ARUP Laboratories Writeup for the ADLM 2025 Data Science Challenge

Paul English[1], Muir Morrison[1], Samir Atiya[3], Mattia Medina Grespan[1], Evelyn Goodman[1], Weston Prestwich[1], Ashley Hutchings[1], Christopher Monson[1], Ray Linford[1], Austin Wheeler[1], Mark Dewey[1], Brendan O'Fallon[1], and Nick Spies[1,2]

[1]ARUP Institute for Research and Innovation in Diagnostic and Precision Medicine, ARUP Laboratories, Salt Lake City, Utah, USA
[2]Department of Pathology, University of Utah, Salt Lake City, Utah, USA
[3]Department of Pathology, University of Chicago Medical Center, Chicago, IL, USA

November 15, 2025

1 Introduction

The 2025 ADLM Data Science Challenge offers a unique perspective in the challenge of searching and understanding complex lab medicine documents, both proprietary and public. Our objective in this work is to develop and evaluate a retrieval-augmented generation (RAG)–based assistant that supports question answering over a corpus of laboratory documents, including regulatory filings and related materials. We focus on a pragmatic, end-to-end system that can be deployed locally, emphasizing transparency of the retrieval process and the ability to handle challenging content such as dense tables and scanned PDFs.

To accomplish this, we (1) parse PDFs using multiple open-source document understanding tools, (2) construct a hybrid retrieval pipeline combining full-text search and dense vector embeddings, (3) incorporate multimodal table understanding using a vision–language model, (4) apply reranking to refine retrieved evidence, and (5) serve answers through a web-based chat interface backed by a Django API. Throughout, we favor well-established but robust components that can be reasoned about and extended in real laboratory settings.

This writeup describes our system design and implementation, including PDF parsing and chunking strategies, search and embedding infrastructure, multimodal table retrieval, and LLM usage. We then summarize the data artifacts we provide for reproducibility, report qualitative results and limitations of the current prototype, and discuss opportunities for future improvement, including knowledge-graph–based retrieval and more sophisticated chunking and indexing tailored to laboratory documentation.

2 Methods

Broadly, this project requires several interlocking pieces:

- A web interface for users to interact with an LLM, prompting it with questions and receiving answers.
- A database API to which the user-facing frontend can submit user queries and receive responses. This implements the retrieval component of RAG.
- The database itself, which intuitively should contain the collected content and knowledge of the dataset of interest, but pre-processed in ways that relevant pieces can be searched for, found, and injected into the LLM's context when responding to a user prompt.

There are potentially many ways of constructing the database, and this is where we devoted most of our effort. Plain-text search still has a place, even with the advent of LLMs. But the core of most RAG systems is a vector embedding database. In short, text documents are divided into chunks which are fed through a transformer encoder to generate numerical representations of each chunk, which ideally capture the semantic meaning of the chunk while ignoring irrelevant linguistic details. Much of the subtlety of this approach lies in choosing chunks well. For this competition's dataset in particular, the abundance of tables presents a challenge. Two options we explored are (1) converting tables to Markdown and embedding the resulting Markdown, and (2) converting the table to an image and embedding the resulting image with a vision language model. It is worth pointing out that all these approaches for pre-processing the data and building the database are not mutually exclusive; any or all can be used to populate the database with the same data expressed in multiple formats, hopefully increasing the chance that relevant results will be contained in the results of a query.

2.1 Web Frameworks

For the user-facing UI, we opted for React, which is still by far the most widely used web interface development framework and made for a simple way to present the UI for a chat assistant easily.

We chose Django for the database API. It is "boring software" [14], providing a proven framework for organizing simple CRUD-like web applications, making it easier for the team to contribute as needed. We utilized the package

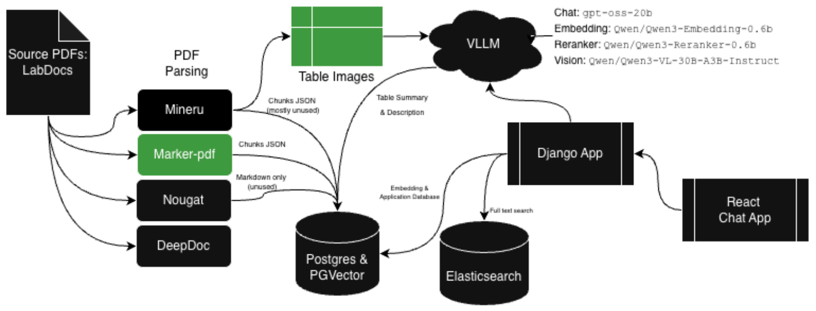


Figure 1: High-level pipeline view of the components in our application. The main source of content for the retrieval pipeline are the chunks natively exported by Marker, and the text descriptions generated from images of the tables in each PDF.

`django-ninja` to provide a "Flask"-like way of creating REST APIs quickly with documentation.

- https://www.djangoproject.com/
- https://django-ninja.dev/
- https://react.dev/

2.2 PDF Parsing

Numerous well-made parsing libraries and tools exist to extract text and information from PDFs. While the PDFs for this challenge contain accessible text, it is worthwhile to utilize specialized libraries that preserve semantic structure and classify various content elements. These libraries typically convert PDF content into HTML or Markdown format that maintains the original PDF layout and design flow.

We explored four parsers: **Marker** [4], an advanced document conversion tool that uses deep learning models to detect text regions, recognize content, and identify layout elements such as headers, footers, tables, and equations. It offers high accuracy and computational efficiency (up to 25 pages per second on high-end GPUs), with support for GPU, CPU, and Apple Silicon hardware. A key strength is its hybrid mode that leverages LLMs to enhance extraction quality for complex tables and unconventional layouts.

MinerU [8, 10, 15] is a state-of-the-art tool that converts PDFs into markdown, JSON, and LaTeX formats. It employs a 1.2 billion parameter multimodal vision-language model with a two-stage pipeline that decouples layout analysis from content recognition. MinerU excels at parsing complex layouts including rotated or borderless tables and mixed-language documents, with OCR support for 109 languages. It was particularly useful for our multimodal table retrieval approach, as it conveniently captures bounded images of every extracted table chunk.

Nougat [3] is a neural optical understanding tool specifically designed for academic documents, with advanced capabilities for parsing PDFs containing LaTeX mathematics and tables. It employs a visual transformer model optimized for scientific literature and converts PDF content into Mathpix Markdown. While it excels at recognizing mathematical formulas and table structures, it has restricted language support, performing best on English and other Latin-based languages.

DeepDoc [11] is a comprehensive document parsing tool that integrates advanced OCR, layout recognition, and table structure recognition (TSR) capabilities. It identifies ten fundamental layout components and applies TSR techniques to decode hierarchical headers and spanning cells, reconstructing tables into natural language sentences suitable for LLMs. However, it can be computationally intensive and time-consuming for highly complex layouts.

Detailed analyses of each parser's architecture, advantages, and limitations are provided in Appendix A.

2.3 Retrieval Pipeline

Our retrieval pipeline combines multiple search strategies to identify relevant document chunks for answering user queries. The pipeline processes queries through the following stages:

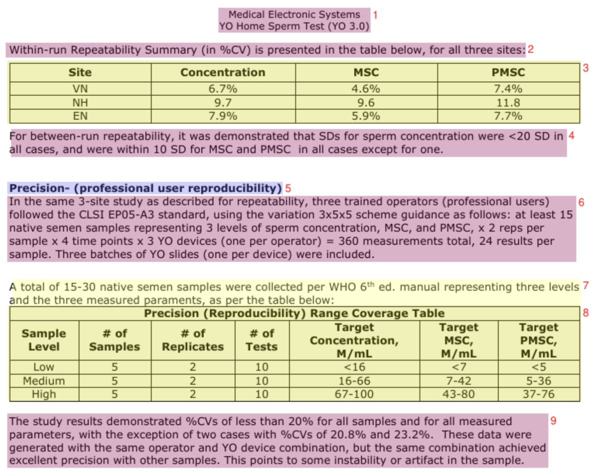
Chunking. We use chunks as provided by the PDF parsers without additional post-processing. This is a limitation of the current system, as more sophisticated chunking strategies tailored to laboratory documentation could improve retrieval quality.

Plain text search. We utilize the Django plugin `django-haystack` to integrate different search engines. We initially tested the Python `whoosh` framework for small-scale plain text searching, but it became impractically slow as we indexed more than a few thousand documents. We instead settled on a single-node Elasticsearch instance running via Docker.

Embedding search. We compute dense embeddings using the `Qwen3-0.6B-Embedding` model and perform vector similarity search over parsed chunks. This enables semantic retrieval that can match queries to relevant content even when exact keyword matches are absent.

Multimodal retrieval of tabular data. We took inspiration from RAG-Anything [6] to utilize the Qwen3-VL-30B model [13, 2] to extract detailed information from tables parsed from the PDFs. The MinerU package for PDF parsing (also used by RAG-Anything) conveniently captures a bounded image of every table chunk extracted from the PDF. We use a prompt similar to RAG-Anything, with additional context from neighboring chunks to provide more context when table headers and captions are not found.

Many tables span multiple pages. MinerU extracts each slice as a separate chunk. Stitching would help but requires layout heuristics not implemented here.


Medical Electronic Systems 1
YO Home Sperm Test (YO 3.0)

Within-run Repeatability Summary (in %CV) is presented in the table below, for all three sites: 2

Site	Concentration	MSC	PMSC
VN	6.7%	4.6%	7.4%
NH	9.7	9.6	11.8
EN	7.9%	5.9%	7.7%

3

For between-run repeatability, it was demonstrated that SDs for sperm concentration were <20 SD in all cases, and were within 10 SD for MSC and PMSC in all cases except for one. 4

Precision- (professional user reproducibility) 5

In the same 3-site study as described for repeatability, three trained operators (professional users) followed the CLSI EP05-A3 standard, using the variation 3x5x5 scheme guidance as follows: at least 15 native semen samples representing 3 levels of sperm concentration, MSC, and PMSC, x 2 reps per sample x 4 time points x 3 YO devices (one per operator) = 360 measurements total, 24 results per sample. Three batches of YO slides (one per device) were included. 6

A total of 15-30 native semen samples were collected per WHO 6th ed. manual representing three levels and the three measured paraments, as per the table below: 7

Precision (Reproducibility) Range Coverage Table 8

Sample Level	# of Samples	# of Replicates	# of Tests	Target Concentration, M/mL	Target MSC, M/mL	Target PMSC, M/mL
Low	5	2	10	<16	<7	<5
Medium	5	2	10	16-66	7-42	5-36
High	5	2	10	67-100	43-80	37-76

The study results demonstrated %CVs of less than 20% for all samples and for all measured parameters, with the exception of two cases with %CVs of 20.8% and 23.2%. These data were generated with the same operator and YO device combination, but the same combination achieved excellent precision with other samples. This points to some instability or artifact in the sample. 9

Figure 2: Parsed PDF from `K241628.pdf` showing tables highlighted in yellow and other detected content overlaid with different colors. Model-generated descriptions of the extracted tables are shown in Figures 3 and 4.

Reranking. We apply the `Qwen3-Reranker-0.6B` model to refine the candidate set retrieved from hybrid search (combining plain text, embedding, and multimodal results) before passing context to the final LLM.
LLM usage. We utilized local LLMs exclusively throughout development and experimentation, using HuggingFace transformers, Ollama, and VLLM for inference. Our primary models include Qwen3-VL-30B for vision-language tasks, Qwen3-Reranker-0.6B for reranking, Qwen3-Embedding-0.6B for embeddings, and GPT-OSS-20B for text generation.

3 Data

We have shared most of the intermediate data artifacts generated and used with our solution. We provide exports of the working PostgreSQL database in parquet files. We include archives of the PDF parsing for Marker, MinerU, and Nougat, which include images and layouts that show where the parsers captured content to extract. https://huggingface.co/datasets/paul-english/adlm25

4 Results

Our RAG-based agent performs at a basic level and is able to retrieve results across the document corpus, including tabular data found in the documents via summarization from a vision-language model. This offers basic Q&A, though significant work would still be needed to ensure this is an accurate and robust tool for use in any lab or professional capacity. PDF parsing is relatively good,

```
This is a table that presents data on the percentage of different
cell types (likely from a scientific or medical study) at three
distinct sites: VN, NH, and EN.

The table is organized into four columns:
- **Site:** The location where the data was collected (VN, NH, EN).
- **Concentration:** A percentage value, likely representing the
  total concentration of a specific cell type or a general measure
  of cell density.
- **MSC:** A percentage value, likely representing the percentage
  of Mesenchymal Stem Cells.
- **PMSC:** A percentage value, likely representing the percentage
  of Perivascular Mesenchymal Stem Cells.

The data shows the following values for each site:
- **VN:** Concentration 6.7%, MSC 4.6%, PMSC 7.4%
- **NH:** Concentration 9.7%, MSC 9.6%, PMSC 11.8%
- **EN:** Concentration 7.9%, MSC 5.9%, PMSC 7.7%
```

Figure 3: Qwen3-VL-30B model output for the first table in Figure 2, accurately capturing all table information.

```
This is a table titled "Precision (Reproducibility) Range Coverage
Table." It outlines the parameters for a precision study, likely
for a laboratory test, based on different sample levels (Low,
Medium, High). The table specifies the number of samples,
replicates, and tests, along with the target concentration and the
corresponding target ranges for MSC (Methicillin-Susceptible
*Staphylococcus aureus* or a similar analyte) and PMSC (a related
or different analyte) in M/mL.
```

Figure 4: Qwen3-VL-30B model output for the second table in Figure 2. The description is accurate but lacks exact numeric details.

though some sections are missed. Retrieval works well with hybrid full-text and vector-based searching, though chunking and indexing of the documents could be far more tailored to the specific types of documents we are processing.

We believe that knowledge graph techniques would offer improvements both in simple retrieval and for more complex "multi-hop" type reasoning [5, 7]. We explored extracting entities and relationships with LLM generation, and while the quality was high, we believe it was too slow to index for any large or dynamic corpus. Some modern techniques using an Open information extraction type method could offer a strong middle ground for knowledge graph building, trading off some quality for greater efficiency in index building [1, 9].

Challenges like tracking user sessions, conversation history, and continued polish and development into a system for searching and referencing material would require a dedicated team to develop and maintain. Many of the community-developed tools such as RagFlow or R2R could provide strong complete systems or backend loops for core RAG search [11, 12].

References

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, 2015.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

[3] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural optical understanding for academic documents, 2023.

[4] Datalab. Datalab's marker repository on github.

[5] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2025.

[6] Zirui Guo, Xubin Ren, Lingrui Xu, Jiahao Zhang, and Chao Huang. Rag-anything: All-in-one rag framework, 2025.

[7] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[8] Conghui He, Wei Li, Zhenjiang Jin, Chao Xu, Bin Wang, and Dahua Lin. Opendatalab: Empowering general artificial intelligence with open datasets. *arXiv preprint arXiv:2407.13773*, 2024.

[9] Adrian Lucas Malec. triplet-extract: Gpu-accelerated python implementation of stanford openie, 2025.

[10] Junbo Niu, Zheng Liu, Zhuangcheng Gu, Bin Wang, Linke Ouyang, Zhiyuan Zhao, Tao Chu, Tianyao He, Fan Wu, Qintong Zhang, Zhenjiang Jin, Guang Liang, Rui Zhang, Wenzheng Zhang, Yuan Qu, Zhifei Ren, Yuefeng Sun, Yuanhong Zheng, Dongsheng Ma, Zirui Tang, Boyu Niu, Ziyang Miao, Hejun Dong, Siyi Qian, Junyuan Zhang, Jingzhou Chen, Fangdong Wang, Xiaomeng Zhao, Liqun Wei, Wei Li, Shasha Wang, Ruiliang Xu, Yuanyuan Cao, Lu Chen, Qianqian Wu, Huaiyu Gu, Lindong Lu, Keming Wang, Dechen Lin, Guanlin Shen, Xuanhe Zhou, Linfeng Zhang, Yuhang Zang, Xiaoyi Dong, Jiaqi Wang, Bo Zhang, Lei Bai, Pei Chu, Weijia Li, Jiang Wu, Lijun Wu, Zhenxiang Li, Guangyu Wang, Zhongying Tu, Chao Xu, Kai Chen, Yu Qiao, Bowen Zhou, Dahua Lin, Wentao Zhang, and Conghui He. Mineru2.5: A decoupled vision-language model for efficient high-resolution document parsing, 2025.

[11] Ragflow. Deepdoc subproject of the ragflow repository.

[12] SciPhi-AI. R2r project repository.

[13] Qwen Team. Qwen3 technical report, 2025.

[14] Maurits van der Schee. The “boring software” manifesto.

[15] Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, Bo Zhang, Liqun Wei, Zhihao Sui, Wei Li, Botian Shi, Yu Qiao, Dahua Lin, and Conghui He. Mineru: An open-source solution for precise document content extraction, 2024.

Appendices

A PDF Parsing Tools

A.1 Marker

Marker is an advanced open-source document conversion tool designed to transform PDFs and other file formats into structured markdown, JSON, chunks, or HTML. Its processing pipeline uses deep learning models to detect text regions, recognize and transcribe text, identify layout elements such as headers, footers, tables, and equations, and clean and format the extracted content.

The pipeline starts with a segmentation model that locates text areas on scanned or digital pages, followed by an encoder-decoder recognition model that extracts the textual content. Layout detection models capture page structure and reading order, essential for preserving semantic flow. Marker also converts inline mathematical expressions into LaTeX format, enhancing downstream usability for scientific documents.

Key advantages of Marker include high accuracy in extracting text and complex elements such as tables, code blocks, equations, and embedded images, along with support for multi-language documents. Marker is computationally efficient, enabling rapid batch processing—up to 25 pages per second on high-end GPUs—and flexible, with support for GPU, CPU, and Apple Silicon hardware.

A distinct strength is its hybrid mode which leverages large language models (LLMs) to enhance extraction quality, especially for tables spanning multiple pages or unconventional layouts, by merging segmented elements and improving formatting accuracy.

Limitations of Marker primarily stem from the inherent complexity of PDF formats. Documents with very complex nested tables or forms can challenge the parser, leading to incomplete or imperfect extraction. Additionally, some forms and graphical elements may not render optimally. These issues can be mitigated by enabling OCR forcing and LLM-assisted extraction modes, though perfect accuracy remains difficult to guarantee for highly irregular layouts.

Overall, Marker provides a powerful, extensible foundation for semantic PDF parsing in our system, balancing speed, accuracy, and flexibility, while accommodating the diverse nature of laboratory documentation critical for effective retrieval-augmented question answering.

A.2 MinerU

MinerU is a state-of-the-art open-source document parsing tool that converts PDFs and related document types into machine-readable formats such as markdown, JSON, and LaTeX. Originating from the pre-training process of InternLM, MinerU addresses challenges in accurately recognizing and converting scientific literature, including complex symbols and layouts [10].

How MinerU Works: MinerU employs a multi-model fusion approach with a powerful yet compact 1.2 billion parameter multimodal vision-language model (MinerU2.5). It uses a two-stage pipeline decoupling layout analysis from content recognition for robustness. Key steps include removal of extraneous elements (headers, footers, footnotes), human-readable text ordering, and precise extraction of images, tables (converted into HTML), and formulas (converted into LaTeX). Its OCR supports 109 languages, and it is compatible across major platforms including Windows, Linux, and macOS.

Advantages: MinerU achieves high parsing speed and accuracy, supporting batch processing and large-scale applications with resource optimization that allows execution on modern GPUs and Apple Silicon devices. It excels at parsing complex layouts such as rotated or borderless tables, and mixed-language documents, including multilingual formulas. Continuous backend optimizations have improved OCR speed (200–300% gains) and table captioning accuracy. MinerU also offers diverse deployment options from CLI to Dockerized services, and web-based demos for easy evaluation.

Limitations: Despite its strengths, MinerU can struggle with extremely complex document layouts, such as nested tables, uncommon list formats, and vertical text, which may lead to misplaced reading order or incomplete extraction. Certain document types like comic books, textbooks, or heavily formatted exercise sheets remain challenging. Additionally, some formulas may not render perfectly in Markdown outputs. The parsing accuracy depends on hardware resources and model configurations, with some advanced features requiring substantial GPU memory.

Overall, MinerU provides a highly capable and extensible solution for parsing scientific and laboratory documentation, combining accuracy, speed, and rich extraction features suited to downstream retrieval-augmented workflows.

A.3 Nougat

Nougat is a neural optical understanding tool specifically designed for academic documents, with advanced capabilities to parse PDFs containing LaTeX mathematics and tables [3]. It employs a visual transformer model that performs OCR optimized for scientific literature, converting PDF content into a lightweight markup language compatible with Mathpix Markdown.

The core workflow begins with page image processing to predict layout, text, equations, and table regions. It integrates models trained on large scientific datasets like arXiv and PMC papers, enabling it to accurately recognize structured elements such as formulas and tables that traditional OCR systems often fail to capture properly. The output preserves semantic structure necessary for downstream tasks like retrieval-augmented generation.

Advantages of Nougat include its specialized design for scientific and academic text, which leads to improved recognition of mathematical formulas, table cells, and captions. Its open-source implementation allows flexible usage through a command-line interface, API, or dataset generation tools. Nougat supports output in markdown format that facilitates easy integration in pipelines requiring LaTeX and structured text.

However, Nougat's limitations include its restricted language support; it performs best on English and other Latin-based languages, with poor or no support for languages like Chinese, Russian, or Japanese. Additionally, partial failure detection heuristics can produce outputs on some hardware or document types, requiring manual disabling of skipping for robustness. The model's performance is also dependent on available GPU resources for efficient processing.

Overall, Nougat is a powerful tool for academic PDF parsing that complements other document understanding models by focusing on precise semantic extraction of scientific papers and maintaining compatibility with LaTeX-based workflows.

A.4 DeepDoc

DeepDoc is a comprehensive document parsing tool designed to handle diverse document formats, focusing primarily on PDFs with complex layouts [11]. Its architecture integrates advanced OCR, layout recognition, and table structure recognition (TSR) capabilities to extract semantically rich and structured information from documents spanning various domains.

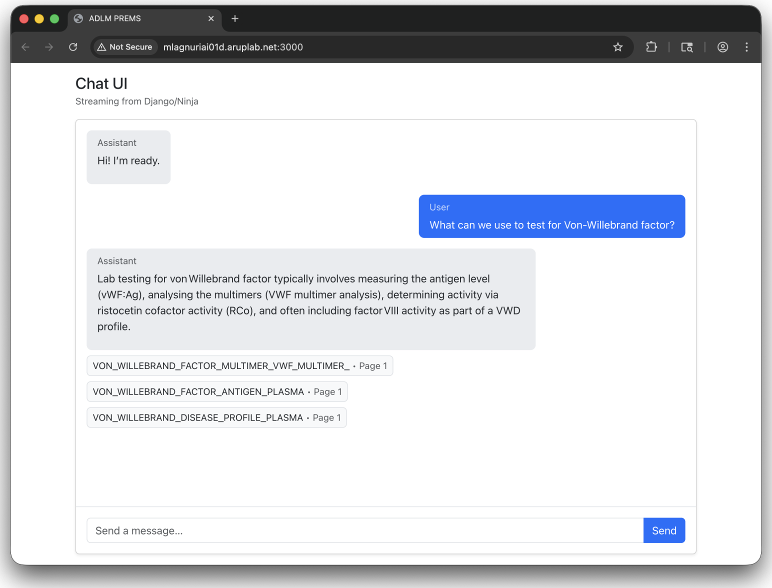
DeepDoc's vision module performs OCR on document images or PDFs to extract text along with positional information. It identifies ten fundamental layout components including text blocks, titles, figures, captions, tables, headers, footers, references, and equations. Accurate layout analysis enables downstream tasks by determining reading order and contextual relationships. For tables, DeepDoc applies TSR techniques to decode hierarchical headers, spanning cells, and projected row headers, reconstructing tables into comprehensible natural language sentences suitable for large language models.

The parser supports multiple file types such as PDF, DOCX, Excel, and PPT, generating detailed outputs that include text chunks with positions, images with captions, and aggregated natural language content from table cells. While PDF parsing is the most complex due to flexible formatting, DeepDoc excels with visual model techniques to provide robust OCR, table, and layout understanding.

Advantages of DeepDoc include its end-to-end design for multi-format support, accurate semantic segmentation of document elements, and explicit table content reassembly that enhances interpretability for AI workflows. However, it can be computationally intensive and time-consuming, with known challenges in processing highly complex multi-column layouts or documents with unusual formats. The ongoing development aims to optimize performance and expand parsing robustness.

Overall, DeepDoc provides a powerful, extensible basis for laboratory document analysis with a focus on improving semantic extraction and retrieval in challenging document contexts.

B Screenshots



Chat UI

Streaming from Django/Ninja

Assistant
Hi! I'm ready.

User
What can we use to test for Von-Willebrand factor?

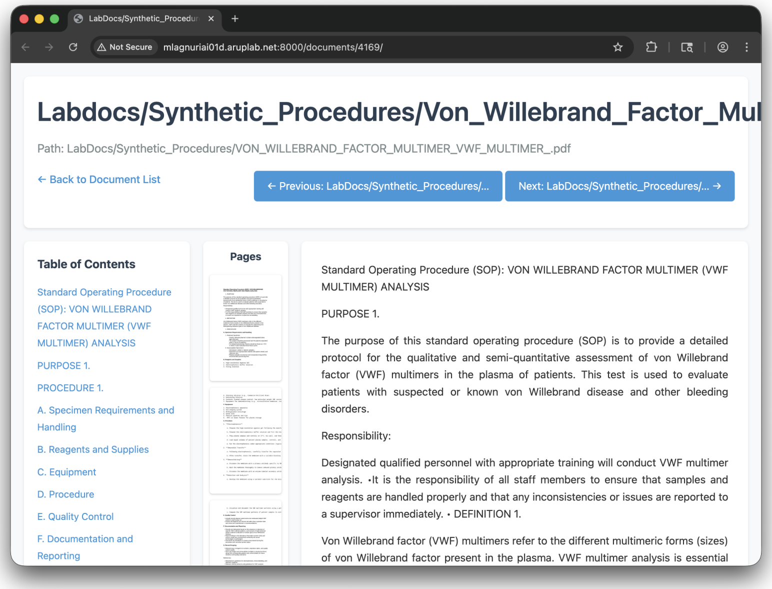
Assistant
Lab testing for von Willebrand factor typically involves measuring the antigen level (vWF:Ag), analysing the multimers (VWF multimer analysis), determining activity via ristocetin cofactor activity (RCo), and often including factor VIII activity as part of a VWD profile.

VON_WILLEBRAND_FACTOR_MULTIMER_VWF_MULTIMER_ · Page 1

VON_WILLEBRAND_FACTOR_ANTIGEN_PLASMA · Page 1

VON_WILLEBRAND_DISEASE_PROFILE_PLASMA · Page 1

Figure 5: The main chat UI.


LabDocs/Synthetic_Procedur
Not Secure mlagnuriai01d.aruplab.net:8000/documents/4169/

Labdocs/Synthetic_Procedures/Von_Willebrand_Factor_Mul

Path: LabDocs/Synthetic_Procedures/VON_WILLEBRAND_FACTOR_MULTIMER_VWF_MULTIMER_.pdf

← Back to Document List
← Previous: LabDocs/Synthetic_Procedures/...
Next: LabDocs/Synthetic_Procedures/... →

Table of Contents

Standard Operating Procedure (SOP): VON WILLEBRAND FACTOR MULTIMER (VWF MULTIMER) ANALYSIS

PURPOSE 1.

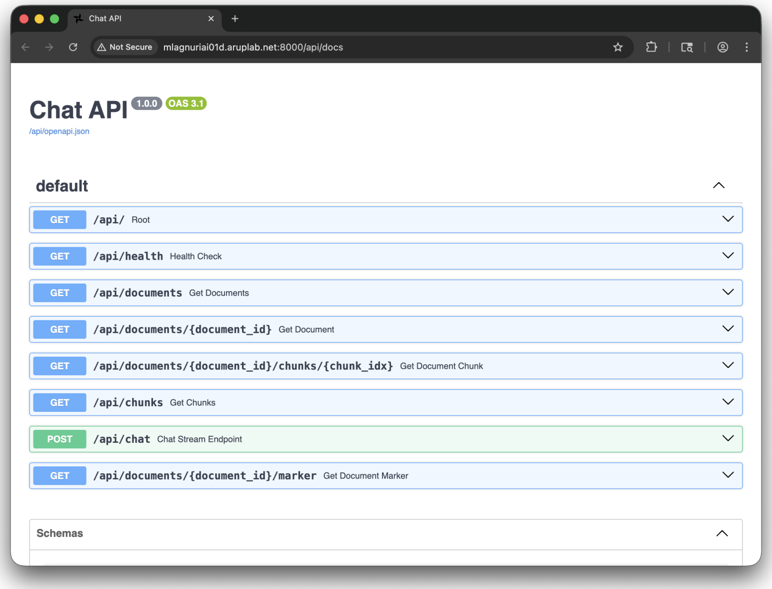
The purpose of this standard operating procedure (SOP) is to provide a detailed protocol for the qualitative and semi-quantitative assessment of von Willebrand factor (VWF) multimers in the plasma of patients. This test is used to evaluate patients with suspected or known von Willebrand disease and other bleeding disorders.

Responsibility:

Designated qualified personnel with appropriate training will conduct VWF multimer analysis. •It is the responsibility of all staff members to ensure that samples and reagents are handled properly and that any inconsistencies or issues are reported to a supervisor immediately. • DEFINITION 1.

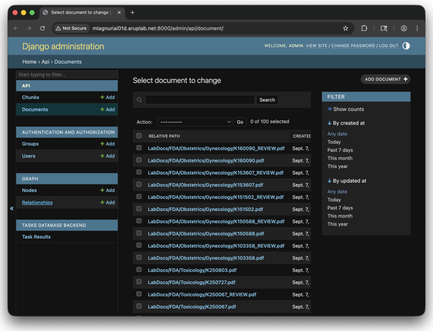
Von Willebrand factor (VWF) multimers refer to the different multimeric forms (sizes) of von Willebrand factor present in the plasma. VWF multimer analysis is essential

Figure 6: Parsed document viewer that the chat interface links to when a document is cited.



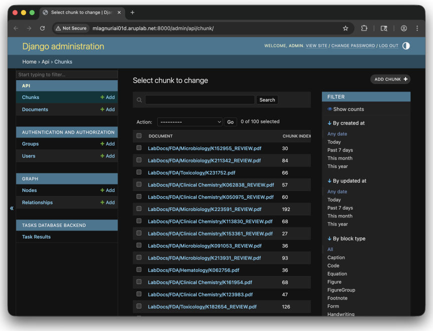
```
GET   /api/                                               Root
GET   /api/health                                         Health Check
GET   /api/documents                                      Get Documents
GET   /api/documents/{document_id}                        Get Document
GET   /api/documents/{document_id}/chunks/{chunk_idx}     Get Document Chunk
GET   /api/chunks                                         Get Chunks
POST  /api/chat                                           Chat Stream Endpoint
GET   /api/documents/{document_id}/marker                 Get Document Marker
```

Figure 7: The server API includes basic documentation.



RELATIVE PATH	CREATED
LabDocs/FDA/Obstetrics/Gynecology/K160090_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K160090.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K153607_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K153607.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K151502_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K151502.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K150588_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K150588.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K103358_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Obstetrics/Gynecology/K103358.pdf	Sept. 7,
LabDocs/FDA/Toxicology/K250803.pdf	Sept. 7,
LabDocs/FDA/Toxicology/K250727.pdf	Sept. 7,
LabDocs/FDA/Toxicology/K250067_REVIEW.pdf	Sept. 7,
LabDocs/FDA/Toxicology/K250067.pdf	Sept. 7,

(a) Documents admin



DOCUMENT	CHUNK INDEX
LabDocs/FDA/Microbiology/K152955_REVIEW.pdf	30
LabDocs/FDA/Microbiology/K211342_REVIEW.pdf	84
LabDocs/FDA/Toxicology/K231752.pdf	66
LabDocs/FDA/Clinical Chemistry/K062838_REVIEW.pdf	57
LabDocs/FDA/Clinical Chemistry/K050975_REVIEW.pdf	60
LabDocs/FDA/Microbiology/K223591_REVIEW.pdf	192
LabDocs/FDA/Clinical Chemistry/K113830_REVIEW.pdf	68
LabDocs/FDA/Clinical Chemistry/K153361_REVIEW.pdf	27
LabDocs/FDA/Microbiology/K091053_REVIEW.pdf	36
LabDocs/FDA/Microbiology/K213931_REVIEW.pdf	93
LabDocs/FDA/Hematology/K062756.pdf	36
LabDocs/FDA/Clinical Chemistry/K161954.pdf	68
LabDocs/FDA/Clinical Chemistry/K123983.pdf	47
LabDocs/FDA/Toxicology/K182654_REVIEW.pdf	126

(b) Chunks admin

Figure 8: Django admin interface for browsing the database.