FINCOMBOT AI CHATBOT

COLLABORATORS

- *Agnes Chomba - Data Scientist/ Scrum Master*
- *Judah Odida- Data Scientist*
- *Olgah Omollo – Data Scientist*
- *Nick Mwai – Data Scientist*
- *Derrick Malinga- Data Scientist*
- *Lucas Ominde – Data Scientist*
- *Eric Okacha- Data Scientist*

FINCOMBOT COMPLIANCE CHATBOT

FinComBot is an AI-powered compliance chatbot designed to streamline customer onboarding and strengthen regulatory compliance in financial institutions. It achieves this by providing staff with instant, policy-aligned answers to KYC and AML-related queries. The project will follow a phased implementation approach, beginning with a pilot focused on KYC and onboarding procedures.

BUSINESS OBJECTIVES

- Build a chatbot the retrieves accurate compliance information from the bank`s Know your Customer(KYC), Anti Money Laundering(AML), Counter Proliferation Financing (CPF) policies and responds to staff queries
- This aims to address several challenges in financial institutions
    - Delays in onboarding
    - Inconsistent application of compliance procedures
    - Overdependence on compliance officers for basic guidance
    - Increased risk of regulatory breaches which may lead to fines and license suspension

TARGET AUDIENCE

- Front office/Relationship Managers
- Operations Staff
- Compliance officers
- New Staff
- Risk and Audit teams

DATA UNDERSTANDING

Data Source:
•Internal **Compliance Policy Document** stored in **Word (.docx)** format.

Content Overview:
•Includes **KYC procedures**, **AML red flags**, **Risk Rating Methodology**, and **Regulatory Guidelines** from **FATF**, **CBK**, and **CMA**.

Data Characteristics:
•**Unstructured text** – paragraphs, checklists, and detailed descriptions.
•Organized into **multiple sections** covering **policies, procedures, and workflows**.

DATA PREPROCESSING

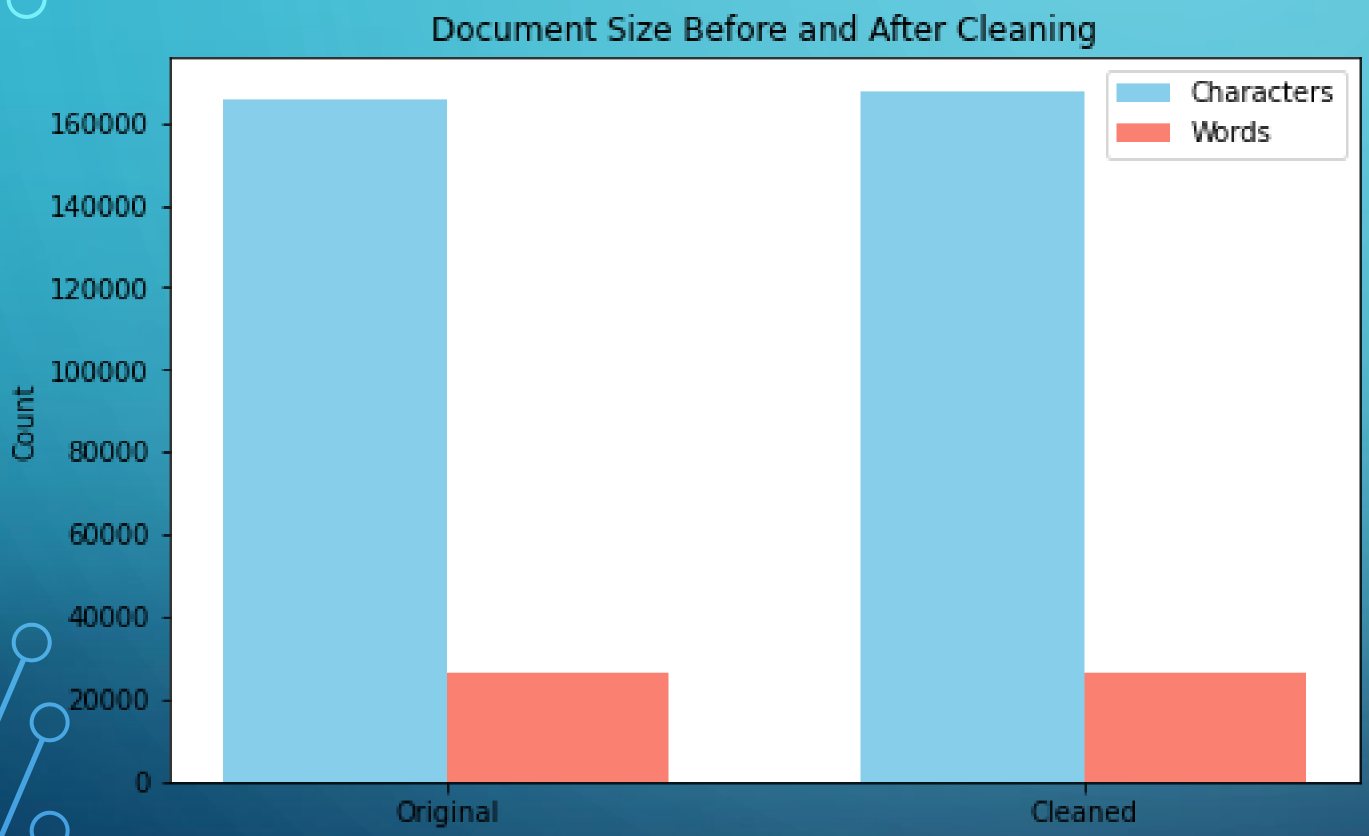
i. Cleaning & Structuring: Fixed encoding issues, removed line breaks, and organized text for readability.

ii. Removing Noise: Eliminated unwanted symbols, smart quotes, and page numbers.

iii. Standardizing Formats: Unified casing, spacing, and punctuation for consistency.

iv. Tokenizing Content: Split text into smaller chunks for efficient embedding and retrieval.

EXPLATORY DATA ANALYSIS



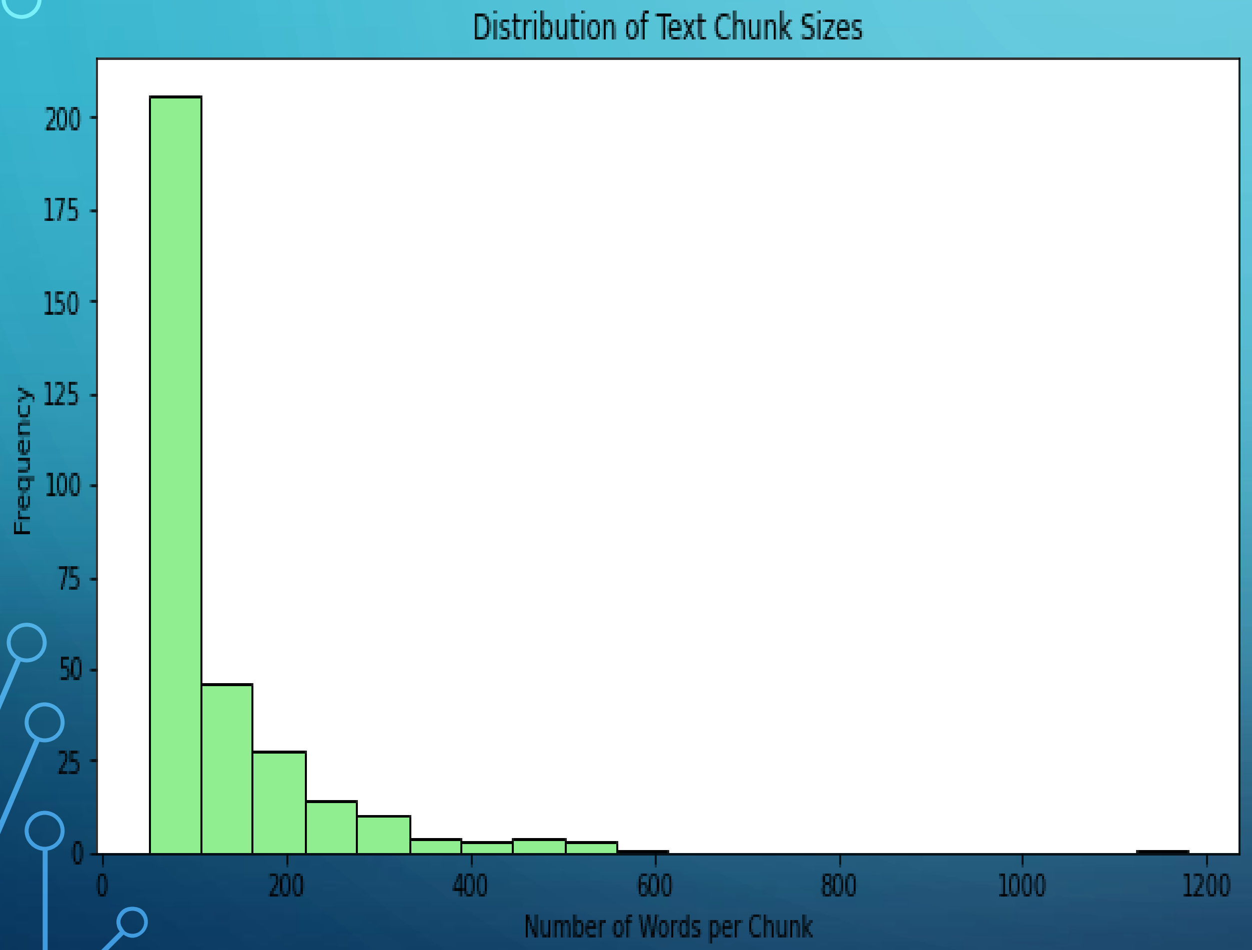
We visualized our document size before and after cleaning.

The difference is insignificant, meaning our data sources was clean.

Our Metrics are Characters and Words

Metric	Original	Cleaned- ----
Characters	165696	167563
Words	26098	26098

SPLITTING TEXT INTO CHUNKS VISUALIZATION-HISTOGRAM



-The chunk size distribution is centered around ~500 words, which aligns well with our target for embeddings.

-This indicates a mostly consistent chunking strategy. However, the histogram reveals a tail of smaller chunks (<200 words) and some oversized chunks (>800 words).

-These could impact efficiency: small chunks may not carry enough semantic content, while large ones risk exceeding model context limits.

HANDLING OUTLIERS IN CHUNK SIZES

- Merge very small chunks with their neighbors-This is to avoid low content pieces
- Split very large chunks into smaller ones- This is to stay within the context limits

VECTORIZATION

- In this step cleaned text is transformed into sentence embeddings ,which convert sentences into dense vectors that capture meaning rather than just Words
- The Project uses transformer embeddings with MiniLM to represent the Know Your Customer policy text.
- The document is split into smaller chunks, and each chunk is encoded into a 384-dimensional vector that captures its semantic meaning. These embeddings enable advanced tasks such as semantic search, similarity comparison, and document retrieval, while keeping the process efficient and scalable.

VECTORIZATION CONT`D…

- Indexing /Storing- 320 text chunks were converted into 384-dimensional embeddings, all stored in the FAISS index. The index was saved successfully, meaning the document is ready for fast semantic search and retrieval
- The code pairs each text chunk with its embedding and stores them in a list of dictionaries.

MODEL EVALUATION

- The MiniLM model was used for this project due to its strong performance in capturing semantic meaning while remaining lightweight and efficient

Recall	Precision	Mean Reciprocal Rank
0.56	0.33	0.33

- FAISS indexing was integrated to enable faster and more accurate similarity searches, after which the model was re-evaluated to confirm improved retrieval performance.

Recall	Precision	Mean Reciprocal Rank
0.73	0.40	0.81

- The recall, which is the primary evaluation metric for this project, improved from 0.56 to 0.73 after introducing FAISS Indexing to the MiniLM model.
- Recall is the main evaluation metric as it measures the proportion of relevant documents successfully retrieved. A higher recall ensures model captures as many relevant pieces of information as possible

MODEL EVALUATION CONT

- TF-IDF weighting was then applied to the base MiniLM model to enhance text relevance scoring, after which the model was re-evaluated for performance improvements.

Recall	Precision	Mean Reciprocal Rank
0.64	0.61	0.83

- The model outperformed the base model alone but ranked second to the combination of the base model with FAISS indexing, as shown in the earlier table.
- This highlighted a recall–precision tradeoff, where the chosen model's precision dropped from 0.61 to 0.43 compared to the TF-IDF–enhanced model, despite achieving higher recall.

PROJECT DEPLOYMENT

Deployment Setup: Involved creating key configuration files:

- **requirements.txt** – specifies project dependencies.
- app.py – main application script for model interaction and user queries.
- **Docker file** – defines the container environment for consistent deployment.

Platform

- The model was deployed using Streamlit Cloud, a lightweight and user-friendly platform for hosting machine learning applications.
- Streamlit enables easy deployment, sharing, and user interaction through an intuitive web interface.

Outcome: The setup ensured efficient retrieval performance, real-time query response, and accessible user experience for end users.

RECOMMENDATIONS AND NEXT STEPS

- 1. **Refine Data Segmentation**
- - Break down the compliance manual into **sections, policies, and rules** for more precise retrieval.
- - Work with **compliance officers** to validate **accuracy** of extracted content.

- 2. **Enhance Modeling**
- - Move from simple similarity search to **advanced retrieval models** (e.g., Retrieval-Augmented Generation).
- - Explore **fine-tuned QA models** on compliance-specific data for better contextual answers.

- 3. **Evaluation Framework**
- - Define metrics with domain experts (**precision, recall, relevance**).
- - Conduct **human-in-the-loop evaluation** where staff review chatbot responses.

RECOMMENDATION AND NEXT STEPS…..

- 4. **User Experience & Deployment**
- - Build a **web-based or intranet chatbot** for easy staff access.
- - Integrate into the bank's existing **CRM or core banking systems.**
- - Add feedback mechanisms so staff can flag inaccurate or outdated answers.

- 5. **Scalability & Governance**
- - Expand to cover **other policy documents** (credit manuals, HR policies, risk frameworks).
- - Ensure robust **data privacy, auditability, and compliance with regulatory standards.**

Q & A

THANK YOU