Document Query Application

1. System Architecture and Design
    a. This application is designed to allow users to upload documents, query them, and interact with the data securely. It comprises of the following key components:
        i. Frontend (Streamlit):
            1. User interface for uploading documents, querying them, viewing history, and downloading chat history.
            2. Renders the UI elements and interacts with the backend to perform document querying and manages user history.
        ii. Backend (Python):
            1. Handles document processing, querying, and interaction with the database.
            2. Uses encryption for secure storage and retrieval of sensitive data.
        iii. Database (SQLite):
            1. Stores user data, uploaded documents, and interaction history.
            2. Provides secure data management and retrieval.

2. Database Schema
    Tables:
        Users:
            a. id (Primary Key): Unique identifier for each user.
            b. username: Stores the username of the user.
        Documents:
            c. id (Primary Key): Unique identifier for each document.
            d. user_id: Foreign key linking to the Users table.
            e. doc_name: Name of the uploaded document.
            f. doc_content: Encrypted content of the document.
            g. upload_date: Timestamp of when the document was uploaded.
        Queries:
            h. id (Primary Key): Unique identifier for each query.
            i. user_id: Foreign key linking to the Users table.
            j. query_text: The text of the user's query.
            k. response_text: The response generated by the application.
            l. query_date: Timestamp of when the query was made.

3. Instructions for Adding New Documents:
    a. Navigate to the document upload section in the application.

b. Select the document file (.pdf, .docx, or .txt) to be uploaded.
c. Enter your query.

4. Security Measures Implemented
    i. Encryption:
        1. All sensitive data (documents and user interactions) are encrypted using the Fernet encryption method from the cryptography package.
    ii. Database Security:
        1. SQLite database is secured with proper file permissions.
        2. Database connections are handled securely to prevent SQL injection and unauthorized access.
    iii. Environment Variable Management:
        1. Encryption keys and other sensitive information are stored in environment variables to prevent exposure.