Technical Information

The algorithm and system we have developed serve as a **core component of a Web3-based privacy-enhanced AI chatbot framework**. The overall workflow can be summarized as follows:

1. **User input handling**

Natural language queries provided by users are ingested within the Docker container, using the JobDetails structure to flexibly handle file-based input.

1. **PII (Personally Identifiable Information) masking**

Before any external communication occurs, the input data undergoes Named Entity Recognition (NER) using SpaCy's en_core_web_sm model.

Entities labeled as PERSON, GPE, and EMAIL are detected and replaced with the [MASKED] token to prevent leakage of sensitive information to external APIs.

This is the core mechanism for ensuring user privacy.

1. **Context retrieval and semantic search integration**

Based on the dataset determined by the query (cameroon or enron), the system generates semantic embeddings using SentenceTransformer (all-MiniLM-L6-v2 model).

It then performs efficient top-3 nearest neighbor search via FAISS index to retrieve relevant contextual documents.

These results are incorporated into a carefully engineered prompt for external LLM inference.

1. **Ollama API integration for external LLM inference**

The constructed prompt is sent to the Ollama API (/api/generate endpoint), invoking the external mistral model for inference.

The Docker container uses an environment variable (OLLAMA_URL) to dynamically configure the backend API URL, ensuring compatibility in containerized environments.

1. **Result saving and Docker I/O compliance**

Final results, comprising original question, sanitized question, dataset, context, and response, are saved in JSON format to /data/outputs/result.json.

This path aligns with Docker volume mounts, allowing seamless automation of output verification in test and deployment scenarios.

This system is designed to **integrate with the Clio-X Web3 privacy-preserving framework**, where the Ollama LLM server operates securely outside Clio-X itself, communicating with the algorithm container over REST APIs.

In summary,

our algorithm provides a fully Dockerized, privacy-aware AI chatbot component that incorporates PII masking, semantic search, external LLM inference, and Web3 readiness, meeting both performance and security objectives.