🔵 Backend (Django): Detailed Work Plan

I. APIs to Build In-House

API Endpoint	Purpose	Developer Work
POST /api/symptom-checker	Accept symptoms input from user, call AI API to return diagnosis	Build endpoint to take user data ➔ Call OpenAI/Gemini ➔ Process result ➔ Return structured JSON
GET /api/first-aid-guidance/{case}	Provide voice-guided first aid steps	Build endpoint that fetches pre-written first aid instructions by category (stored in DB or static files)
GET /api/daily-coaching	Serve personalized daily health advice	Build API that triggers GPT or Gemini API to create custom short health tips based on user profile
POST /api/chat-health-assistant	Health Q&A chat feature	Handle user question ➔ Call GPT ➔ Return chat response
GET /api/local-health-alerts	Serve localized outbreak/environment alerts	Fetch from public health APIs based on geolocation, or hard-code basic examples if API not available
GET /api/nearby-hospitals	Locate nearby health centers	Geolocation API integration (like OpenStreetMap Nominatim or Google Places) based on user coordinates
GET /api/healthz	Health check endpoint for uptime monitoring	Return simple `{ "status": "healthy" }` if server is OK; for Uptime Kuma

---

II. APIs to Integrate from Third Parties

Third-Party API	Purpose	Developer Work

OpenAI API / Gemini AI API	Medical Q&A, daily coaching, symptom checker suggestions	Integrate using `openai` or Gemini SDKs ➔ Monitor responses, handle slow/failures
OpenFDA API (optional)	Pull verified drug/health data	Build optional lookup feature (drug names, symptoms info) if needed
Geolocation APIs (Google Geolocation API / Nominatim)	Get user location for nearby hospitals	Integrate API to turn latitude/longitude into nearby hospital names
Healthchecks.io	Backend uptime & cron job monitoring	Add backend ping URLs to Healthchecks.io dashboard

---

III. Required Monitoring/Recovery Work (Backend)

Task	Developer Work
Set up **Uptime Kuma**	Monitor `/api/healthz` every 30 sec
Add **pg_isready cron job**	Ping DB every 2 min, alert if down
Add **Custom AI middleware**	Monitor latency and failures of GPT/Gemini calls
Set up **PM2 process manager**	Backend auto-restarts on crash
Configure **Discord alerts**	Uptime Kuma + Healthchecks.io post alerts

---

🟠 Frontend (React): Detailed Work Plan

I. UI Components / Features to Build

Frontend Feature	Purpose	Developer Work
Symptom Checker Form	Let user input symptoms (checkboxes / free-text)	Create form ➔ POST to `/api/symptom-checker` ➔ Render AI diagnosis results nicely

Voice-Guided First Aid UI	Emergency section to select case and hear voice instructions	Build simple selection menu ➔ Call `/api/first-aid-guidance/{case}` ➔ Use browser SpeechSynthesis API to read aloud
Daily Health Coaching Card	Display daily tips in homepage/dashboard	GET `/api/daily-coaching` ➔ Render card with today's advice
Health AI Chatbot UI	Simple chat window for health Q&A	POST user input ➔ `/api/chat-health-assistant` ➔ Stream response back
Local Alerts Banner	Top notification banner for alerts	Call `/api/local-health-alerts` ➔ Render any active health warnings
Nearby Hospitals Map/Link	Map showing hospitals nearby	Integrate simple Google Maps link from `/api/nearby-hospitals` response
Error Fallback Modals	Handle API failures gracefully	If backend/AI fails, show user-friendly modals (ex: "Service unavailable, please try again.")

II. Monitoring/Observability Work (Frontend)

Task	Developer Work
Integrate **Sentry SDK**	`npm install @sentry/react @sentry/tracing` ➔ Catch all runtime errors
Integrate **Web Vitals tracking**	Use Web Vitals API ➔ send scores (LCP, FID, CLS) to backend or console
Configure **Healthchecks.io alerts**	(Optional) Use Healthchecks.io for front-end build pinging
Add **Frontend recovery popups**	On critical React crashes, suggest user to refresh the page (fallback modal)

✍️ Delegation Made Easy

Team	Members	Suggested Assignments
Backend Team (Django)	4 people	1 for AI API integrations, 1 for database & healthz endpoints, 1 for symptom checker/alerts APIs, 1 for monitoring setup (Kuma, Healthchecks, PM2)
Frontend Team (React)	6 people	2 for Symptom Checker & Chatbot UI, 2 for First Aid + Coaching cards, 1 for Alerts banner + Hospitals, 1 for Sentry, Web Vitals, and Recovery modals

---

🛠️ Example Third-Party Tools Quick Install List

Tool	Quick Install Command
OpenAI API Python SDK	`pip install openai`
Google Geolocation / Places API	N/A (use HTTP request)
Sentry (Frontend)	`npm install @sentry/react @sentry/tracing`
PM2	`npm install pm2 -g`
Uptime Kuma	Install via Docker / Node.js
Healthchecks.io	Signup and create checks (no install)

✅ Final Reminder

✅ Make backend APIs lightweight and simple (don't overcomplicate).
✅ Make frontend fast, with graceful fallbacks if backend or AI APIs are slow.
✅ Focus especially on **Symptom Checker**, **Chatbot**, and **First Aid** for MVP.
✅ Monitor everything using Discord alerts for instant action.