation:** Every chat response must include a `[Page X]` citation, making claims verifiable against the source. - **“Based on the document…” framing:** Responses are prefixed to remind users the

answer is scoped to their contract, mitigating model overconfidence on unfamiliar contract types. - **Automated hallucination test:** A regression test feeds a question about a topic not present in the document and asserts the model responds “I cannot find this.”

UI / human-in-the-loop guardrails: - **Inline correction:** Users can edit any extracted term; the original AI value is stored separately to feed the improvement loop. - **PDF auto-highlight:** Low-confidence terms auto-highlight the nearest matching page span so the user can verify in one click. - **“Not legal advice” disclaimer:** Present on every results page — *“This is an AI-assisted review tool, not legal advice. Always verify critical terms with a qualified lawyer.”*

---

10. Evaluation Strategy

Ground truth sources

- CUAD (Contract Understanding Atticus Dataset) — 13,000+ annotations across 510 commercial contracts; used for offline baseline evaluation
- 30 manually labelled NDA contracts (internal, annotated by a legal SME before beta)
- 20 manually labelled MSA contracts (internal, annotated by a legal SME before beta)
- User-corrected terms (opt-in, anonymised) — fed into ongoing prompt improvement

Evaluation Plan

Eval Type	Method	Target	Cadence
Key term extraction accuracy	Precision / Recall / F1 against manually labelled test set (30 NDA + 20 MSA)	≥ 88% F1 (NDA); ≥ 85% F1 (MSA)	Every release
Confidence score calibration	Calibration curve: predicted confidence vs. actual accuracy bucketed by 10% intervals	Calibration error ≤ 0.10 (each bucket’s predicted confidence within 10% of observed accuracy)	Monthly
Page number accuracy	% of terms where returned page_number matches ground truth page	≥ 92% correct page attribution	Every release
Custom term extraction accuracy	F1 on a set of 10 predefined custom terms injected into 15 test contracts	≥ 80% F1	Every release
Chat groundedness	Expert review: 50 Q&A pairs from real contracts; score each response as Grounded (from doc) / Hallucinated (from general knowledge) / Not found (correct)	≤ 5% hallucinated responses	Monthly

Eval Type	Method	Target	Cadence
End-to-end latency	P95 timing from upload submission to results panel rendered	≤ 30 seconds	Every release
User satisfaction (beta)	Post-review survey: “Were the extracted terms accurate?” (Yes / Partially / No)	≥ 75% “Yes” in beta	Beta phase

AI Performance Monitoring (Post-Launch)

- Automated regression suite runs on every deploy using the 50-contract labelled test set
- Weekly drift check: sample 10 recent user-corrected terms and compare against expected extraction
- Alert: if correction rate exceeds 12% in any 7-day rolling window, trigger an immediate prompt review
- Monthly: legal SME audits 5 random contracts from production output for quality assurance

Evaluation Spreadsheet

[Link to evaluation spreadsheet — to be created before beta] — Columns: `Contract_ID | Contract_Type | Term_Name | Expected_Value | AI_Extracted_Value | Expected_Page | AI_Page | Confidence_Score | F1_Match | Expert_Rating | Notes`

---

11. Production Readiness Criteria & Metrics (HHH)

HHH Evaluation

Pillar	Strength	Risk	Mitigation
Helpful	Reduces NDA/MSA review from 90 minutes to ≤ 15 minutes; surfaces terms non-lawyers routinely miss; custom terms accommodate specific business needs	Output could overwhelm users with too many terms; low-confidence terms might be acted on without verification	Prioritise the 10–12 most material terms by default; confidence warnings are prominent and non-dismissible
Honest	All extracted values show the verbatim source sentence; confidence score is always displayed; “I cannot find this in the document” is a valid chat response	Confidence scores may be miscalibrated early; model may over-report confidence	Monthly calibration evaluation; show calibration warning in UI if eval shows ≥ 15% miscalibration; disclaimer: “ContractIQ is not legal advice” on every results page

Pillar	Strength	Risk	Mitigation
Harmless	Domain is factual contract text; no toxic or harmful content generated; user corrections prevent erroneous terms from persisting	User acts on an incorrect extraction and signs a contract with unfavourable terms	Prominent disclaimer on every results page: “This is an AI-assisted review tool, not legal advice. Always verify critical terms with a qualified lawyer.”; confidence warnings for any term < 50%

Launch Criteria

Stage	Helpful	Honest	Harmless	Go Criteria
Internal Alpha (team only)	Basic extraction working	Source sentences shown	Disclaimer present	Core upload-extract-display flow works end-to-end without crashes
Measurement Beta (≤ 50 users)	≥ 75% user satisfaction in post-review survey	Correction rate ≤ 20%	0 incidents of misleading output without confidence warning	No P0 bugs; latency ≤ 45s P95; F1 ≥ 82% on eval set
Public Launch	≥ 80% user satisfaction	Correction rate ≤ 12%; calibration error ≤ 0.10	Security audit passed; RLS verified; legal disclaimer approved	F1 ≥ 88% NDA / 85% MSA; latency ≤ 30s P95; Supabase Pro provisioned; DPA with OpenAI confirmed

Responsible AI

Accountability:

Question	Answer
Efficacy & limitations of the product	ContractIQ accurately extracts standard NDA and MSA terms from English-language, text-layer PDFs at ≥ 88% F1. It does not provide legal advice, does not handle scanned PDFs, does not support non-English contracts, and may miss highly unusual or bespoke clauses not present in standard NDA/MSA structures
Compliance policies for sensitive data	Contracts contain commercially sensitive and potentially personal data. All files are stored encrypted at rest in Supabase Storage (AES-256); transferred over TLS 1.3; accessible only via time-limited signed URLs. GDPR Article 28 DPA required with Supabase and OpenAI before EU user onboarding

Question	Answer
How is sensitive data managed	Uploaded PDFs are stored in Supabase Storage for 90 days and then auto-deleted. The plain-text content extracted from the PDF is stored in `contracts.contract_text` in the database (encrypted at rest) so the AI pipeline does not need to re-download the file on every request. Only the text content and structured key term output are persisted — no model training occurs on this data. Users can delete their contracts and all associated data at any time from their dashboard
Human oversight and control	Confidence scores and source sentences are always shown, enabling human verification. Users can correct any extracted term. A legal disclaimer is present on every results page. Chat responses cite page numbers for traceability. No irreversible action is taken by the AI

Transparency:

Question	Answer
Direct and indirect use cases	Direct: SMBs and freelancers reviewing NDA/MSA contracts before signing. Indirect (potential misuse): extracting competitive intelligence from contracts shared without authorisation. Mitigation: terms of service prohibit use of third-party confidential contracts without permission
How are results generated	PDF → text extraction (pdf-parse) → structured prompt to GPT-4o (OpenAI API) → JSON output parsed and stored → displayed in key terms panel. No training of any model on user data
Benchmarks shared with users	Accuracy benchmarks (F1 on NDA/MSA test sets) and confidence calibration results will be published on a public trust page once post-launch eval is complete
Disclosures required	“Not legal advice” disclaimer on every results page; confidence scores visible per term; source sentence expandable per term; “Powered by OpenAI GPT-4o” attribution in footer

Fairness:

Question	Answer
Which groups are underrepresented	Non-US/non-UK contract conventions; contracts from South Asian, African, or Latin American jurisdictions; highly specialised industries with non-standard clause structures (healthcare, defence)

Question	Answer
Why they don't work well and the plan	Training data (CUAD) is heavily weighted toward US commercial contracts. Prompt few-shot examples are US/UK-biased. Plan: after launch, collect opt-in anonymised data from non-US users; add non-US few-shot examples in v1.2; evaluate by jurisdiction in monthly audit
Test/feedback loop to identify gaps	Monthly accuracy audit segmented by contract jurisdiction (where available) and industry; user feedback tagged with contract type helps surface systematic gaps

Reliability & Safety:

Question	Answer
Acceptable error rates	≤ 12% of terms corrected by users in production; ≤ 5% hallucinated chat responses; 0% critical failures (data exposed to wrong user)
Consequences of bad input	Corrupted PDF → graceful error message, no partial output stored. Non-contract document (e.g. invoice) → AI extracts what it can, confidence scores will be low, user sees ⚠️ warnings on most terms
Recovery plan if system fails	OpenAI API failure: 3-retry with backoff, then surface error to user with "Try again" CTA; contract status set to `'error'` in DB so user can retry without re-uploading. Supabase downtime: frontend shows maintenance banner; no data loss risk as all writes are transactional
How is system health monitored	Vercel deployment logs; Supabase dashboard for DB and storage metrics; OpenAI usage dashboard for token consumption and error rates; Uptime Robot for endpoint monitoring with alerts to team Slack
Customer communication plan	P0 incident (data exposure, complete outage): in-app banner + email to all affected users within 1 hour; status page updated within 30 minutes. P1 incident (degraded performance): in-app banner within 2 hours

---

12. Pricing

Development Costs (One-Time / MVP — 14-week build)

Item	Estimated Cost
Supabase Pro setup (3 months during build)	$75

Item	Estimated Cost
OpenAI API credits (development + testing — 2,000 test analyses)	$400
Vercel Pro (3 months during build)	$60
Domain + SSL	$20
Misc. tooling (Figma, Notion, GitHub)	$100
Infrastructure subtotal	**$655**

Role	Qty	Monthly (assumed)	Duration	Subtotal
Product Manager	1	$8,000	3.5 months	$28,000
Lead Fullstack Engineer	1	$10,000	3.5 months	$35,000
Frontend Engineer	1	$8,000	3.5 months	$28,000
QA / DevOps	0.5	$6,000	3.5 months	$10,500
UX Designer	0.5	$7,000	2 months	$7,000
Manpower subtotal				**$108,500**
Total one-time (MVP)				**~$109,155**

Operational Costs (Monthly, at 500 active users / ~2,000 contracts/month)

Item	Monthly Cost
Supabase Pro	$25
Supabase Storage add-on (estimated 5 GB/month PDF storage)	$0 (within Pro tier)
OpenAI API usage (2,000 analyses × $0.15 avg)	$300
Vercel Pro (hosting + edge functions)	$20
Uptime monitoring	$10
Total monthly operational	**~$355**

Cost per active user per month at 500 users: **$0.71** — unit economics are extremely favourable.

Market Size

- **TAM:** $4.1B — global contract lifecycle management (CLM) software market, projected 2027 (Markets and Markets, assumed)
- **SAM:** $820M — English-speaking SMBs and freelancers (defined as companies under 250 employees) needing NDA/MSA review tools; approximately 20% of CLM TAM

- **SOM:** $16M — attainable within 24 months by targeting 8,000 paying customers at $166 average ARR, achievable through direct-to-user acquisition (content SEO, Product Hunt, LinkedIn)

Revenue Potential

Scenario	Paying Customers (Year 2)	ARPU	ARR
Conservative	800	$180	$144,000
Target	3,000	$210	$630,000
Optimistic	8,000	$240	$1,920,000

Pricing Models Considered

Model	Pros	Cons	Verdict
Per-document	Low friction for light users; natural upsell signal	Unpredictable revenue; discourages frequent use	Offered as pay-as-you-go add-on only
Monthly subscription tiers	Predictable ARR; encourages habit formation; simple to communicate	Tier boundary friction; churn risk if user doesn't review enough contracts	**Primary model**
Usage-based (tokens/API calls)	Scales with value delivered	Confusing to end users; invisible cost anxiety	Not recommended for B2C
Freemium	Reduces sign-up friction; large top-of-funnel	Conversion risk; free users cost money to serve	Free trial only (14 days, not permanent free tier)

Directional Pricing

Plan	Price	Includes	Target User
Free Trial	$0 / 14 days	5 contract analyses, full features	All new users
Starter	$19 / month	10 contract analyses/month, NDA + MSA, chat included	Freelancers, early-stage founders
Growth	$49 / month	40 contract analyses/month, custom terms, export, priority support	SMB operations managers
Pro	$129 / month	Unlimited analyses, team workspace (up to 5 seats), API access	Legal ops teams, agencies

Value anchor: "$19/month = less than 5 minutes of lawyer time — and you get 10 full contract reviews."

---

13. Assumptions

The following assumptions were made to produce a complete PRD. Each is a risk item that should be validated before the corresponding phase begins:

1. **OpenAI GPT-4o achieves ≥ 82% F1** on NDA and MSA key term extraction with few-shot prompting, without fine-tuning. This is the most critical assumption — if false, a fine-tuned model or alternative LLM (Claude 3.5, Gemini 1.5 Pro) must be evaluated before v0.2 ships.
2. **Target users are English-speaking** and review contracts governed by US or UK law. International contract conventions are out of scope for the initial 12 months.
3. **Uploaded contracts are text-layer PDFs**, not scanned images. Scanned PDFs will receive a graceful error message. OCR support is planned for v1.2.
4. **Supabase free tier is sufficient for development and early alpha testing** (< 200 contracts stored). The Pro plan at $25/month is assumed for beta and production.
5. **Contract length is ≤ 20 pages / ≤ 15,000 tokens** for MVP. Contracts longer than this will be rejected with a clear error message and a note that longer contract support is coming.
6. **The team has access to a legal SME** who can annotate 50 ground-truth contracts for the evaluation set before beta launch. If no SME is available, the CUAD dataset will be used as the sole ground truth, reducing eval confidence.
7. **MVP build team consists of 2–3 engineers and 1 PM**, working full-time for 14 weeks. Timeline estimates assume this capacity.
8. **OpenAI pricing remains within ±30% of current rates** ($0.005/1k input, $0.015/1k output for GPT-4o) over the first 12 months post-launch.
9. **Supabase Row Level Security correctly isolates user data** without a custom auth middleware layer. This assumption is validated by Supabase's documented RLS capability — it must be verified by a security review before launch.
10. **Users are comfortable with browser-based PDF viewing** and do not require a native desktop application for PDF review workflows.
11. **Chat uses full contract text as context in every turn**, not a chunked RAG approach. This works for contracts ≤ 15,000 tokens but will require a chunking strategy if contract length limits are increased post-v1.0.
12. **Pricing model and tiers are directional only** — final pricing will be validated through user interviews and a pricing sensitivity survey during the beta phase.
13. **The Supabase Storage bucket and its RLS policies are created via SQL**, not via the Supabase dashboard. The `database.sql` file must include `INSERT INTO storage.buckets` and three `CREATE POLICY ON storage.objects` statements (INSERT, SELECT, DELETE). If these are omitted from the SQL file, PDF uploads will silently fail — the upload route will catch the storage error, leave `file_path = null`, and the PDF viewer will not render. The text viewer fallback will still work because `contract_text` is stored independently in the DB.
14. **The full conversation history is passed to the chat model on every turn**, not just the last 10 messages. The chat route fetches all messages for the session (up to 200) in ascending order and passes them as the message array. This enables memory-style questions ("what did you say earlier about X?"). The query classification layer (`contract` / `history` / `both`) adjusts the system prompt and context inclusion without an extra API call.