Approach Document: Receipt Data Extraction and Chatbot

1. Objective

The goal of this project is to extract structured information from a receipt image and enable a chatbot to answer queries related to the extracted data. The chatbot should be able to:

- Identify key receipt details (e.g., restaurant name, address, items purchased, total amount, tax, payment method).
- Allow users to ask questions about the extracted receipt data.
- Provide accurate responses based strictly on the receipt's content.

2. Technology Stack

- **Language:** Python
- **Framework:** Streamlit (for UI)
- **AI Model:** GPT-4-Turbo (for both data extraction and chatbot responses)
- **Libraries Used:**
  - `openai` (for API access to GPT-4-Turbo)
  - `PIL` (for image processing)
  - `base64` (for encoding images)
  - `io` (for handling image streams)
  - `re` (for text cleanup and regex processing)

3. Approach

Step 1: Image Upload & Preprocessing

- Users upload a receipt image in **PNG, JPG, or JPEG** format.
- The image is displayed in the Streamlit app for user confirmation.
- The image is converted to **base64 encoding** for API compatibility.

Step 2: Receipt Data Extraction using GPT-4-Turbo

- The encoded image is sent to OpenAI's **Vision API** with a prompt instructing it to extract:

- **Restaurant Name** (identified as the largest text at the top)
- **Address**
- **Date & Time**
- **Itemized List of Purchases**
- **Total Amount & Tax**
- **Payment Method** (if available)

- The extracted data is **cleaned using regex** to remove unnecessary characters.
- The structured information is stored for chatbot interactions.

Step 3: Chatbot Interaction

- Users can input questions related to the receipt (e.g., "What was the total amount?").
- The chatbot:
  1. Retrieves the extracted receipt details.
  2. Uses **GPT-4-Turbo** to generate responses **strictly based on the extracted data**.
  3. Stores chat history for context-aware interactions.
- If the query relates to **missing or ambiguous data**, the chatbot returns an appropriate response instead of guessing.

Step 4: Handling Edge Cases

- **Incomplete or unclear receipts**: The chatbot informs users when data is missing.
- **Irrelevant queries**: If a user asks something unrelated to the receipt, the chatbot redirects them.
- **Multiple interactions**: The chatbot maintains context using session history to ensure continuity.

4. Justification for GPT-4-Turbo

- **High Accuracy in Text Recognition**: OpenAI's Vision API processes images with a strong understanding of **layout, fonts, and structures**.
- **Context-Aware Responses**: The chatbot can provide logical answers instead of just extracting data.
- **Flexibility**: Handles **structured and unstructured receipts**, improving adaptability to different formats.

5. Conclusion

This approach ensures an **automated, interactive, and efficient** system for receipt data extraction and question-answering. The combination of **image processing, AI-driven extraction, and chatbot functionality** enhances usability, making it a valuable tool for handling receipt-based queries.