PRD: MVP Medical Diagnosis Tool (Hackathon Edition) Shifaa Healthcare

TL;DR for Dummies

This is a simple mobile app that helps doctors in Gaza collect and save patient data—especially when internet is unreliable. Doctors can type or speak symptoms. The app turns voice into text, summarizes the input using local AI, and saves a secure fingerprint of that data to a local blockchain. No internet needed. Everything runs locally using Docker and works on phones.

Why this tech?

- **React Native + Expo Go**: Fast to build and test on iPhones and Android
- **Whisper + LLM (via Docker)**: Smart voice input and summarization, no cloud
- **Ganache Blockchain**: Tamper-proof record keeping, runs offline

Built for speed, privacy, and resilience—because Gaza's doctors don't have time or connectivity to waste.

Overview

This app is designed as a practical support tool for doctors working in Gaza, where infrastructure challenges such as power outages, intermittent internet, and lack of data security are daily realities. The goal is to provide an offline-first, privacy-aware tool for symptom collection, voice transcription, and basic diagnostic documentation.

By leveraging:

- **React Native** for rapid mobile app development across Android and iOS
- **Dockerized AI services** for voice transcription (Whisper) and symptom summarization (LLM) running on local machines or a LAN
- **Blockchain (Ganache)** to offer tamper-proof medical data storage without needing a centralized or online system

This specific tech combination was chosen to meet these constraints:

- **Offline capability** ensures continued use during internet blackouts
- **Self-hosted AI models** eliminate dependency on cloud APIs
- **Simple blockchain integration** provides transparent and immutable record-keeping

The result is a tool that empowers frontline medical workers to document, access, and trust the data they're generating—even in low-resource or disconnected environments.

Docker will be used to host the self-hosted AI components (Whisper and LLM). This approach is chosen for simplicity, speed, and portability—ideal for a proof-of-concept hackathon build. The containers will expose HTTP endpoints on localhost or a shared LAN, enabling smooth integration with the React Native frontend. Docker will be used to host the self-hosted AI components (Whisper and LLM). This approach is chosen for simplicity, speed, and portability—ideal for a proof-of-concept hackathon build. The containers will expose HTTP endpoints on localhost or a shared LAN, enabling smooth integration with the React Native frontend. A lightweight, offline-first medical diagnosis proof of concept app built using React Native. It supports form-based and voice-based input to collect patient symptoms and diagnostic details. Voice inputs are transcribed via a self-hosted voice-to-text AI model. An optional lightweight LLM provides symptom summarization or tag suggestions. Records are then saved to a barebones blockchain setup for tamper-proof auditability.

Goals

- Allow doctors to input patient data via form or voice.
- Transcribe voice using local/self-hosted AI.
- Use LLM to suggest symptom summaries or tags.
- Store records to blockchain (most minimal viable form).
- Prioritize offline-first functionality.
- Simple and usable UI for demo.

Key Features (MVP)

1. **User Interface**
    - Patient intake form (text fields: name, age, symptoms, notes)
    - Voice recording button (start/stop recording)
    - Display transcribed voice input in form field
    - LLM button to summarize or tag symptoms
    - Submit button to trigger save to blockchain
2. **Voice Input & Transcription**
    - Local voice recording (RN Audio or similar)
    - Self-hosted voice-to-text model (e.g., Whisper on local server or LAN)
3. **LLM Summarization (Optional)**
    - Self-hosted lightweight LLM (e.g., Mistral-7B-Instruct)
    - Exposed as a REST API endpoint
    - Takes transcribed symptoms and returns summarized notes or tags
4. **Blockchain Storage**
    - Barebones setup using Ganache local testnet
    - Store only record hash and timestamp in a smart contract
    - No IPFS, no syncing logic
5. **Offline-First**
    - Local storage of data (AsyncStorage/SQLite)
    - No sync logic needed

Must-Have Setup (Don’t worry about this too much)

Expo Go will be used to test the React Native app on iPhone and Android. This allows fast iteration without the need for full native builds during development.

Each team member should have the following installed and configured before Day 1:

- **Git** (latest stable version)
- **Node.js** (v18 or later)
- **npm** or **yarn**
- **Docker** + Docker Compose -> only specific dev and demo person need this
- **Python 3.10+** (for Whisper server) -> only specific dev and demo person need this
- **Ganache CLI or GUI** (for local Ethereum testnet) -> only dev and demo person need this
- **Code Editor** (e.g., VS Code with ESLint and Prettier extensions)

Ensure that Docker containers for Whisper and LLM are built and tested on the local machine or shared LAN by Day 1.

Tech Stack

- **Frontend**: React Native with shadcn/ui for styled components
- **Frontend**: React Native with shadcn/ui (Expo workflow with Expo Go for testing)
- **Voice Transcription**: Whisper (self-hosted via Python server) on docker
- **LLM**: Mistral-7B-Instruct or GPT4All (self-hosted) on docker
- **Blockchain**: Ganache local Ethereum testnet + simple smart contract
- **Storage**: offline local storage (avoid backend if possible we dont have time)

Team Assignments

Hamza (DevOps / Self-hosted AI)

- Focus: Infrastructure setup, Dockerized services (Whisper, LLM, Ganache), backend API endpoints

Tayyeb (Frontend Developer)

- Focus: Core app functionality, integrating voice, LLM, and blockchain features into the UI

Akram (Designer / Vibe Coder)

- Focus: UI/UX design, styling, and front-end development (paired with Liban)

Liban (Beginner Dev)

- Focus: UI development and testing support (paired with Akram)

Temis (Full Stack)

- Focus: Blockchain logic, local storage, frontend-backend integration, and final app polish

---

Feature Roadmap

- ⬜ Set up GitHub repository
- ⬜ Initialize React Native app with Expo workflow
- ⬜ Configure shadcn/ui components
- ⬜ Build patient intake form UI (text inputs)
- ⬜ Add voice recording button and functionality
- ⬜ Set up Whisper server in Docker
- ⬜ Create Whisper transcription API endpoint
- ⬜ Connect voice recording feature to Whisper API
- ⬜ Display transcribed text in form field
- ⬜ Set up LLM server in Docker (Mistral/GPT4All)
- ⬜ Create LLM summarization API endpoint
- ⬜ Send transcribed text to LLM API for summarization
- ⬜ Display LLM output in form field
- ⬜ Store completed form data locally using AsyncStorage
- ⬜ Set up Ganache blockchain environment using Docker
- ⬜ Write minimal smart contract to save record hash + timestamp
- ⬜ Deploy contract to Ganache and connect frontend

- ⬜ Save hashed form record to blockchain via contract
- ⬜ Build basic confirmation and feedback UI
- ⬜ Conduct full end-to-end test of app workflow
- ⬜ Prepare and rehearse demo presentation

Out of Scope for MVP

- Authentication and user roles & Anything Backend
- Full fledged diagnosis form (be prepared to hard code values where needed)
- Full diagnosis engine or symptom checker
- Real blockchain smart contract

Deliverables

- Working mobile app (React Native)
- Self-hosted Whisper model
- Self-hosted lightweight LLM
- Barebones blockchain record storage
- Demo script and pitch presentation

Success Criteria

- App runs offline and accepts form + voice input

- Voice transcription and LLM outputs populate the form
- Data is saved and retrievable with blockchain proof