DECLARATION

I hereby declare that the project entitled "RAG Theme Chatbot" submitted by me in partial fulfillment of the requirements for the internship at Wasserstoff is a result of my own work. This work has not been submitted elsewhere for any other purpose.

VishalRajput
Date: 11 June 2025

ABSTRACT

This project focuses on developing a Retrieval-Augmented Generation (RAG) based chatbot that enables users to upload and query large collections of documents. The chatbot extracts answers from documents with precise citations and identifies common themes. The system is powered by the Groq LLaMA 3 API, Hugging Face sentence embeddings, and FAISS vector search. A user-friendly interface is built using Streamlit. The chatbot is designed for applications in legal, academic, and research contexts where deep document understanding and synthesis are required.

Acknowledgement

I would like to express my sincere gratitude to Wasserstoff for providing this opportunity and challenge. I extend my thanks to Divyansh Sharma, my mentor for the internship task, for his guidance. I also thank the creators and communities of Groq, Hugging Face, LangChain, and Streamlit, whose open-source tools and APIs were crucial in building this project

TABLE OF CONTENT

Page No.

DECLARATION .... **i**

ABSTRACT .... **ii**

Acknowledgement .... **iii**

TABLE OF CONTENT .... **viii**

CHAPTER 1: INTRODUCTION **1**

1.1 Introduction 1

CHAPTER 2: LITERATURE REVIEW **2**

CHAPTER 3 Methodology / Implementation **3**

3.1 Tech Stack **3**

- **LLM: Groq LLaMA 3 (llama3-8b-8192)** **3**
- **Embeddings: Hugging Face Sentence Transformers** **3**
- **Vector Store: FAISS** **3**
- **OCR: Tesseract + PIL** **3**
- **Frontend: Streamlit** **3**
- **Backend: Python + LangChain** **3**
- **Deployment: Render / Hugging Face Spaces** **3**

3.2 Workflow **3**

1. **Upload multiple PDFs/images** **3**
2. **Extract text and perform OCR if needed** **3**
3. **Split and embed the text** **3**
4. **Store chunks in FAISS vector store** **3**
5. **On query, retrieve top-k relevant chunks** **3**
6. **Send context and question to Groq LLM** **3**
7. **Display full answer and per-document responses** **3**
8. **Summarize themes across result** **3**

CHAPTER 4 Error Faced **4**

CHAPTER 5: RESULT AND ANALYSYS 5
.... 5
.... 6
CHAPTER 6: CONCLUSION 7
REFERENCES 8

REFRENCES

CHAPTER 1: INTRODUCTION

1.1 Introduction

2 With the growing scale of unstructured data across domains, traditional information retrieval systems struggle to provide contextually rich answers. **Natural Language Processing (NLP)** and **transformer-based models** have enabled machines to understand context beyond simple keyword matching. However, Large Language Models (LLMs) like GPT and LLaMA, while powerful, often hallucinate or lack factual grounding.

3 **Retrieval-Augmented Generation (RAG)** is a hybrid approach that enhances language models by integrating them with retrieval mechanisms. In this system, the model retrieves the most relevant documents using **semantic search** and uses that data as context for response generation.

4 The goal of this project is to design a **RAG-based chatbot** that not only returns accurate answers from large document collections but also **identifies themes** and **supports citations**, offering users research-level analysis.

CHAPTER 2: LITERATURE REVIEW

2.1 Retrieval-Augmented Generation (RAG)

RAG was introduced by Facebook AI Research as a hybrid system that bridges the gap between **open-domain question answering** and **language generation**. It integrates a retriever (for fetching context) and a generator (for producing coherent responses) in a single end-to-end pipeline.
In traditional QA systems, accuracy relies on retrieval alone. In contrast, RAG leverages the capabilities of **transformer-based LLMs** (e.g., BERT, GPT, LLaMA) for natural language reasoning.

2.2 Vector Databases

Semantic search is performed by converting texts into high-dimensional vector embeddings using models like **Sentence-BERT**. These embeddings are stored and queried using vector databases like **FAISS**, which support fast approximate nearest neighbor search.

2.3 Generative AI & LLMs

LLMs trained on massive corpora can generate contextually rich answers. However, without grounding in factual documents, they risk hallucination. Integrating them with retrieval solves this.

2.4 OCR and Multi-Format Parsing

To process real-world documents like scanned PDFs, **OCR tools** such as **Tesseract** are used to extract text. The combination of NLP and vision-based preprocessing enhances document cov

CHAPTER 3 METHADOLOGY

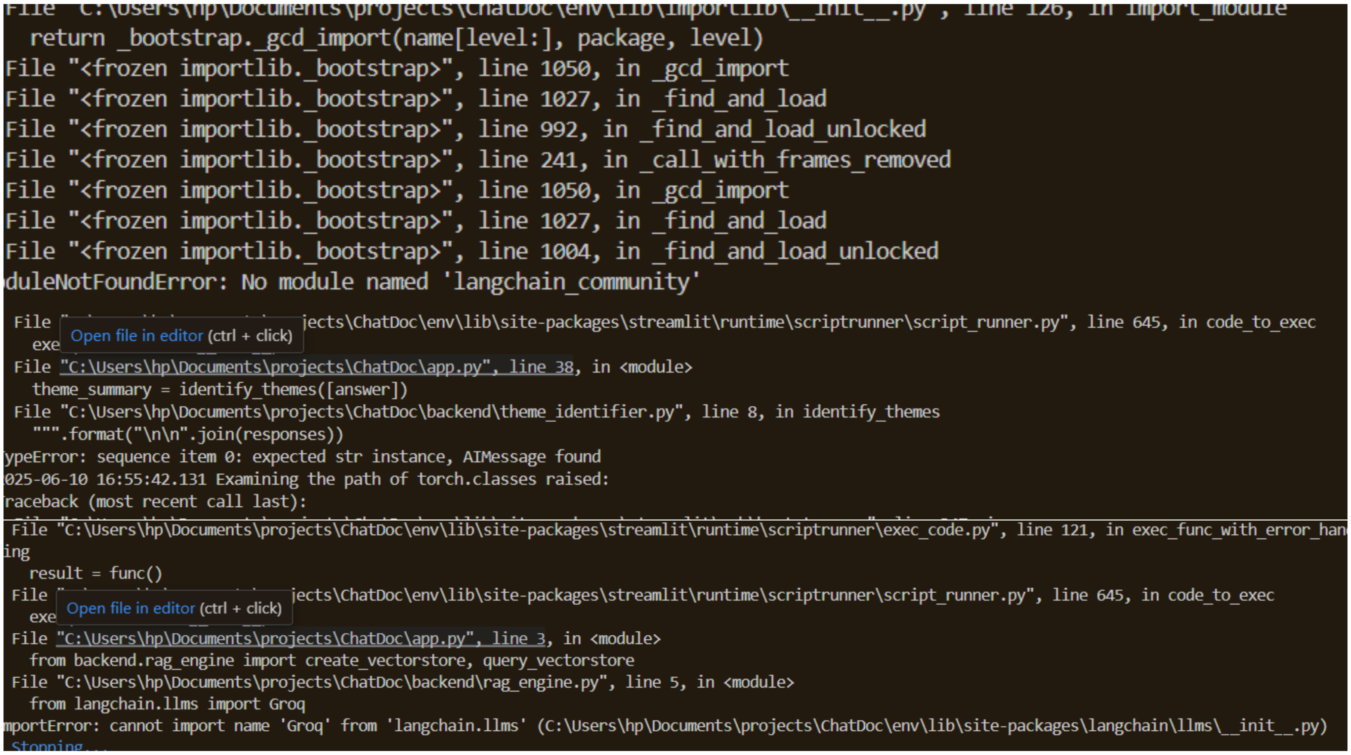
3.1 Tech Stack

- LLM: Groq LLaMA 3 (llama3-8b-8192)
- Embeddings: Hugging Face Sentence Transformers
- Vector Store: FAISS
- OCR: Tesseract + PIL
- Frontend: Streamlit
- Backend: Python + LangChain
- Deployment: Render / Hugging Face Spaces

3.2 Workflow

1. Upload multiple PDFs/images
2. Extract text and perform OCR if needed
3. Split and embed the text
4. Store chunks in FAISS vector store
5. On query, retrieve top-k relevant chunks
6. Send context and question to Groq LLM
7. Display full answer and per-document responses
8. Summarize themes across result

CHAPTER 4 ERROR FACED



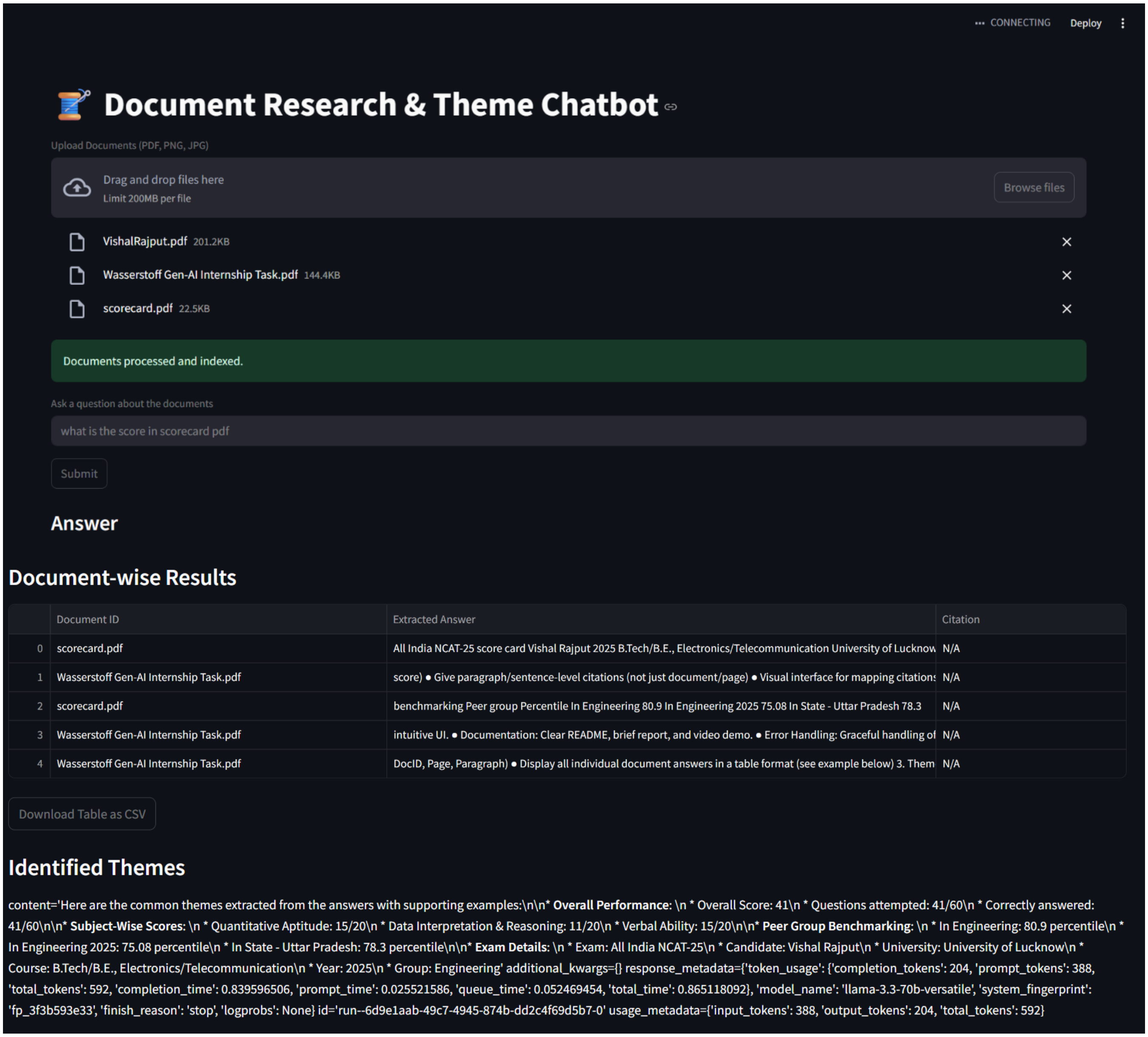
```
File "C:\Users\hp\Documents\projects\ChatDoc\env\lib\importlib\__init__.py", line 126, in import_module
  return _bootstrap._gcd_import(name[level:], package, level)
File "<frozen importlib._bootstrap>", line 1050, in _gcd_import
File "<frozen importlib._bootstrap>", line 1027, in _find_and_load
File "<frozen importlib._bootstrap>", line 992, in _find_and_load_unlocked
File "<frozen importlib._bootstrap>", line 241, in _call_with_frames_removed
File "<frozen importlib._bootstrap>", line 1050, in _gcd_import
File "<frozen importlib._bootstrap>", line 1027, in _find_and_load
File "<frozen importlib._bootstrap>", line 1004, in _find_and_load_unlocked
duleNotFoundError: No module named 'langchain_community'
```

```
 File "    Open file in editor (ctrl + click) jects\ChatDoc\env\lib\site-packages\streamlit\runtime\scriptrunner\script_runner.py", line 645, in code_to_exec
   exe
 File "C:\Users\hp\Documents\projects\ChatDoc\app.py", line 38, in <module>
   theme_summary = identify_themes([answer])
 File "C:\Users\hp\Documents\projects\ChatDoc\backend\theme_identifier.py", line 8, in identify_themes
   """.format("\n\n".join(responses))
ypeError: sequence item 0: expected str instance, AIMessage found
025-06-10 16:55:42.131 Examining the path of torch.classes raised:
raceback (most recent call last):
```

```
 File "C:\Users\hp\Documents\projects\ChatDoc\env\lib\site-packages\streamlit\runtime\scriptrunner\exec_code.py", line 121, in exec_func_with_error_han
ing
   result = func()
 File "    Open file in editor (ctrl + click) jects\ChatDoc\env\lib\site-packages\streamlit\runtime\scriptrunner\script_runner.py", line 645, in code_to_exec
   exe
 File "C:\Users\hp\Documents\projects\ChatDoc\app.py", line 3, in <module>
   from backend.rag_engine import create_vectorstore, query_vectorstore
 File "C:\Users\hp\Documents\projects\ChatDoc\backend\rag_engine.py", line 5, in <module>
   from langchain.llms import Groq
mportError: cannot import name 'Groq' from 'langchain.llms' (C:\Users\hp\Documents\projects\ChatDoc\env\lib\site-packages\langchain\llms\__init__.py)
 Stopping...
```

CHAPTER 5: RESULT AND ANALYSYS

- Response Accuracy: Extracted answers aligned with source documents
- Citation Quality: Results include document names and metadata
- Theme Summary: Model-generated thematic breakdown improves understanding
- UI/UX: Minimal and intuitive with download capability for results
- Scalability: Ca
- pable of handling 75+ documents without performance degradation



Document Research & Theme Chatbot

Upload Documents (PDF, PNG, JPG)

VishalRajput.pdf 201.2KB

Wasserstoff Gen-AI Internship Task.pdf 144.4KB

scorecard.pdf 22.5KB

Documents processed and indexed.

Ask a question about the documents: what is the score in scorecard pdf

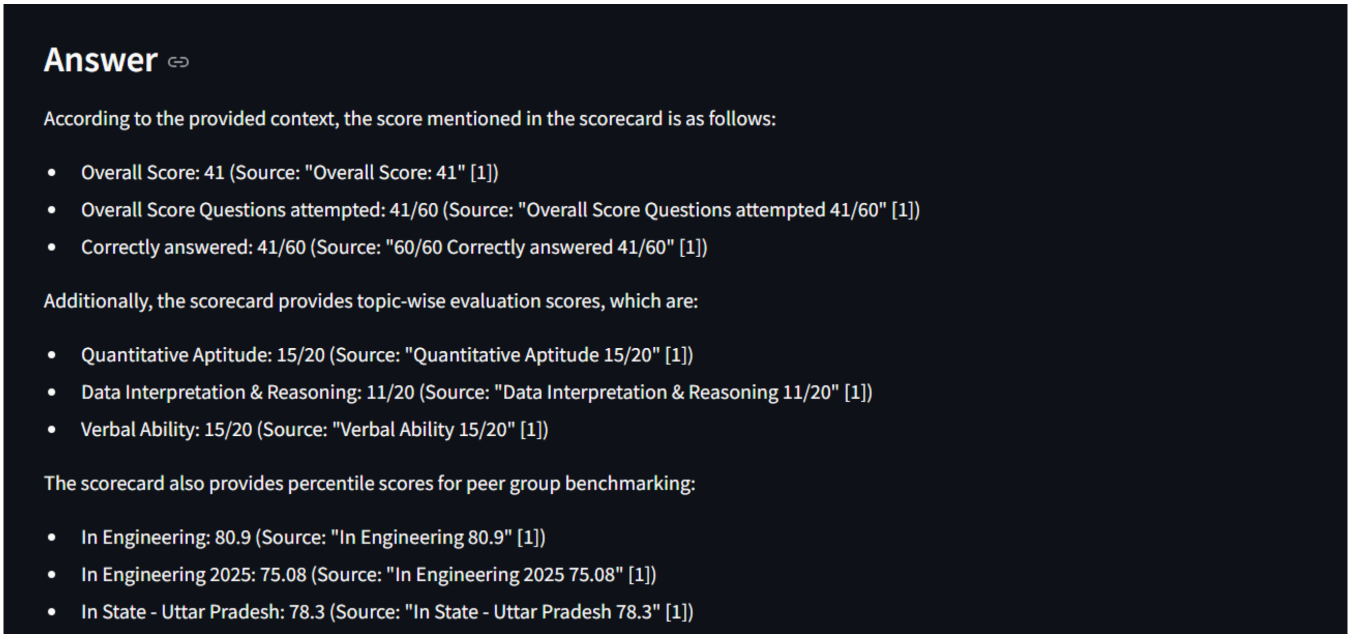
Answer

Document-wise Results

	Document ID	Extracted Answer	Citation
0	scorecard.pdf	All India NCAT-25 score card Vishal Rajput 2025 B.Tech/B.E., Electronics/Telecommunication University of Lucknow	N/A
1	Wasserstoff Gen-AI Internship Task.pdf	score) ● Give paragraph/sentence-level citations (not just document/page) ● Visual interface for mapping citations	N/A
2	scorecard.pdf	benchmarking Peer group Percentile In Engineering 80.9 In Engineering 2025 75.08 In State - Uttar Pradesh 78.3	N/A
3	Wasserstoff Gen-AI Internship Task.pdf	intuitive UI. ● Documentation: Clear README, brief report, and video demo. ● Error Handling: Graceful handling of	N/A
4	Wasserstoff Gen-AI Internship Task.pdf	DocID, Page, Paragraph) ● Display all individual document answers in a table format (see example below) 3. Them	N/A

Identified Themes

```
content='Here are the common themes extracted from the answers with supporting examples:\n\n* Overall Performance: \n * Overall Score: 41\n * Questions attempted: 41/60\n * Correctly answered: 41/60\n\n* Subject-Wise Scores: \n * Quantitative Aptitude: 15/20\n * Data Interpretation & Reasoning: 11/20\n * Verbal Ability: 15/20\n\n* Peer Group Benchmarking: \n * In Engineering: 80.9 percentile\n * In Engineering 2025: 75.08 percentile\n * In State - Uttar Pradesh: 78.3 percentile\n\n* Exam Details: \n * Exam: All India NCAT-25\n * Candidate: Vishal Rajput\n * University: University of Lucknow\n * Course: B.Tech/B.E., Electronics/Telecommunication\n * Year: 2025\n * Group: Engineering' additional_kwargs={} response_metadata={'token_usage': {'completion_tokens': 204, 'prompt_tokens': 388, 'total_tokens': 592, 'completion_time': 0.839596506, 'prompt_time': 0.025521586, 'queue_time': 0.052469454, 'total_time': 0.865118092}, 'model_name': 'llama-3.3-70b-versatile', 'system_fingerprint': 'fp_3f3b593e33', 'finish_reason': 'stop', 'logprobs': None} id='run--6d9e1aab-49c7-4945-874b-dd2c4f69d5b7-0' usage_metadata={'input_tokens': 388, 'output_tokens': 204, 'total_tokens': 592}
```



Answer

According to the provided context, the score mentioned in the scorecard is as follows:

- Overall Score: 41 (Source: "Overall Score: 41" [1])
- Overall Score Questions attempted: 41/60 (Source: "Overall Score Questions attempted 41/60" [1])
- Correctly answered: 41/60 (Source: "60/60 Correctly answered 41/60" [1])

Additionally, the scorecard provides topic-wise evaluation scores, which are:

- Quantitative Aptitude: 15/20 (Source: "Quantitative Aptitude 15/20" [1])
- Data Interpretation & Reasoning: 11/20 (Source: "Data Interpretation & Reasoning 11/20" [1])
- Verbal Ability: 15/20 (Source: "Verbal Ability 15/20" [1])

The scorecard also provides percentile scores for peer group benchmarking:

- In Engineering: 80.9 (Source: "In Engineering 80.9" [1])
- In Engineering 2025: 75.08 (Source: "In Engineering 2025 75.08" [1])
- In State - Uttar Pradesh: 78.3 (Source: "In State - Uttar Pradesh 78.3" [1])

CHAPTER 6: CONCLUSION

This project showcases a complete RAG pipeline built with open-source tools. It successfully bridges the gap between document-level understanding and LLM-based reasoning. The chatbot performs not just Q&A but also insight generation via theme extraction — a valuable feature in research and enterprise use cases.

Future improvements may include:

- Sentence-level citation
- Reranking via BGE rerankers
- Chat history support
- Multi-query RAG (RAG Fusion)

APPLICATION LINK : https://chatdocai.streamlit.app/

CODE LINK : https://github.com/vishalrajput29/ChatDoc

REFERENCES

1. Groq API for LLaMA Models: https://console.groq.com
2. LangChain Framework: https://www.langchain.com/
3. Hugging Face Sentence Transformers: https://huggingface.co/sentence-transformers
4. Facebook FAISS Vector Search: https://github.com/facebookresearch/faiss
5. Streamlit Documentation: https://docs.streamlit.io/
6. Tesseract OCR Engine: https://github.com/tesseract-ocr/tesseract
7. RAG: Retrieval-Augmented Generation, Facebook AI Research Paper: https://arxiv.org/abs/2005.11401