Contract Analyzer with Generative AI

Alireza Esmaeilzehi

1. Introduction

Automated end-to-end document processing for assessing compliance with internal policies is of critical importance in the insurance industry, including organizations such as Manulife. Recent advances in generative artificial intelligence (Generative AI), retrieval-augmented generation (RAG), and large language models (LLMs) have significantly improved the feasibility and effectiveness of building intelligent systems capable of understanding and reasoning over complex legal and contractual documents. These technologies enable the extraction of semantically relevant information, contextual reasoning, and the generation of structured, explainable outputs from unstructured text.

In this report, I present an end-to-end contract compliance analysis system that leverages natural language processing (NLP), Generative AI, and LLMs to automate the evaluation of contractual documents against predefined internal policy requirements. The proposed system accepts a contract PDF as input, performs structured content extraction and semantic retrieval, applies LLM-based reasoning to assess compliance, and produces a structured JSON output summarizing compliance states, supporting evidence, and explanations. The following sections describe the design choices, system architecture, and key components of the proposed solution in detail.

2. Proposed Contract Compliance Analysis Scheme

In this section, I describe the proposed end-to-end approach for contract compliance analysis, covering both the backend processing pipeline and the frontend user interface design.

2.1. Preprocessing

The preprocessing stage converts raw contract PDFs into a clean, retrieval-ready textual corpus while preserving evidentiary traceability. Since security and compliance obligations may appear both in narrative clauses and in structured exhibits, the pipeline is designed to capture complementary document representations rather than relying on a single extraction pathway.

To this end, both free-form textual content and structured tabular content are extracted from each page of the document. Narrative text captures contractual language such as obligations, requirements, and exceptions, while tables often encode compliance controls, responsibilities, and review schedules in a condensed form. Extracting both modalities ensures that relevant evidence is not lost due to formatting differences across contracts and appendices.

The extracted content is then lightly normalized to remove formatting artifacts introduced during PDF rendering. Excessive line breaks, empty lines, and inconsistent spacing are cleaned while preserving paragraph boundaries and logical structure. This normalization step improves downstream segmentation and retrieval quality without altering the original contractual wording.

Following normalization step, the document content is segmented into overlapping text chunks of fixed length. Overlapping segmentation is critical in legal and compliance documents, where relevant obligations frequently span multiple sentences or clauses. The overlap ensures that semantically complete requirements remain intact even when they cross chunk boundaries, improving recall during semantic retrieval.

All resulting text segments, whether originating from narrative text or tabular data, are aggregated into a unified corpus. Each segment is explicitly linked to its source page in the original document. Maintaining this page-level provenance is essential for the task of contract compliance analysis, as it enables later stages to produce evidence-grounded conclusions with verifiable citations rather than abstract summaries.

Overall, the above preprocessing pipeline balances robustness, efficiency, and auditability. It ensures broad coverage of contractual evidence, minimizes retrieval failures caused by document formatting, and preserves traceability required for enterprise-grade compliance assessment.

2.2. Information Retrieval

After preprocessing step, the proposed scheme employs a semantic retrieval stage to identify the most relevant contract excerpts for each compliance requirement. Rather than relying on keyword matching, the retriever operates in a continuous embedding space, enabling it to capture semantic equivalence between compliance questions and contract language even when terminology differs across vendors or documents.

Each text segment in the corpus is encoded into a dense vector representation using a pretrained sentence-level embedding model. These embeddings are designed to capture the semantic meaning of clauses, allowing requirements such as "password rotation," "credential vaulting," or "secure administrative access" to be matched even when expressed using different phrasing. To support efficient similarity search at scale, all embeddings are normalized and indexed in a vector similarity structure optimized for inner-product search.

At query time, each compliance question is embedded using the same representation space as the document corpus. The retriever then performs a nearest-neighbor search to identify the top-ranked document segments that are semantically closest to the query. This dense retrieval approach prioritizes meaning rather than surface form, which is especially important in legal and security documents where equivalent obligations may be expressed indirectly or across multiple clauses.

The retriever returns a small set of high-scoring text segments, each annotated with its source page. These retrieved segments collectively serve as the evidentiary context for downstream compliance analysis. By constraining later stages to operate only over retrieved evidence, the system reduces hallucination risk and ensures that all conclusions are grounded in the original contract language.

Overall, the retrieval stage acts as a semantic filter between raw contract text and compliance assessment. It enables scalable, high-recall evidence selection while maintaining strict traceability, forming a reliable foundation for the task of compliance analysis in subsequent stages.

2.3. Large Language Model for Reasoning and Text Generation

This stage of the proposed scheme uses a large language model to perform structured compliance reasoning over the evidence retrieved from the contract. Rather than processing the entire document, the model is provided only with the most relevant excerpts identified by the retrieval stage, ensuring that all conclusions are grounded in explicit contractual language. This retrieval-augmented design constrains the model's reasoning space and reduces the risk of unsupported or speculative outputs.

The LLM stage of the proposed scheme uses a Mistral instruction-tuned large language model to perform evidence-based compliance reasoning over retrieved contract excerpts. Mistral models are decoder-only transformer architectures optimized for strong reasoning performance, efficient inference, and high-quality instruction following, making them well suited for contract compliance analysis tasks on long, technical documents.

Mistral is used in this stage because it provides a strong balance between reasoning capability and computational efficiency, enabling local deployment without reliance on external APIs. This allows sensitive contractual data to remain within the execution environment while still benefiting from advanced natural language understanding for interpreting nuanced security and compliance obligations.

For each compliance requirement, the model is prompted with a clear statement of the obligation, followed by verbatim contract excerpts annotated with their source pages. The model is instructed to assess compliance strictly based on this provided evidence and to avoid introducing external assumptions or knowledge. This setup mirrors how a human compliance reviewer evaluates contractual controls by comparing requirements against specific clauses and exhibits.

The model produces a structured assessment consisting of a compliance classification, a confidence score, supporting quotes, and a concise rationale. Enforcing a fixed output schema ensures consistency across compliance categories and enables downstream validation and integration with automated review workflows. Structured output also simplifies quality control by making missing evidence, uncertainty, or partial coverage immediately visible.

To improve reliability, the system includes safeguards that validate and normalize model outputs before acceptance. If the model output is malformed or inconsistent with the expected structure, the system retries the generation using a more compact evidence context. When valid structured output cannot be obtained, the system defaults to a conservative compliance assessment rather than emitting unverified conclusions. This design prioritizes robustness and trustworthiness over optimistic inference.

Overall, the LLM stage functions as a constrained reasoning engine rather than a free-form text generator. By combining retrieval-bounded context, explicit instructions, and strict output validation, the model is able to synthesize defensible, evidence-grounded compliance assessments suitable for enterprise risk and audit use cases.

2.4. Output Structure of the Proposed Scheme

The proposed model outputs its assessment in a structured JSON format to ensure consistency, machine readability, and downstream validation. This format enforces explicit compliance states, confidence scores, supporting evidence quotes, and rationales, enabling reliable integration with automated review and audit workflows.

The proposed scheme is developed in Python using the PyTorch deep learning framework and associated ecosystem libraries for document processing, retrieval, and model inference. All developments are conducted on an NVIDIA A100 GPU within the Google Colab environment.

2.5. Frontend Design of the Proposed Scheme

A lightweight web-based user interface is developed to enable interactive execution and inspection of the proposed contract compliance analysis method. The interface allows users to upload a PDF contract, trigger the analysis process, and receive structured compliance results without requiring direct interaction with the underlying code or models. This design supports rapid evaluation and demonstration of the system in a user-friendly manner.

Once a document is uploaded, the interface orchestrates the end-to-end analysis workflow, including document parsing, semantic retrieval, and large language model reasoning. Progress indicators are displayed during execution to provide transparency into the system's runtime behavior. Upon completion, the compliance results are presented in both machine-readable and human-readable forms, enabling immediate inspection and validation.

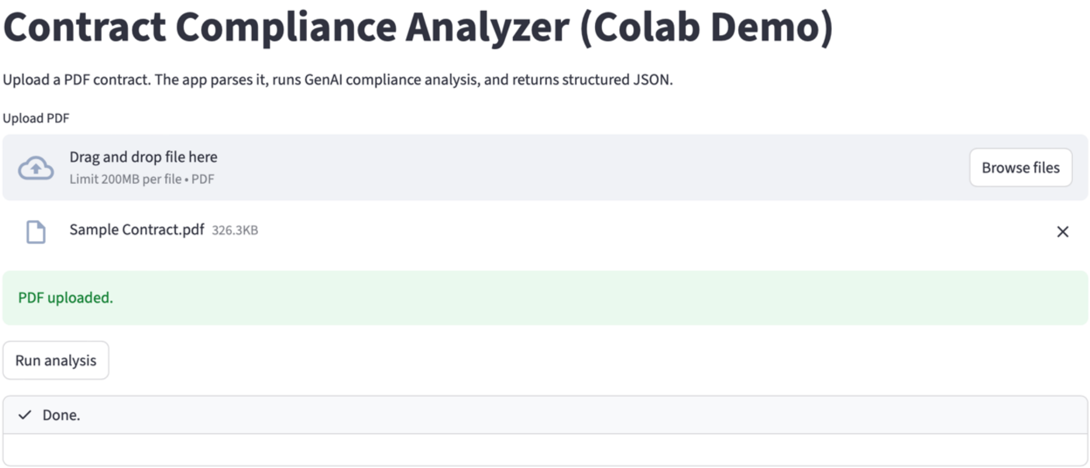
The primary output is shown as structured JSON, reflecting the standardized compliance schema used throughout the method. In addition, the results are rendered as a tabular view that summarizes compliance states, confidence scores, supporting evidence, and rationales across all evaluated requirements. This dual presentation facilitates both technical integration and manual review by compliance or security stakeholders.

To support portability and downstream use, the interface also provides an option to export the compliance report as a JSON file. This enables seamless integration with external governance, risk, and compliance workflows or archival systems.

The frontend is deployed within a cloud-based notebook environment and exposed via a secure tunneling mechanism, allowing external access to the interface without requiring permanent server infrastructure. This setup enables convenient sharing and demonstration of the system while preserving the flexibility and reproducibility of an ephemeral execution environment.

Streamlit is used to implement the frontend interface, providing a lightweight framework for rapidly building interactive web applications directly in Python. It enables seamless integration with the analysis method while offering an intuitive interface for document upload, execution control, and result visualization.

Figs. 1-4 illustrate the typical examples of the frontend design of the proposed contract compliance analysis scheme.



Contract Compliance Analyzer (Colab Demo)

Upload a PDF contract. The app parses it, runs GenAI compliance analysis, and returns structured JSON.

Upload PDF

Drag and drop file here

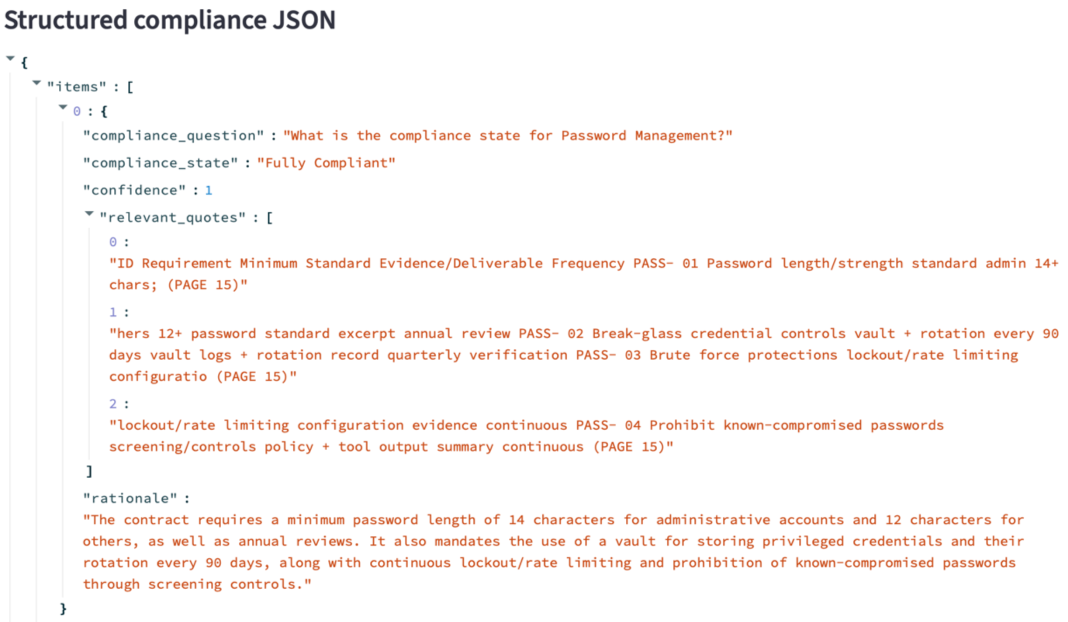
Limit 200MB per file • PDF

Sample Contract.pdf 326.3KB

PDF uploaded.

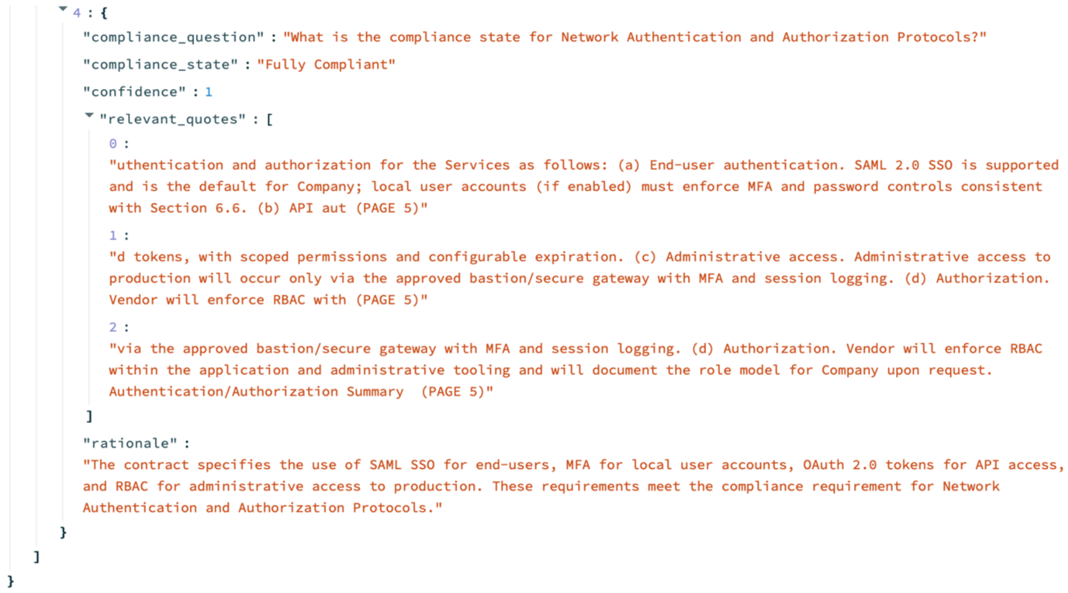
✓ Done.

Fig. 1. Uploading PDF in the User Interface of the Proposed Scheme.



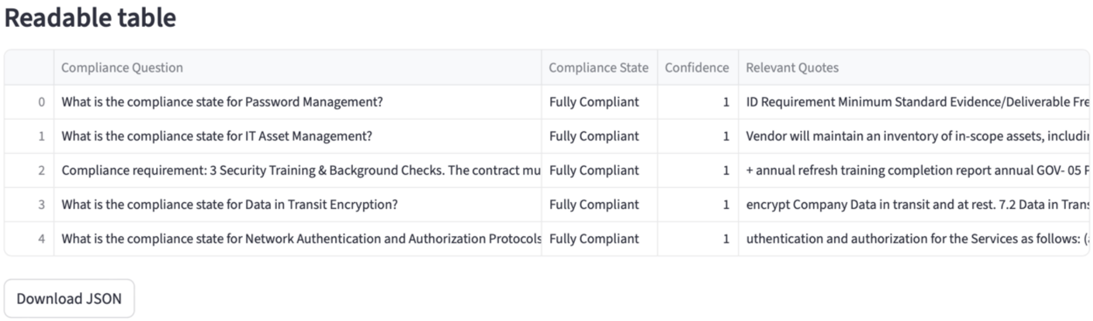
```
Structured compliance JSON
{
  "items" : [
    0 : {
      "compliance_question" : "What is the compliance state for Password Management?"
      "compliance_state" : "Fully Compliant"
      "confidence" : 1
      "relevant_quotes" : [
        0 :
        "ID Requirement Minimum Standard Evidence/Deliverable Frequency PASS- 01 Password length/strength standard admin 14+
        chars; (PAGE 15)"
        1 :
        "hers 12+ password standard excerpt annual review PASS- 02 Break-glass credential controls vault + rotation every 90
        days vault logs + rotation record quarterly verification PASS- 03 Brute force protections lockout/rate limiting
        configuratio (PAGE 15)"
        2 :
        "lockout/rate limiting configuration evidence continuous PASS- 04 Prohibit known-compromised passwords
        screening/controls policy + tool output summary continuous (PAGE 15)"
      ]
      "rationale" :
      "The contract requires a minimum password length of 14 characters for administrative accounts and 12 characters for
      others, as well as annual reviews. It also mandates the use of a vault for storing privileged credentials and their
      rotation every 90 days, along with continuous lockout/rate limiting and prohibition of known-compromised passwords
      through screening controls."
    }
```

Fig. 2. Output of the Proposed Scheme to Question 1.



```
    4 : {
      "compliance_question" : "What is the compliance state for Network Authentication and Authorization Protocols?"
      "compliance_state" : "Fully Compliant"
      "confidence" : 1
      "relevant_quotes" : [
        0 :
        "uthentication and authorization for the Services as follows: (a) End-user authentication. SAML 2.0 SSO is supported
        and is the default for Company; local user accounts (if enabled) must enforce MFA and password controls consistent
        with Section 6.6. (b) API aut (PAGE 5)"
        1 :
        "d tokens, with scoped permissions and configurable expiration. (c) Administrative access. Administrative access to
        production will occur only via the approved bastion/secure gateway with MFA and session logging. (d) Authorization.
        Vendor will enforce RBAC with (PAGE 5)"
        2 :
        "via the approved bastion/secure gateway with MFA and session logging. (d) Authorization. Vendor will enforce RBAC
        within the application and administrative tooling and will document the role model for Company upon request.
        Authentication/Authorization Summary  (PAGE 5)"
      ]
      "rationale" :
      "The contract specifies the use of SAML SSO for end-users, MFA for local user accounts, OAuth 2.0 tokens for API access,
      and RBAC for administrative access to production. These requirements meet the compliance requirement for Network
      Authentication and Authorization Protocols."
    }
  ]
}
```

Fig. 3. Output of the Proposed Scheme to Question 5.



Readable table

	Compliance Question	Compliance State	Confidence	Relevant Quotes
0	What is the compliance state for Password Management?	Fully Compliant	1	ID Requirement Minimum Standard Evidence/Deliverable Fre
1	What is the compliance state for IT Asset Management?	Fully Compliant	1	Vendor will maintain an inventory of in-scope assets, includi
2	Compliance requirement: 3 Security Training & Background Checks. The contract mu	Fully Compliant	1	+ annual refresh training completion report annual GOV- 05 F
3	What is the compliance state for Data in Transit Encryption?	Fully Compliant	1	encrypt Company Data in transit and at rest. 7.2 Data in Tran
4	What is the compliance state for Network Authentication and Authorization Protocols	Fully Compliant	1	uthentication and authorization for the Services as follows: (

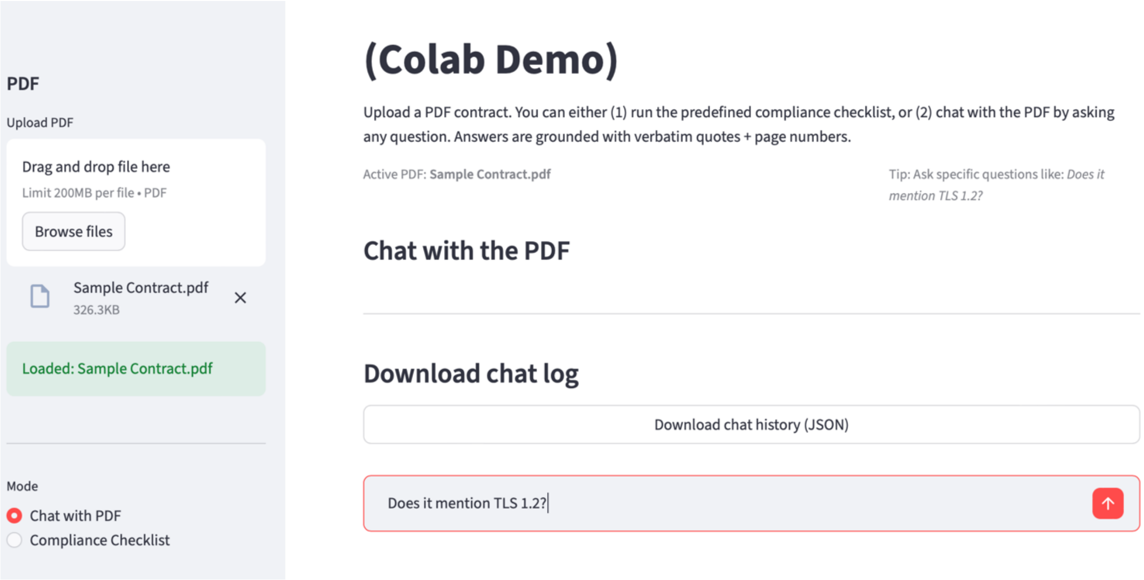
Fig. 4. Table Containing the Summary of Outputs of the Proposed Scheme.

2.5. Bonus: Chat Functionality

To extend the system from a predefined compliance analyzer to a conversational assistant, the backend was modified to support dynamic user queries while preserving the original retrieval-augmented generation (RAG) architecture. Instead of iterating over a fixed list of requirements, the pipeline now accepts free-form questions at runtime and processes them using the same stages of document parsing, semantic chunking, embedding generation, and similarity-based retrieval. This design allows users to interactively explore the document while maintaining the structured and evidence-driven workflow established in the original implementation.

A generalized prompting strategy was introduced to transform the language model from a task-specific classifier into an evidence-constrained question-answering agent. Retrieved document excerpts remain the primary source of truth, and the conversational context is incorporated only to improve coherence without influencing factual grounding. The model is instructed to generate structured outputs based solely on retrieved evidence, ensuring that responses remain interpretable, verifiable, and aligned with the underlying document content.

Figs. 5-7 illustrate the typical examples of the conversational variant of the proposed contract compliance analysis scheme.



Loaded: Sample Contract.pdf

● Chat with PDF

(Colab Demo)

Upload a PDF contract. You can either (1) run the predefined compliance checklist, or (2) chat with the PDF by asking any question. Answers are grounded with verbatim quotes + page numbers.

Active PDF: **Sample Contract.pdf**

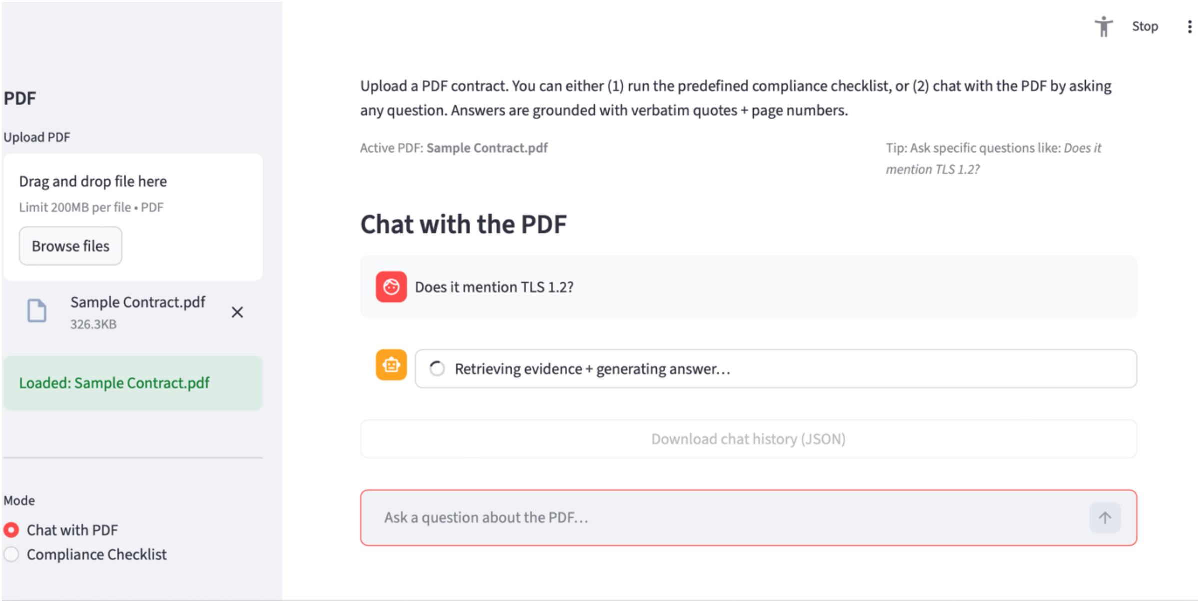
Tip: Ask specific questions like: *Does it mention TLS 1.2?*

Chat with the PDF

Download chat log

Does it mention TLS 1.2?

Fig. 5. Front End Design of the Conversational Variant of the Proposed Scheme.



Loaded: Sample Contract.pdf

● Chat with PDF

Upload a PDF contract. You can either (1) run the predefined compliance checklist, or (2) chat with the PDF by asking any question. Answers are grounded with verbatim quotes + page numbers.

Active PDF: **Sample Contract.pdf**

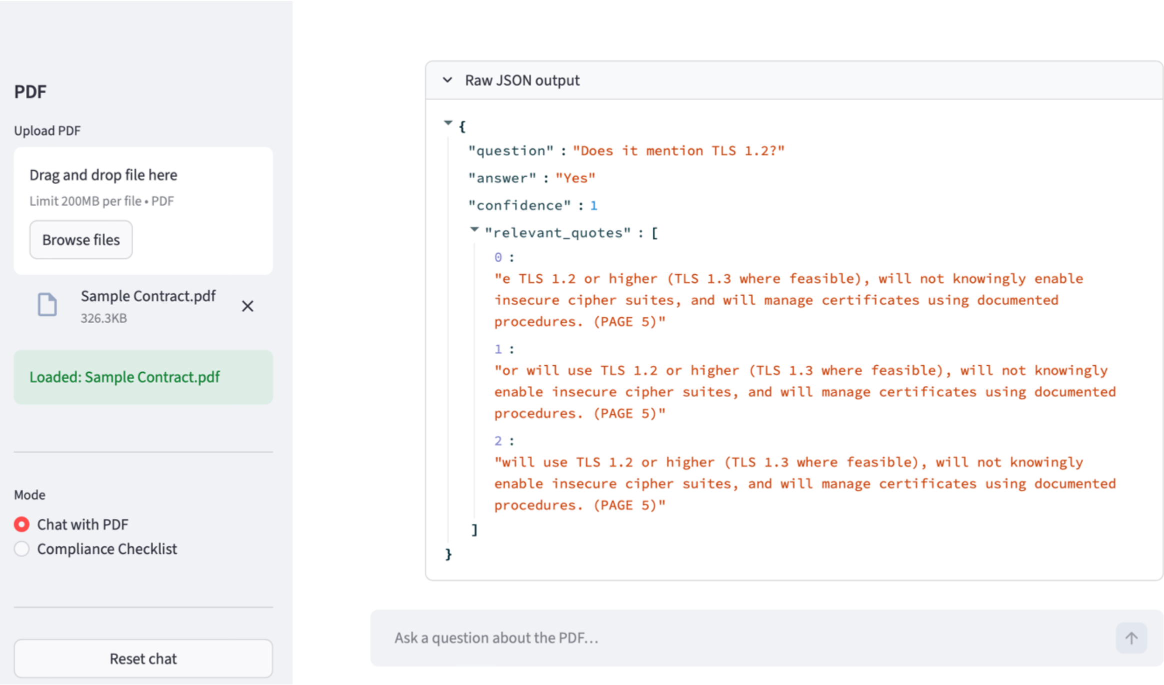
Tip: Ask specific questions like: *Does it mention TLS 1.2?*

Chat with the PDF

Does it mention TLS 1.2?

Retrieving evidence + generating answer…

Fig. 6. Processing of the Input PDF in the Conversational Variant of the Proposed Scheme.



```
{
  "question" : "Does it mention TLS 1.2?"
  "answer" : "Yes"
  "confidence" : 1
  "relevant_quotes" : [
    0 :
    "e TLS 1.2 or higher (TLS 1.3 where feasible), will not knowingly enable
    insecure cipher suites, and will manage certificates using documented
    procedures. (PAGE 5)"
    1 :
    "or will use TLS 1.2 or higher (TLS 1.3 where feasible), will not knowingly
    enable insecure cipher suites, and will manage certificates using documented
    procedures. (PAGE 5)"
    2 :
    "will use TLS 1.2 or higher (TLS 1.3 where feasible), will not knowingly
    enable insecure cipher suites, and will manage certificates using documented
    procedures. (PAGE 5)"
  ]
}
```

Fig. 7. Output of the Conversational Variant of the Proposed Scheme to the User Question.

3. Conclusion

In this work, I have presented an end-to-end, retrieval-augmented contract compliance analysis method for contractual documents that combines robust PDF preprocessing, semantic retrieval, and large language model reasoning. The proposed approach enables evidence-grounded, structured compliance assessments while preserving traceability to source document pages, addressing key challenges in scalability and auditability of manual reviews. By integrating efficient retrieval with constrained instruction-tuned reasoning and an interactive frontend, the system demonstrates a practical and reproducible framework for automating contract compliance analysis in enterprise settings.