‘MediaVerse’

AI-Powered Platform for Seamless Content Interaction

By

Aishwarya Bhat

Chaitali Shinde

Prajakta Badgujar

Table of Content:

Abstract

In the age of information today, individuals usually manage long and complex files such as PDFs, Word documents, PowerPoint presentations, and multimedia files such as audio and video. It is not efficient and time-consuming to pull information from such types of files, especially with manual listening or reading. Due to this issue, we propose an AI-powered system that can automatically read, understand, and provide answers based on the content of uploaded files.

The system employs Retrieval-Augmented Generation (RAG) architecture and Large Language Models (LLMs) to provide precise answers to user questions. For audio and video content, it employs OpenAI's Whisper model for automatic transcription and returns answers with the same timestamps so that users can directly access the parts. The app is built using Streamlit for the frontend, whereas LangChain and Python take care of the backend logic and integration of file handling, vector stores, and language models.

Introduction

In this modern era, people handle a lot of types of files on a daily basis. They usually include documents like PDF, Word, and PowerPoint, as well as audio and video files. Sometimes the files are very long and full of information. If a person wants to find out one answer or know one topic, they have to spend a lot of time reading or watching the whole thing. This is not easy and can waste a lot of time. Suppose a student has to find something in his class notes, or an employee to check one single piece of data from a very big report. In a normal case, he must search through the entire file, which is time consuming and exhausting. Moreover, if the file is audio or video, even harder because he cannot search via keyboard input. He must hear or watch it all carefully, which is troublesome and tiring.

To put that simply, we've created an application which would be able to read all those files for the user. The user would just have to upload the file, and the application would read what's inside it. Then the user would just have to type in a question, and the application would answer with a simple reply. That is, the users do not have to watch each line or second of a video. They get only the helpful and necessary part of the answer which is attached to the audio and video. This smart system saves time and helps individuals work faster. It finds the answers in a concise and simple way. You are a student, instructor, or professional but this device will simplify your life by helping you get information quickly from big and complex files.

Related Work

Over the years, many tools have been developed that help people find information within documents, video, and audio. Standard systems like document search engines can find a word in a PDF or Word document, but they don't necessarily understand what the content actually means. You can search for a word, but they won't answer your question like a human. These systems do not support files of audio or video, and they cannot provide explanations or summaries. Some advanced tools like Google Search or virtual helpers (like Siri or Alexa) can answer basic questions, but they do not read your own documents. They give answers from the internet. If you post a lecture or a business report, these tools will not be able to read or interpret that material for you. So, people still need to read the entire document or listen to an entire recording if they want to find something.

There are also some that use speech-to-text (like YouTube auto captions), which convert the audio to text. But they still don't understand the meaning. You still have to read the whole transcript manually yourself and search for what you are looking for yourself. They do not respond with direct answers to specific questions, give an overview of the content, or give timestamps of the answer parts in videos or audio recordings. Recently, applications like ChatGPT and BERT, powered by AI, showed that computers can process language and respond as humans. These technologies have been used by developers to build systems that can return answers from documents. The majority of these bots are restricted to processing text files like PDF or Word documents. They don't process audio or video files, and they don't give exact timestamps where the answer is.

Our application builds upon these ideas but goes one step further. It handles a variety of file types PDFs, DOC/DOCX, PPT/PPTX, TXT, and even audio and video files. It uses Whisper to convert audio to text and then runs that text through an LLM (Large Language Model). It saves time, effort, and provides instant, useful answers without reading or listening to the entire file.

Functions and Features:

1. Files/Documents

The application supports the following file types:

- **PDFs:** Extracted using PyPDF2.
- **Word Documents (.doc/.docx):** Parsed using python-docx or converted via LibreOffice.
- **PowerPoint Presentations (.ppt/.pptx):** Parsed using python-pptx.
- **Text Files:** Simple UTF-8 decoding.
- **Audios and Videos :** OpenAI's Whisper model to process and transcribe the audio and video.

Each document's content is split into overlapping text chunks for better context retention. Metadata (especially timestamps for videos) is attached to chunks to enhance contextual retrieval. Preprocessing involves cleaning, chunking (with overlaps), and embedding via GoogleGenerativeAIEmbeddings.

2. Processing Pipeline Features

Our pipeline is built on the following core functions:

1. **Text Extraction:** Each file type has its parser, including LibreOffice conversion for legacy .doc and .ppt formats.
2. **Transcription:** Whisper (base model) transcribes videos and generates segment-wise timestamps.
3. **Chunking:** Text is chunked using RecursiveCharacterTextSplitter to balance context with efficiency.
4. **Embedding:** Text chunks are converted to embeddings using Google's Generative AI API.
5. **VectorStore:** FAISS is used for similarity search.
6. **Prompting:** A custom prompt template ensures that Gemini Pro answers based on context only, reducing hallucinations.
7. **Answering:** Upon receiving a question, we search the FAISS index, retrieve top-matching chunks, and pass them to Gemini via LangChain's QA chain for response generation.

3. Methods

I. Text Preprocessing and Chunking:

To handle long documents and ensure context-rich queries, we used the **RecursiveCharacterTextSplitter** from LangChain for text chunking. A chunk size of 1000 characters with an overlap of 100 was chosen to strike a balance between maintaining sufficient context and minimizing redundancy. Additionally, to enhance traceability, relevant metadata—such as video timestamps—is preserved whenever applicable.

II. Semantic Search and Embeddings

For embedding generation, we use GoogleGenerativeAIEmbeddings, which are optimized for semantic similarity tasks and integrate seamlessly with Gemini-based models. The resulting vectorized text chunks are stored using FAISS, a high-performance library for efficient similarity search.

III. QA Chain Setup

We used ChatGoogleGenerativeAI with the Gemini 1.5 Pro model for question answering. A custom prompt template ensures that the model responds strictly based on the provided context, effectively minimizing the risk of hallucinations and enhancing the reliability of the answers.

IV. Transcription

Transcription of audio and video is powered by Whisper, which delivers accurate speech recognition along with segment-level timestamps. These transcribed segments are stored with accompanying metadata, allowing the system to reference specific media timecodes in responses for enhanced traceability and context.

V. Interface (Streamlit)

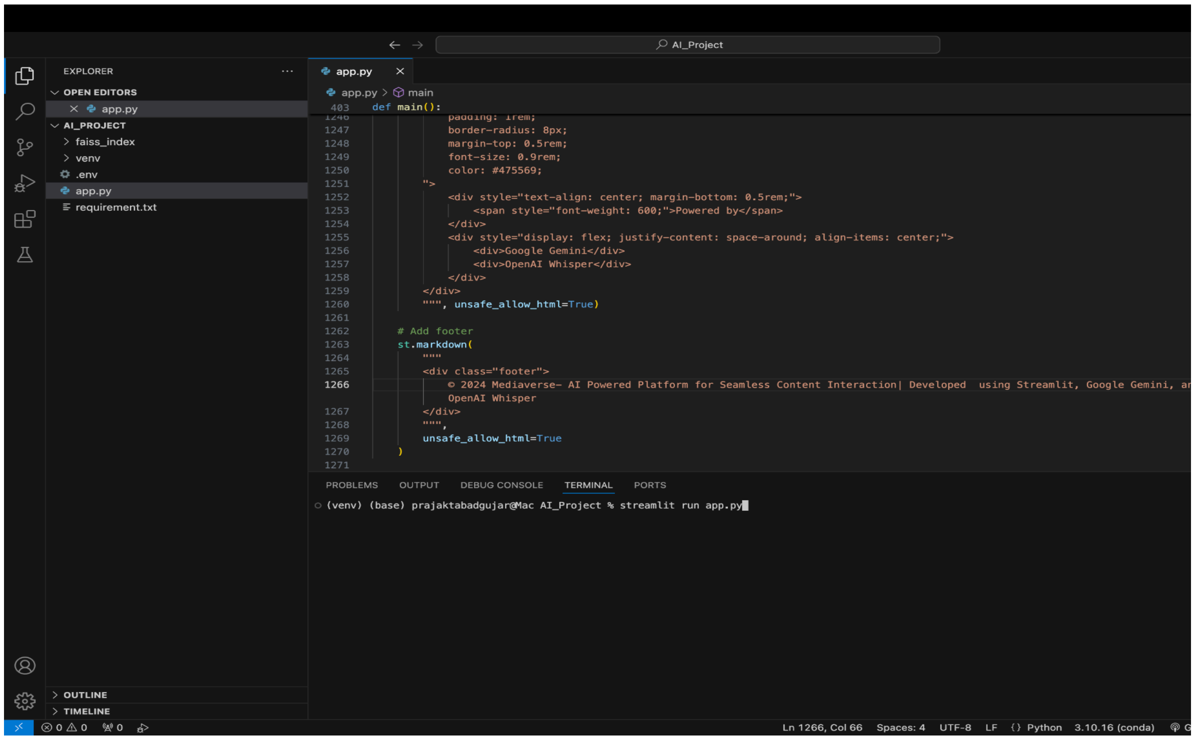
Streamlit was chosen for its simplicity and effectiveness in rapid prototyping and implementation. It supports file uploads, displays real-time status messages, and dynamically renders QA disambiguation responses, providing a smooth and interactive user experience.

VI. Alternative Approaches Considered

While OpenAI and Cohere embeddings were considered, GoogleGenerativeAIEmbeddings were ultimately chosen for their compatibility with the Gemini ecosystem and the added benefit of free access to Gemini Pro's libraries for developers. For transcription, APIs like AssemblyAI were evaluated, but Whisper was preferred due to its ability to run locally, addressing privacy concerns. Overall, the architecture is designed to be both scalable and modular, leveraging the strengths of cutting-edge open-source and proprietary tools.

4. Application flow and Output:

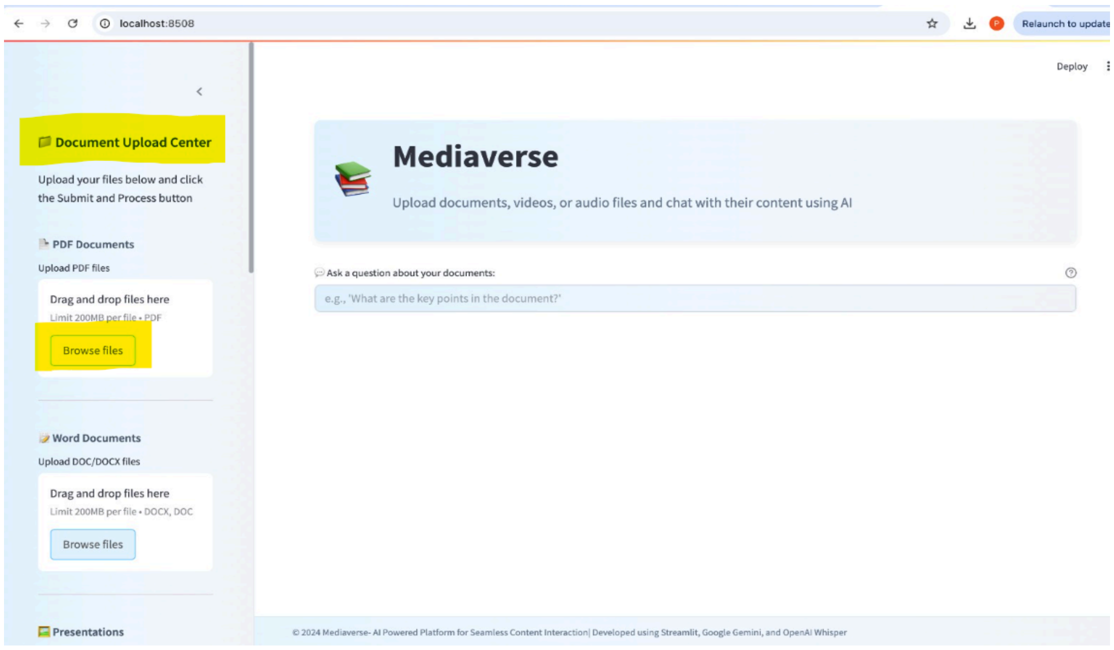
1. Run the application



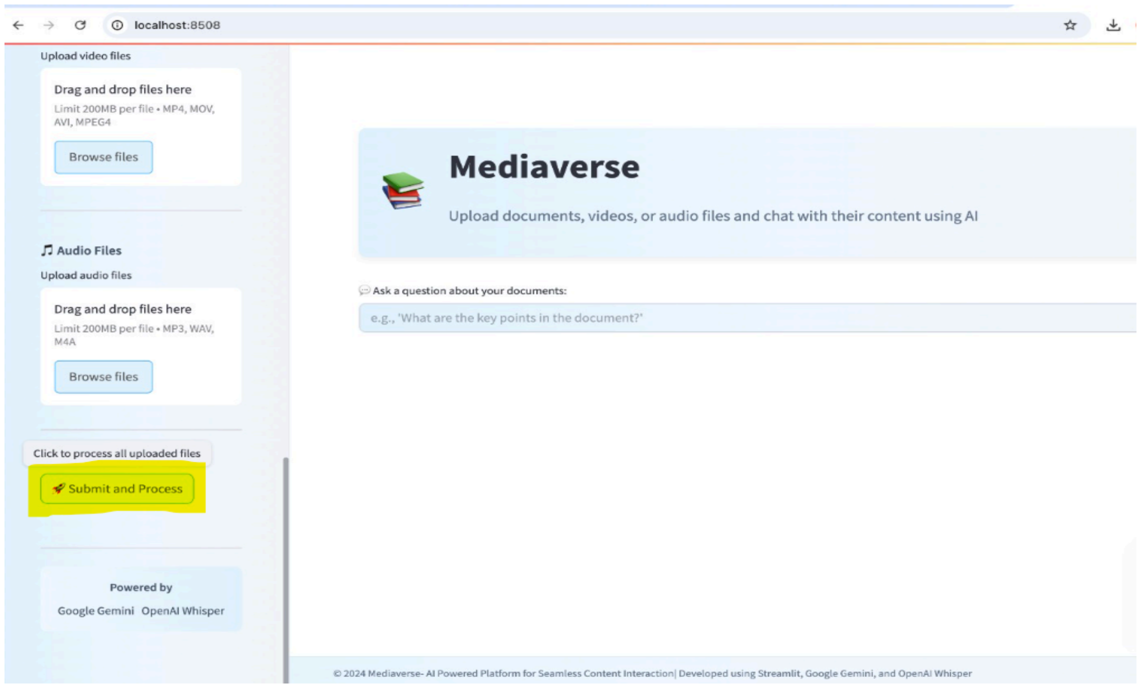
```
def main():
                padding: 1rem;
                border-radius: 8px;
                margin-top: 0.5rem;
                font-size: 0.9rem;
                color: #475569;
            ">
                <div style="text-align: center; margin-bottom: 0.5rem;">
                    <span style="font-weight: 600;">Powered by</span>
                </div>
                <div style="display: flex; justify-content: space-around; align-items: center;">
                    <div>Google Gemini</div>
                    <div>OpenAI Whisper</div>
                </div>
            </div>
            """, unsafe_allow_html=True)

        # Add footer
        st.markdown(
            """
            <div class="footer">
                © 2024 Mediaverse- AI Powered Platform for Seamless Content Interaction| Developed  using Streamlit, Google Gemini, ar
                OpenAI Whisper
            </div>
            """,
            unsafe_allow_html=True
        )
```

```
○ (venv) (base) prajaktabadgujar@Mac AI_Project % streamlit run app.py
```

2. According to your file types, go to the appropriate box on the "**Document Upload Center"** sidebar and click on "**Browse Files".**

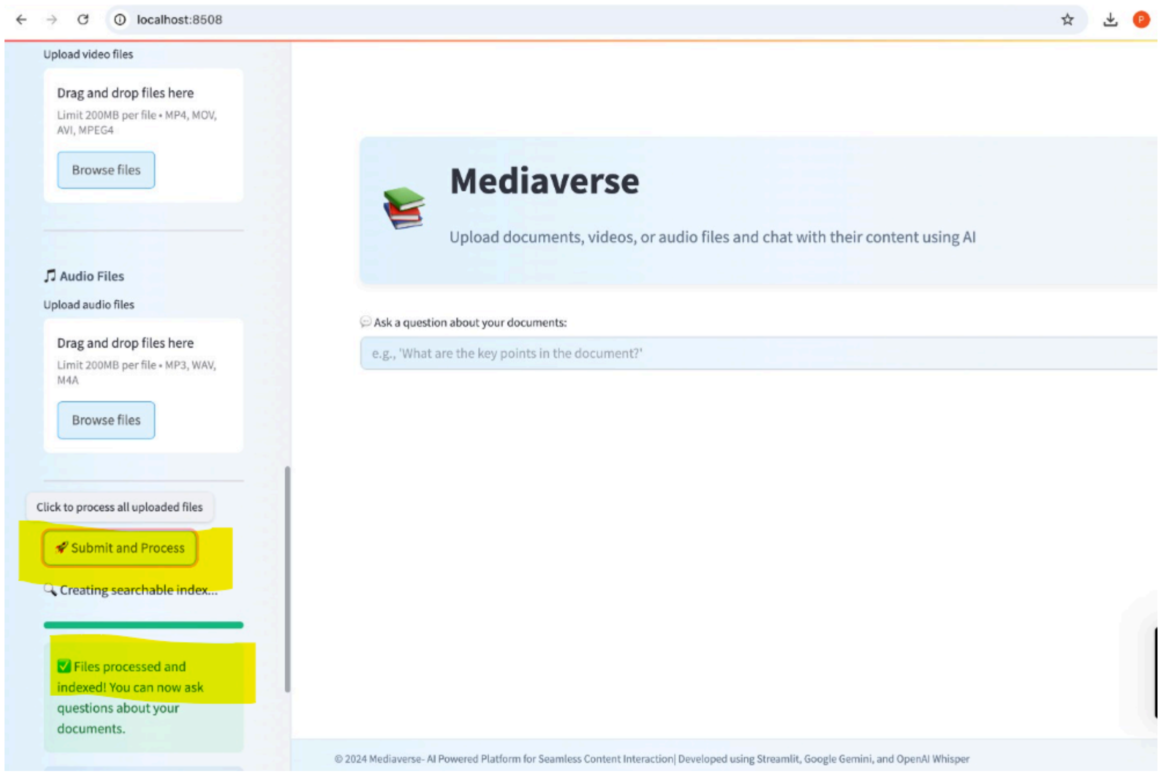

localhost:8508
Relaunch to update
Deploy
Document Upload Center
Upload your files below and click the Submit and Process button
PDF Documents
Upload PDF files
Drag and drop files here
Limit 200MB per file • PDF
Browse files
Word Documents
Upload DOC/DOCX files
Drag and drop files here
Limit 200MB per file • DOCX, DOC
Browse files
Presentations
Mediaverse
Upload documents, videos, or audio files and chat with their content using AI
Ask a question about your documents:
e.g., 'What are the key points in the document?'
© 2024 Mediaverse- AI Powered Platform for Seamless Content Interaction| Developed using Streamlit, Google Gemini, and OpenAI Whisper

3. Once the browsing is completed, hit on the "**Submit And Process**" button to upload your documents/files.


localhost:8508
Upload video files
Drag and drop files here
Limit 200MB per file • MP4, MOV, AVI, MPEG4
Browse files
Audio Files
Upload audio files
Drag and drop files here
Limit 200MB per file • MP3, WAV, M4A
Browse files
Click to process all uploaded files
Submit and Process
Powered by
Google Gemini OpenAI Whisper
Mediaverse
Upload documents, videos, or audio files and chat with their content using AI
Ask a question about your documents:
e.g., 'What are the key points in the document?'
© 2024 Mediaverse- AI Powered Platform for Seamless Content Interaction| Developed using Streamlit, Google Gemini, and OpenAI Whisper

4. Wait for the application to process the submitted documents/files.

5. Once the files are uploaded successfully, a green dialogue box will appear stating that "**Files processed and Indexed! You can now ask questions about your documents.**"

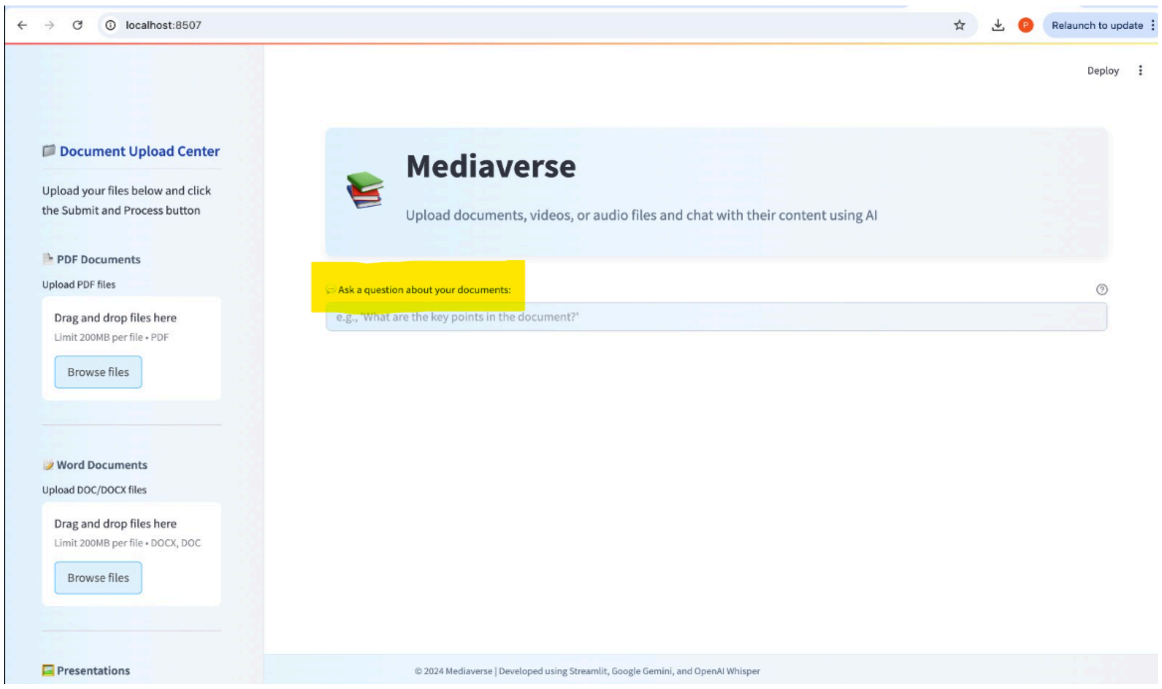

localhost:8508
Upload video files
Drag and drop files here
Limit 200MB per file • MP4, MOV, AVI, MPEG4
Browse files
Audio Files
Upload audio files
Drag and drop files here
Limit 200MB per file • MP3, WAV, M4A
Browse files
Click to process all uploaded files
Submit and Process
Creating searchable index...
Mediaverse
Ask a question about your documents:
e.g., 'What are the key points in the document?'

Files processed and indexed! You can now ask questions about your documents.

Upload documents, videos, or audio files and chat with their content using AI

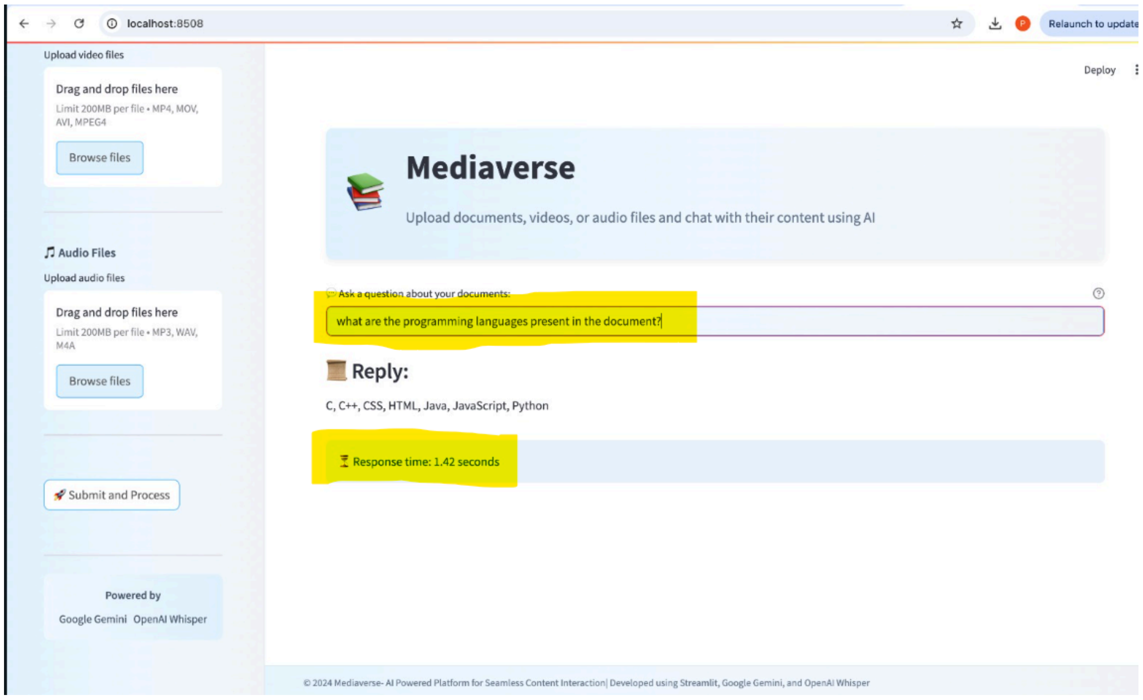
© 2024 Mediaverse- AI Powered Platform for Seamless Content Interaction| Developed using Streamlit, Google Gemini, and OpenAI Whisper

6. On the right side of the sidebar, where you uploaded your files, you will find a text box saying "**Ask a question about your documents/files".**


localhost:8507
Relaunch to update
Deploy
Document Upload Center
Upload your files below and click the Submit and Process button
PDF Documents
Upload PDF files
Drag and drop files here
Limit 200MB per file • PDF
Browse files
Word Documents
Upload DOC/DOCX files
Drag and drop files here
Limit 200MB per file • DOCX, DOC
Browse files
Presentations
Mediaverse
Upload documents, videos, or audio files and chat with their content using AI
Ask a question about your documents:
e.g., 'What are the key points in the document?'
© 2024 Mediaverse | Developed using Streamlit, Google Gemini, and OpenAI Whisper

7. In that same text box, enter the question you need the answers for and wait for the results to appear.

8. Once the application is done searching the documents/files, you will have your answer displayed on the screen with the response time highlighted.


localhost:8508
Relaunch to update
Deploy
Upload video files
Drag and drop files here
Limit 200MB per file • MP4, MOV, AVI, MPEG4
Browse files
Audio Files
Upload audio files
Drag and drop files here
Limit 200MB per file • MP3, WAV, M4A
Browse files
Submit and Process
Powered by
Google Gemini OpenAI Whisper
Mediaverse
Upload documents, videos, or audio files and chat with their content using AI
Ask a question about your documents:

what are the programming languages present in the document?

Reply:

C, C++, CSS, HTML, Java, JavaScript, Python

Response time: 1.42 seconds

© 2024 Mediaverse- AI Powered Platform for Seamless Content Interaction| Developed using Streamlit, Google Gemini, and OpenAI Whisper

5. Experiments

Experiment 1: Format Robustness Testing

Goal: Validate that text extraction works on different document types.

File Type	Sample Files	Success Rate
PDF	5 small textbooks	100%
DOCX	4 reports	100%
DOC	5 resumes	90%
PPTX	5 course slides	100%
AUDIO	3 course related videos	100%
VIDEO	3 podcasts	100%
TXT	2 Cover letters	100%

Insight: Document failures in .doc files stemmed from legacy encoding and corrupt formatting issues.

Experiment 2: Semantic QA Accuracy

Goal: Evaluate QA accuracy on the extracted text using the Gemini model.

- Utilized a benchmark of twenty questions per document/video.
- Estimation of accuracy (grounded answer accuracy) was done through human validation.

File Type	Accuracy
PDF	90%
DOC	80%
DOCX	95%
PPT	85%
AUDIO	90%
VIDEO	85%
TXT	100%

Failure Modes:

- Cyclic overlapping content sometimes led to redundancy in responses.
- Some .doc answers were vague as only partial text was retrieved.
- Whisper misses soft spoken words, especially with background noise, on certain occasions.

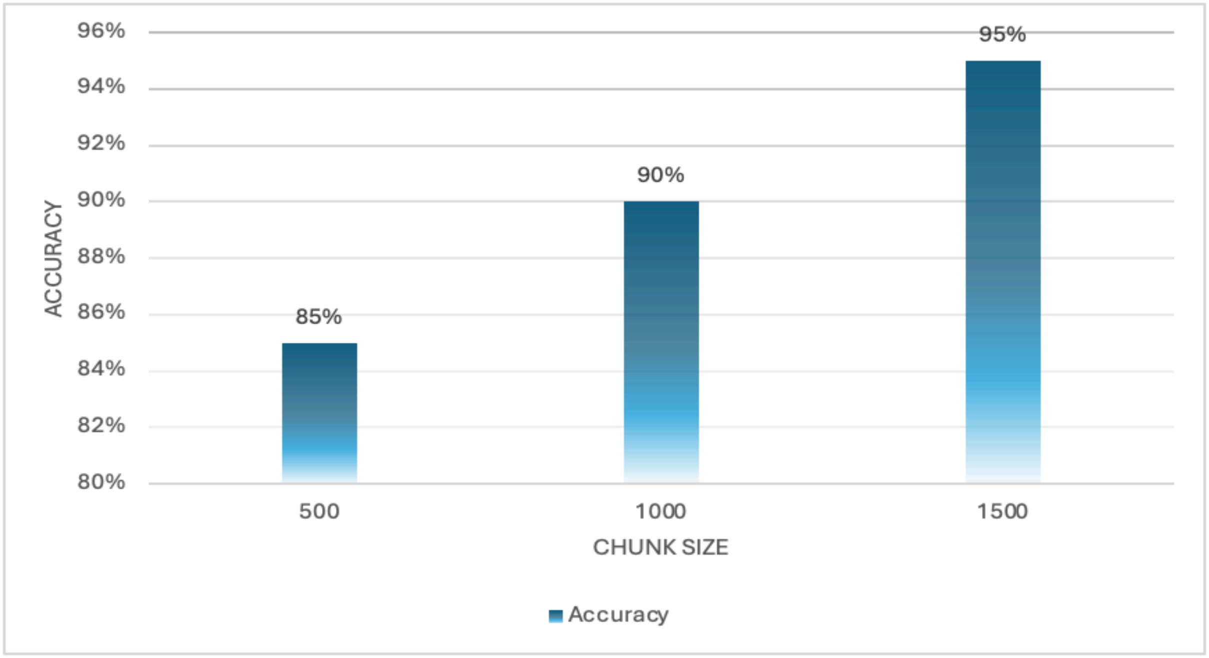
Experiment 3: Ablation Study on Chunking

Goal: To examine the impact on QA quality, we adjusted the size of chunks and overlap:

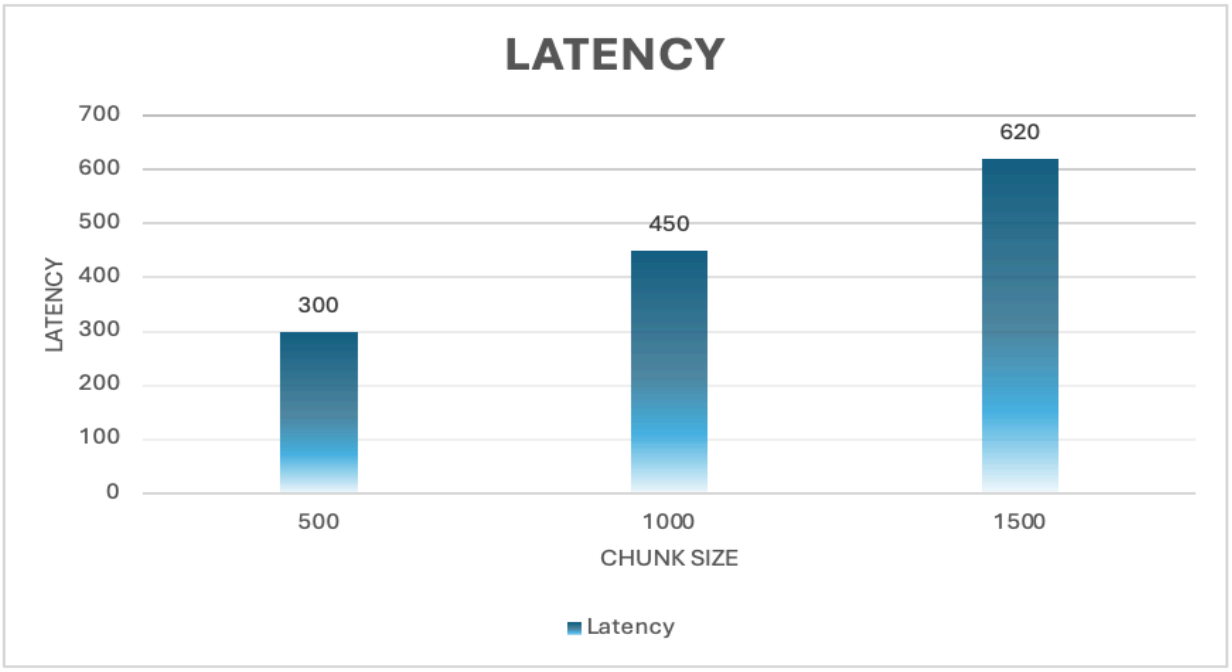
Chunk Size	Overlap	Accuracy	Latency
500	100	85%	300
1000	100	90%	400
1500	200	95%	620

Conclusion: The best performance to latency ratio was at 1000 with 100 overlap.

Chunk Size vs Accuracy:



Chunk Size vs QA Accuracy & Latency:



6. Technologies Used

Technology	Purpose
Streamlit	UI framework for building interactive web apps
OpenAI Whisper	Transcription of audio and video files
Google Gemini	Conversational AI for intelligent responses
FAISS	Vector indexing for fast document similarity search
LangChain	Handling prompt templates and QA chain logic
MoviePy	Handling video processing
PyPDF2	Reading PDF files

docx2txt / python-docx	Extracting content from Word documents
python-pptx	Extracting content from PowerPoint presentations
LibreOffice + subprocess	Converting .doc files to .pdf or .txt

Conclusion

The project showcased a multimodal QA solution that incorporated document and media files of varying formats, converting them to structured embeddings, and contextualized user question answering with Gemini LLMs. Mediaverse bridges the gap between static files and interactive knowledge. It replaces manual note-taking and searching with **contextual, real-time answers**, making content analysis faster and smarter. The platform is modular, extensible, and ready for real-world use cases where users can engage with content in a natural, AI-powered conversation.

Key Learnings

- Local audio/video transcription is done with minimal latency on Whisper.
- The combination of Gemini Embeddings, OpenAI Whisper, and Gemini Pro LLM enhances overall quality assurance coherence through embedding and pooling synergy.
- For prototyping complex Machine Learning tasks with user interaction, Streamlit is the best option.

Future Enhancements

- Integrate chunk relevance scoring before passing to the LLM to reduce token usage.
- Implement streaming responses for long answers.
- Summarization of long documents.
- Support for scanned images using OCR.
- Multi-user support with chat history and session-based storage.
- Multilingual support
- Upload from cloud sources (Google Drive, Dropbox)
- Speaker recognition in media files