Test Plan

# Frontend

* **User Authentication:**
  + Test user registration and login with valid credentials.
  + Test user registration with invalid or duplicate credentials.
  + Ensure users can sign out securely.
* **User Interface (UI):**
  + Verify that the UI is responsive and works on various browsers.
  + Test navigation and links within the app.
  + Check for proper error handling and user-friendly error messages.
* **User Profile:**
  + Test updating user information (e.g., name, email)..
* **Medical Records:**
  + Ensure users can access and view their medical records securely.
  + Test the display of medical data, such as lab results or appointments.
* **Blockchain Integration:**
  + Verify that the frontend successfully communicates with the blockchain to fetch and display user data.

# Blockchain

**Data Transportation:**

* **Encryption Verification**:
  + Test the encryption process by sending a sample data payload and confirming that it is successfully encrypted.
  + Verify that the encryption algorithm used provides data security.
* **Decryption Verification**:
  + Test the decryption process by using an encrypted data payload and confirming that it is successfully decrypted.
  + Ensure that the decryption algorithm correctly reverses the encryption.
* **Data Integrity**:
  + Send an encrypted data payload and confirm that it remains intact and unchanged during decryption.
  + Ensure that the system can detect any tampering with the data during transportation.
* **Key Management**:
  + Test the system's key management functionality, including key generation, storage, and retrieval.
  + Verify that keys are securely managed and protected.

**Guardian - Access Control and Authorization:**

* **Access Control**:
  + Test that only authorized users (e.g., medical professionals, supervisors) can access certain parts of the system.
  + Verify that unauthorized users are denied access to sensitive data.
  + Test role-based access control by creating users with different roles (e.g., regular users, supervisors).
  + Confirm that each role has appropriate access permissions.

**File Upload and Download:**

* **File Upload**:
  + Test the file upload functionality by attempting to upload various types of files (e.g., medical records, documents).
  + Ensure that the files are correctly stored and associated with the user's account.
* **File Download**:
  + Test file download functionality by requesting files associated with a user's account.
* **File Versioning**:
  + Upload a file with the same name as an existing file and confirm that it does not overwrite the previous file.
  + Verify that the system maintains version history if needed.
* **Encryption of Files**:
  + Verify that files are encrypted before storage and decrypted upon download.

**Integration Testing:**

* **Error Handling**:
  + Test the system's response to various error scenarios, such as failed uploads, decryption errors, or unauthorized access attempts.
  + Verify that appropriate error messages are displayed to users.

# AI

**Difference Between Different Parameters:**

* **Model Parameters**:
  + Test the chatbot with different parameters, including different token size of model (7B, 13B, 70B), to evaluate their performance, response quality, relevance, and size.
* **Prompt Variations**:
  + Test the chatbot with various types of user prompts, including short questions, long paragraphs, and ambiguous queries.
  + Evaluate how well the chatbot handles a wide range of input variations.

**Find a Reasonable Scale of Pre-Train Data:**

* **Data Size vs. Performance**:
  + Test the chatbot with varying amounts of pre-trained data, ranging from small to extensive datasets.
* **Fine-Tuning**:
  + Experiment with fine-tuning the chatbot on specific healthcare-related data or knowledge to assess its ability to provide relevant information.

**Chatbot Functionality:**

* **Prompt Understanding**:
  + Test the chatbot's ability to understand and accurately interpret user queries related to healthcare terminology, life advice, and other non-professional topics.
  + Evaluate how well it distinguishes between different types of queries.
* **Context Management**:
  + Evaluate how well the chatbot maintains and utilizes context from previous interactions with the user.
  + Test whether it can recall prior conversations and respond accordingly.

**Chatbot Output:**

* **Response Accuracy**:
  + Test the chatbot with a range of healthcare-related questions and queries.
  + Evaluate the accuracy of responses, including terminology definitions and life advice.
* **Desired Output Example**:
  + Q: The report said I have type II diabetes, what food do I need to avoid?  
    A: According to your report… (reply according to truth, does not make up terms, put allergies as a factor, does not conflict with doctor’s order, allow to ask questions for more detail).
  + Q: The report said I have type II diabetes, any advice?  
    A: Yes, here are some GENERAL advices…
  + etc.

**Additional Testing Considerations:**

* **Quality Assurance**:
  + Continuously monitor and evaluate the chatbot's responses in a production environment.
  + Collect user feedback to improve and fine-tune the chatbot over time.
* **Security and Privacy**:
  + Test the chatbot's ability to handle sensitive healthcare information securely.
  + Verify that it does not disclose personal or confidential data to unauthorized users.
* **Error Handling**:
  + Test the chatbot's response to erroneous or incomplete queries.
  + Ensure that it provides informative and user-friendly error messages when necessary.