Financial Intelligence Chatbot

**Overview Summary**

The Financial Intelligence Chatbot helps users analyse financial documents without needing to be tech experts. Upload a financial file (like a spreadsheet or report), ask a question in plain English, and get useful insights.

**Workflow Components**

1. **Front Office (Streamlit UI)**: This is where you interact with the system - upload files, type questions, and see answers.
2. **Document Processing (File Handler)**: This part figures out what kind of file you uploaded and extracts the important stuff.
3. **Question Interpreter (Query Handler)**: This part reads your question and figures out what you're really asking for.
4. **Task Coordinator (Tool Invoker)**: This decides which specialist tool should handle your request.
5. **Specialist Tools**: Different experts for different jobs:
6. **Summarizer**: Reads long documents and gives you the key points
7. **Data Analyst**: Crunches numbers in spreadsheets
8. **Visual Designer**: Creates charts and graphs
9. **Comparison Expert**: Shows differences between data sets
10. **Web Researcher**: Grabs and summarizes information from websites
11. **Translator**: Converts responses to different languages
12. **Record Keeper (History Logger)**: Keeps track of your conversations so you can refer to them.

**Why We Built It This Way**

**Building Block Approach**

We built the system using separate modules (like LEGO blocks) that can be easily changed without breaking everything else. This means we can update one part without messing up the whole system.

**Using Streamlit**

We chose Streamlit because it lets us create a nice-looking interface without needing to be web design experts. It handles file uploads and displaying charts with minimal fuss.

**Plain Language Understanding**

The system uses keywords in your questions to figure out what you want. While not as fancy as some AI approaches, it works reliably for most financial questions and is easy to update.

**Support for Different File Types**

Financial info comes in all shapes and sizes - spreadsheets, PDFs, Word docs. Our system handles them all, so you don't need to convert files before uploading.

**Keeping Conversation History**

The system remembers your previous questions and answers, making it easier to follow up or review past information.

**Speaking Your Language**

The system can detect different languages, making it useful for international teams or analyzing documents from different countries.

**Core Components**

**File Handler**

* **Purpose**: Processes uploaded financial documents
* **Supports**: CSV, Excel, PDF, and DOCX files
* **Output**: Structured data (DataFrames) or text content

**Query Handler**

* **Purpose**: Interprets user questions to determine intent
* **Detection Method**: Keyword matching
* **Intent Types**:
  + Summarize: "summary", "summarize"
  + Visualize: "trend", "chart", "visualize"
  + Compare: "compare", "difference"
  + Analyze: "average", "total", "stat"
  + Web: "link", "url"
  + Translate: "translate", "language"

**Tool Invoker**

* **Purpose**: Routes requests to appropriate specialized tools
* **Process**: Receives intent + data + query → returns formatted response

**Specialized Tools**

**Summarizer**

* **Function**: Condenses document text
* **Process**: Text in → Summary out
* **Implementation**: Basic extraction of first 100 words

**CSV Analyzer**

* **Function**: Statistical analysis of tabular data
* **Process**: DataFrame + query → Statistical insights
* **Output**: Summary statistics in dictionary format

**Visualizer**

* **Function**: Creates data visualizations
* **Process**: DataFrame + query → Charts
* **Output**: Matplotlib charts displayed in Streamlit

**Comparator**

* **Function**: Highlights differences between datasets
* **Process**: Data + comparison criteria → Comparison results

**Web Reader**

* **Function**: Extracts content from websites
* **Process**: URL → Summarized web content
* **Method**: Extracts paragraphs via BeautifulSoup

**Translator**

* **Function**: Converts text between languages
* **Process**: Text + target language → Translated text
* **Implementation**: Uses Google Translate API

**Utility Functions**

**History Logger**

* **Function**: Records conversation history
* **Storage**: JSON file with timestamped interactions
* **Location**: "history/chat\_logs.json"

**Language Detector**

* **Function**: Identifies text language
* **Process**: Text → Language code
* **Implementation**: Uses langdetect library

**Format Based Questions**

**CSV/Excel Files (Tabular Data)**

**Analysis Questions:**

* "What's the average revenue in this dataset?"
* "Calculate the total expenses across all periods"
* "Show me the statistical summary of this financial data"
* "What's the highest profit margin in this spreadsheet?"

**Visualization Questions:**

* "Create a chart showing sales trends over time"
* "Visualize the relationship between revenue and expenses"
* "Plot the quarterly profit changes"
* "Show me a graph comparing department budgets"

**Comparison Questions:**

* "Compare the Q1 and Q2 earnings in this dataset"
* "What's the difference between 2023 and 2024 revenue?"
* "How do the marketing expenses compare to sales expenses?"

**PDF/DOCX Files (Text Documents)**

**Summarization Questions:**

* "Summarize this financial report"
* "Give me the key points from this document"
* "What are the main insights from this annual report?"
* "Provide a summary of this market analysis"

**Information Extraction:**

* "Extract the financial projections from this report"
* "What does this document say about future investments?"
* "Find the revenue figures mentioned in this document"

**Any Document Type**

**Translation Requests:**

* "Translate this financial data to Spanish"
* "Convert this summary to French"

**Web-Related Questions:**

* "Summarize the financial news from this URL: [link]"
* "What does this financial website say about market trends?"