**Tender Summarization System: Document**

**Table of Contents**

1. [Project Overview](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#project-overview)
2. [Technologies Used](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#technologies-used)
3. [System Architecture](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#system-architecture)
4. [Code Flow](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#code-flow)
5. [Key Functional Components](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#key-functional-components)
6. [Deployment Setup](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#deployment-setup)
7. [ERP iFrame Integration](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#erp-iframe-integration)
8. [Troubleshooting](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#troubleshooting)
9. [Future Improvements](https://chatgpt.com/c/688b3793-6050-8013-b2d6-e65055ddce5f#future-improvements)

**Project Overview**

**Purpose**

This project is a **Tender Document Summarizer** that accepts PDF or Word tender files, analyzes them using a **Cohere LLM API**, and generates a detailed structured summary. It is:

* Hosted via **Streamlit Cloud**
* Embedded in an **ERP system via iFrame**
* Connected to an **API** for forwarding summary results

**Users**

* Government institutions, hospital admins, or sales teams reviewing tender specifications
* Any ERP-integrated team requiring quick insights into uploaded tender documents

**Technologies Used**

| **Area** | **Tech** |
| --- | --- |
| Frontend UI | Streamlit |
| Model API | Cohere (command-a-03-2025) |
| NLP/Text Parsing | re, dateutil, docx, PyPDF2 |
| File Handling | BytesIO, requests, Word tables |

**Code Flow**

**1. File Upload**

* Users upload a .pdf or .docx file
* File is parsed with PyPDF2 or python-docx

**2. Logging**

* Logged details: Filename, Size, Text Length, IP, Timestamp

**3. LLM Prompting & Streaming**

* A custom prompt is sent to **Cohere ChatStream API**
* Summary is streamed back live to the user in the interface
* Summary is also stored for later API use

**4. Postprocessing & Table Generation**

* Key fields (e.g., Tender Name, Dates) are extracted from summary text using regex
* Summary is parsed and formatted into a **Word Table** using python-docx

**5. Upload to Backend API**

* File and summary are sent to a server via POST method
* Payload includes:
  + Original file
  + Generated .docx summary
  + Metadata (tender name, type, dates)

**Key Functional Components**

**🔹 stream\_summary\_from\_cohere(text)**

* Sends formatted prompt to Cohere
* Streams back summary in real-time

**🔹 extract\_tender\_info(text)**

* Extracts Tender Name, Tender Type, Start/End Dates using regex and parser

**🔹 generate\_table\_word(summary)**

* Parses markdown summary into sections
* Outputs a structured .docx table format with key-value pairs

**Deployment Setup**

**GitHub Hosting**

* Code is uploaded to a public/private GitHub repository
* Repo includes app.py and optionally requirements.txt

**Streamlit Cloud Deployment**

1. Sign in to [Streamlit Cloud](https://streamlit.io/)
2. Connect your GitHub account and select the repo
3. Define app.py as entry point
4. Deploy

**Access via URL**

* Public Streamlit URL (e.g., https://streamlit.app)

**ERP iFrame Integration**

**Steps:**

1. Embed the Streamlit URL using an iframe:

<iframe src="https://streamlit.app" width="100%" height="1000px" frameborder="0"></iframe>

1. Ensure CORS & cookie isolation settings are compatible
2. On ERP side, add iframe into the relevant dashboard section

**Troubleshooting**

| **Issue** | **Solution** |
| --- | --- |
| Summary not generated | Check API key or Streamlit network error |
| Upload fails | Ensure file format is PDF or DOCX |
| Text too short error | Likely a scanned image or corrupt document |
| POST API fails | Check target API URL and authentication key |

**Future Improvements**

* Add OCR support for image-based PDFs
* Allow selection between short vs long summary
* Enable multilingual support
* Save summaries in local history for each user
* Dashboard for tender insights

**Conclusion**

This Tender Summarizer project provides a smart, simple interface to extract actionable insights from government and institutional tender documents. It's highly accessible, deployed on Streamlit Cloud, integrated with an ERP, and requires no manual model deployment.

Ideal for sales teams, vendors, or technical teams analyzing large volumes of tenders daily.