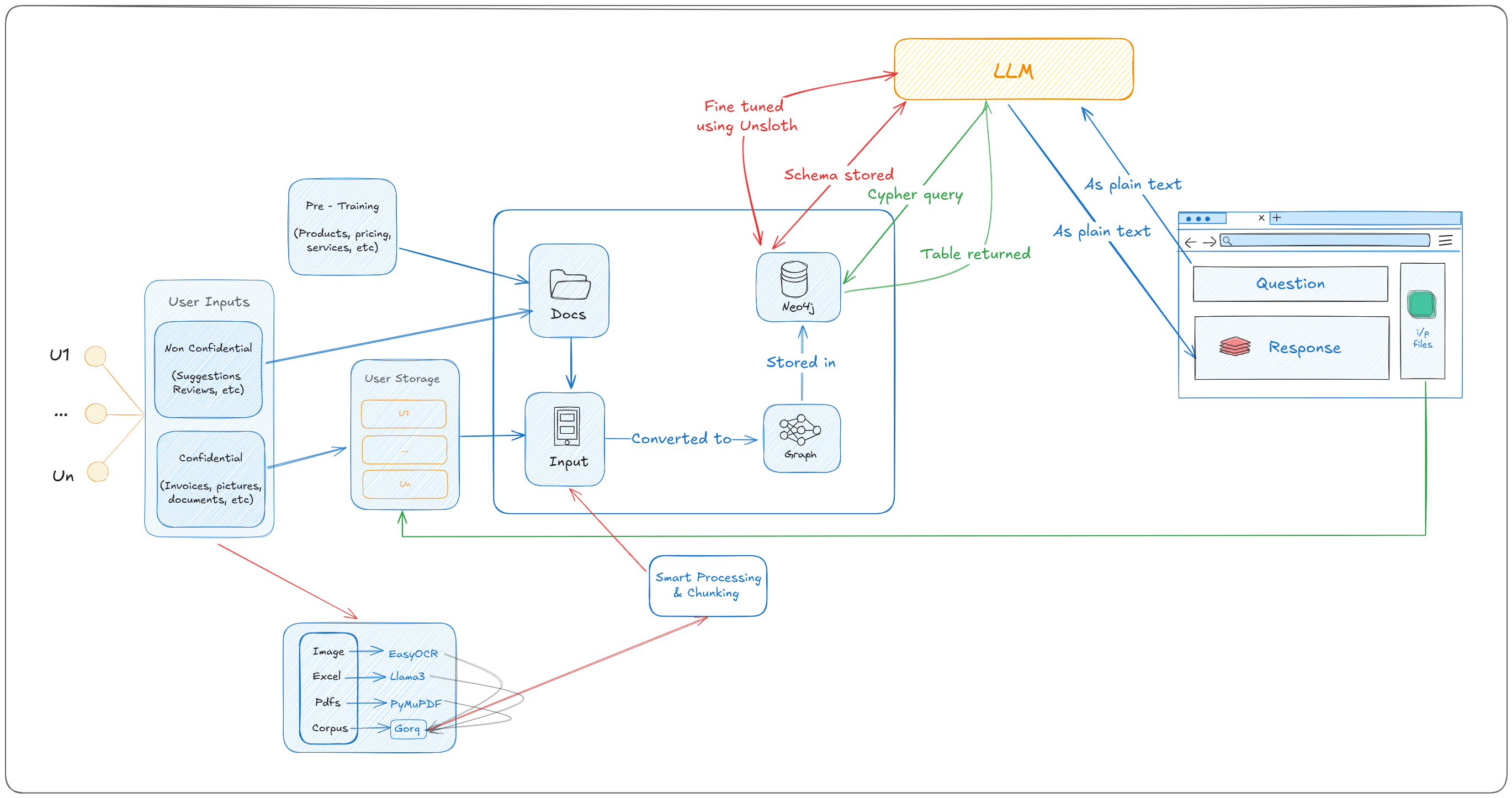
**Project Title**

**"Festty: An Intelligent Virtual Assistant for FESTO’s Website"**

**Objective**

Develop a virtual assistant, **Festty**, integrated into FESTO’s website to provide **advanced chat capabilities** for field technicians, engineers, and customers. Festty will process natural language queries and uploaded documents, offering contextual responses, troubleshooting insights, and document assistance.



**Core Features and Workflow**

**1. User Input Handling**

* **Chat Queries**: Users can ask equipment-related questions directly via an intuitive chat interface.
* **Document Upload**: Supports PDFs and images for processing and contextual question answering.

**2. Document Processing Pipeline**

Handles various file types with specialized modules:

* **PDFs**: Processed with **PyMuPDF** for extracting text and structure.
* **Images**: Text extracted using **EasyOCR** for further processing.
* Extracted data is used for generating contextual responses or augmenting the chatbot's knowledge.

**3. Knowledge Management with Neo4j**

* **Database**: Neo4j is used to store relevant information about equipment, manuals, and past maintenance records in a graph format.
* **Graph Schema**:
  + **Nodes**: Equipment, Manuals, Faults, Resolutions.
  + **Relationships**: "Described In", "Leads To", "Resolved By".

**4. Language Model Integration**

* **LLM**: Powered by **Gemini API** and integrated with **LangChain** for retrieval-augmented generation (RAG).
* **Advanced Chat Options**:
  + Handles plain-text queries.
  + Retrieves data from the Neo4j knowledge graph for contextual responses.
  + Processes uploaded documents for direct answers related to user queries.

**5. Streamlit-Based User Interface**

* **Chat Interface**: Real-time interaction with Festty.
* **Document Upload**: Drag-and-drop functionality for PDFs and images.
* **Response Display**: Clear and concise answers with the option to view document highlights.

**Example Use Cases**

1. **Troubleshooting Assistance**
   * **Query**: "What should I check if my pneumatic valve is leaking?"
   * **Response**: Provides troubleshooting steps, referring to relevant sections of the uploaded manual or stored knowledge.
2. **Document Support**
   * **User Action**: Uploads an invoice or a maintenance manual.
   * **Response**: Summarizes key sections or answers context-specific queries like, "What does section 3.2 of this manual say?"
3. **General Queries**
   * **Query**: "What maintenance does Actuator X need after 500 cycles?"
   * **Response**: Fetches and displays relevant maintenance guidelines from the knowledge base.

**Technologies and Tools**

**Data Processing**

* **PyMuPDF**: PDF text extraction.
* **EasyOCR**: Image text extraction for diagrams or pictures.

**Knowledge Graph**

* **Neo4j**: Stores structured data and supports Cypher-based retrieval.

**Language Model**

* **Gemini API**: Fine-tuned to handle FESTO-specific queries and integrate with the Neo4j graph.

**User Interface**

* **Streamlit**: For developing an interactive, web-based interface.

**End-User Benefits**

* **Real-Time Assistance**: Enables technicians and engineers to resolve issues faster.
* **Enhanced Usability**: Provides document-based contextual responses.
* **Time Savings**: Streamlines query handling and information retrieval.

**Implementation Plan**

**Phase 1: Basic Functionality (1 Month)**

* Set up Neo4j and ingest sample data.
* Develop a basic Streamlit-based chat interface.
* Integrate document upload and processing using PyMuPDF and EasyOCR.

**Phase 2: Advanced Chat Features (2 Months)**

* Fine-tune the Gemini API for FESTO-specific use cases.
* Implement LangChain for RAG using Neo4j data.
* Add document-aware response capabilities.

**Phase 3: Testing and Deployment (1 Month)**

* Test with real-world data and user queries.
* Deploy Festty on FESTO’s website as a virtual assistant.