



Clinical Intelligence Platform – Frontend Implementation PRD

The Clinical Intelligence Platform is a **continuous patient monitoring** system that integrates multi-modal data (vitals, labs, notes, etc.) into a unified timeline for each patient. It computes a **risk score** from longitudinal trends and generates explainable alerts when risk rises ¹ ². The UI/UX must convey trends “at a glance” with time-series charts and risk indicators, and provide contextual explanations for alerts ¹ ³. Alerts and risk summaries are delivered via a web dashboard (for doctors/nurses) and a companion mobile app, enabling timely clinical intervention ⁴ ⁵.

Key User Roles

- **Doctor / Clinician:** Views patient risk dashboards, receives alerts, reviews explanations, and provides feedback.
- **Nurse:** Similar to doctor role; often at bedside with mobile alerts.
- **Administrator:** Manages application settings, user accounts, and data ingestion.

(All roles use hardcoded/demo credentials. For example, doctor1@hospital / password, nurse1@hospital / password, and admin@hospital / adminpass.)

Technology Stack

- **Framework:** React (with functional components and hooks)
- **Styling:** Tailwind CSS, shadcn/ui component library (Radix-based)
- **Charts:** A simple charting library (e.g. Recharts or Chart.js) for time-series plots
- **Routing:** React Router (e.g. /login, /dashboard, /patients/:id, /alerts, /upload, /settings)
- **State:** Mocked data (no real backend); use React state or context for demo data
- **Auth:** Hardcoded user roles; no real authentication (simulate via front-end logic)
- **Testing:** Jest with React Testing Library for components; optional Cypress for critical flows

Screen Map & Navigation

The application includes the following screens/routes:

- **Login Screen** (/login): Authentication entry point.
- **Dashboard (Patient List)** (/dashboard): Main view showing assigned patients with current risk summaries.
- **Patient Detail** (/patients/:id): Detailed longitudinal dashboard for one patient (charts, risk score, key vitals/labs, alerts, explanations).
- **Alerts List** (/alerts): All recent alerts (across patients) in a table or list format.

- **Data Ingestion (Upload)** (/upload): Admin screen to upload patient data (CSV/PDF), preview mappings, and track import progress.
- **Settings/Admin** (/settings): Manage users, view logs, configure app (if any).

Additionally, a **mobile app** (web or native) includes: - **Alerts Feed**: List of incoming alerts.

- **Alert Detail**: Graphs and explanation for a selected alert.

- **Risk Summary**: Quick view of patient's current risk status.

Patient and alert data are entirely **mocked**. All dynamic behavior (risk scoring, thresholds, etc.) is simulated with preset JSON.

User Journeys

1. Doctor/Nurse Dashboard:

2. User logs in and lands on **Dashboard**, which lists their patients. Each patient card shows name, ID, current risk level (e.g. High/Medium/Low), and a small trend sparkline (e.g. recent risk trajectory) ¹ ² .

3. The list can be sorted or filtered (e.g. by risk severity). Clicking a patient navigates to that **Patient Detail** page.

4. Patient Detail Exploration:

5. The page displays the patient's **longitudinal data**: time-series charts for key vitals (HR, BP, oxygen, etc.) and labs (e.g. lactate). These charts automatically highlight trends (up/down arrows or colored lines) ⁶ ³ .

6. The **current risk score** is prominently shown (numeric or gauge) with a label (Low/Medium/High) ⁷ ² . Example: "Risk Level: High (↑ from Medium)" ⁸ .

7. If an alert has recently fired, an **Alert Panel** at the top lists the alert summary and a plain-language explanation of why risk is high (e.g. "Blood glucose ↑ 29% over 18 months; BP steadily rising" ³). Key contributing factors (e.g. "Sustained glucose increase", "Rising BP trend") are shown as bullet points ⁸ .

8. A small **timeline visualization** (e.g. scrollable event timeline) may show major events (lab tests, admissions) aligned with the charts ² .

9. Alerts List and Acknowledgement:

10. The **Alerts** screen shows all outstanding alerts (across patients) in a table with columns: Patient, Time, Severity, and a brief explanation snippet. Alerts are color-coded by urgency.

11. Clicking an alert goes to the Patient Detail view, anchored at the alert explanation.

12. Each alert entry has action buttons for feedback: **Acknowledge**, **Dismiss**, or **Mark as Useful/Not Useful** ⁹ . For example, a doctor can tap "Useful" if the alert was clinically relevant, which is recorded (no real learning, but simulates feedback loop ⁹).

13. Data Ingestion (Admin):

14. The **Upload** page allows an admin to upload patient data files (CSV, JSON or PDF). The UI includes: a file selector or drag-drop area, a preview table of parsed data, and field-mapping controls (e.g. map "Glucose" column to internal vital field).
15. After submission, a progress bar shows import progress. On success, a summary (e.g. "X records added") appears. Errors (e.g. missing columns) display an alert message. This simulates step "Longitudinal Data Ingestion" ¹⁰ ¹¹ .
16. No real storage is needed: the imported data can populate frontend state for demonstration (e.g. new patient profiles).
17. **Settings / Admin:**
18. Simple page for demo: list of hardcoded users (Doctor, Nurse, Admin) with roles. An admin can "edit" roles (no backend effect) and reset the mock data if needed. This area also notes that all credentials are dummy (e.g. table of username/password) for testers.
19. **Mobile App (Alerts & Summary):**
20. On mobile, the main screen is an **Alerts Feed**. It lists incoming alerts (similar to Alerts List above). Each item shows patient name, brief message, and severity.
21. Tapping an alert opens **Alert Detail**: displays the same charts/explanation as web, sized for mobile. Users can swipe between alerts.
22. A **Risk Summary** screen (or widget) can show a user's current patients (or just the active one) with risk levels and a sparkline, enabling quick oversight on the go.

Per-Screen Specifications

Login Screen

- **Layout:** Centered login form with fields for Email and Password, and a **Login** button.
- **Components:** `shadcn` `Input` for text, `Button` for submit, simple heading.
- **Actions:** On submit, validate against hardcoded creds (e.g. `doctor1@.../password`). On success, route to `/dashboard`; on failure, show error toast.
- **States:** Idle, Loading (spinner on button), Error (invalid login).
- **Accessibility:** Labels for inputs, proper focus order, error message announced.

Dashboard (Patient List) Screen

- **Layout:** Responsive grid or table. Sidebar nav (Home/Dashboard, Alerts, Data Upload, etc.) and a top bar (app title, user menu). Main area: a card or row per patient. Each card shows patient info (Name, Age, MRN), current risk badge (e.g. red/yellow/green), and a mini-chart/sparkline of the risk trend.
- **Components:** `Card` or `Table` for patient rows, `Badge` for risk level, small `Chart` component (sparklines), `Button` or clickable row for navigation.
- **Actions:** Click patient → navigate to detail. Filter/search box filters list.
- **States:** Loading (skeleton cards), Empty (no patients – show "No data" message), Error (e.g. "Failed to load patients").

- **Accessibility:** Table headers labeled, keyboard navigation for selection, color blind-friendly colors for risk levels (also use icons).

Example citation: The dashboard “continuously updates an interactive dashboard for each patient” with time-series plots of key vitals/labs ¹. The UI should let providers “see trends at a glance” ¹.

Patient Detail Screen

- **Layout:**
 - Header with patient name/ID and summary (age, sex, location).
- **Risk Score Section:** Prominent display of current risk (e.g. a gauge or large number and descriptor “High/Medium/Low”) ².
- **Charts Section:**
 - Top chart: risk score over time (line chart).
 - Subsequent charts: vital signs over time (HR, BP, O₂, etc.) and select labs (e.g. lactate). Each chart includes trend arrows and highlights (e.g. red segments for recent upticks). ⁶
 - Annotated points for key changes (per an alert).
- **Alerts/Explanation Panel:** If an alert is active, show a boxed panel:
 - **Title:** “Alert: Elevated Risk” or similar.
 - **Annotated Trend Chart:** e.g. last 24h zoomed-in vitals chart (mirroring [14†L200-L209]).
 - **Plain-Language Explanation:** e.g. “Respiratory rate ↑ 15% over 6h, MAP ↓ 12%. Risk of respiratory failure elevated.” ¹².
 - **Key Drivers List:** bullet items “High risk due to increasing creatinine...” ¹³.
 - **Actions:** Buttons “Acknowledge” and a toggle “Useful/Not Useful” ⁹.
- **History/Notes:** Optional section listing recent clinical notes or events (scrollable timeline).
- **Components:** Multiple Chart instances, Card or Alert for explanation, Badge for risk label, Button for feedback. Use shadow forms for any text input (e.g. comments).
- **Actions:**
 - Hovering or clicking a point on a chart shows tooltip with exact values/time.
 - Clicking “Acknowledge” collapses the alert panel (simulating alert handling).
 - Marking useful/not triggers a small confirmation (simulating feedback loop ⁹).
- **States:**
 - **Normal:** Shows charts and most recent alert if any.
 - **Loading Data:** Placeholder for charts until mock data loads.
 - **No Alert:** Alert panel hidden (show “No active alerts” message if needed).
 - **Error:** If data failed to load, show error text.
- **Accessibility:** Charts have aria-labels (e.g. “Heart rate over time”). Ensure alert text is readable with semantic markup. Buttons have aria-labels (e.g. “Acknowledge alert”). Use high contrast.

Citations: The UI must support explainability – each alert “comes with an explanation of its rationale” and lists top factors ¹⁴. Example text: “High risk due to increasing creatinine (last 12h) and decreasing urine output.” ¹⁴ ³. The patient timeline view visualizes subtle trends (“charts reflect increasing or decreasing patterns” ¹).

Alerts Screen

- **Layout:** Table or list view of alerts. Columns: Patient Name, Time, Severity (High/Med/Low with color), Description. Sidebar and header as in Dashboard.

- **Components:** shadcn `Table`, `Badge` for severity, `Button` for feedback.
- **Actions:** Click row → opens patient detail at corresponding alert. In-table buttons for quick “Acknowledge” or “Mark as Useful/Not”.
- **States:** Empty (no alerts – show “All clear”), Loading (skeleton rows).
- **Accessibility:** Table headers, focusable rows, ensure clear tab stops on action buttons.

Data Ingestion (Upload) Screen

- **Layout:**
 - File input area (drag-and-drop or select file).
 - After file selected, show a preview: a small table of parsed data columns vs. sample values. Provide dropdowns to map columns to fields (e.g. map “Blood Pressure” to `BP` field).
- **Submit** button to start import.
- Progress indicator (progress bar or spinner) after submission.
- Result area: success message (“Imported 1 patient, 30 records”) or error messages.
- **Components:** shadcn `Card` for upload area, `Input` for file, `Table` for preview, `Button`, `ProgressBar`, `Alert` for errors.
- **Actions:**
 - File selection triggers parsing (mock), shows preview.
 - Submit triggers a fake delay; then randomly choose success or error for demonstration (but always success is fine).
 - On error, highlight missing fields and show message.
- **States:** Idle (no file), File selected (show preview), Uploading (progress bar), Success (show count), Error (show error).
- **Accessibility:** File input should accept keyboard focus. All inputs labeled. Use `role="progressbar"` for progress.

Citations: The platform must ingest multimodal data (vitals, labs, etc.) into a unified patient timeline ¹¹. This screen simulates that process, allowing admins to upload longitudinal patient data.

Settings Screen

- **Layout:** Tabs or sectioned page with two parts: *Users* and *Configuration*.
- **Users:** List hardcoded users (Doctor, Nurse, Admin) with their roles and dummy credentials. Buttons to “reset” or “delete” (non-functional for demo).
- **Config:** Simple toggles (non-functional) for demo settings (e.g. “Enable Alerts”, “Data Sync”).
- **Accessibility:** Standard form/table semantics.

Reusable Components (shadcn/UI)

We will create a library of React components styled with Tailwind via shadcn. Examples include: - **Navbar/Sidebar:** Navigation layout with icons and links (using `Tabs` or `List` components).

- **PatientCard/PatientTableRow:** Displays patient summary (name, age, risk).
- **AlertCard/AlertRow:** Shows alert summary with icon/badge.
- **Chart:** Wrapper around a chart library, accepting data and config props. Should support line and sparkline charts.
- **Badge:** Status indicator (e.g. risk level colors: green/yellow/red).
- **Modal:** For any dialogs (e.g. confirm dismiss).

- **Button/Input/Form:** Standard form controls from shadcn.
- **ProgressBar:** Shows upload or computation progress.

Each component should handle loading, empty, and error states as props. E.g. `<Chart loading={true}/>`. Accessibility (aria-labels, roles) must be added (shadcn components are mostly accessible by default).

Mock AI Behavior

All AI-driven features are faked with static data: - **Risk Score Generation:** Pre-calc a synthetic risk score (0–100) per patient. E.g. use a simple function of time or random. Display as gauge or percentage.

- **Alerts Trigger:** For demo, assume an alert exists if risk > threshold (e.g. >70). The alert time and explanation text are hardcoded or derived from static history.

- **Explanation Generation:** Use predefined templates. For instance, use copy from docs: “Blood glucose increased 29% over 18 months” ³, “Respiratory rate ↑ 15% over 6h” ¹², or “High risk due to increasing creatinine (last 12h)...” ¹⁴. These can be stored in JSON and selected per scenario.

- **Trend Analysis:** Compute simple trend arrows: if a vital's last point > first point, show ↑, else ↓. Color lines if trending. The mock data should allow these patterns.

- **Feedback Loop:** When user clicks “Useful/Not Useful” on an alert, record it in local state and show a toast confirmation. No real ML adapts, but optionally remove the alert to simulate suppression.

These behaviors simulate key system features: continuous monitoring (“watches every patient and warns before a crisis” ¹⁵), trend detection, and explainability ¹⁴ ³. For example, an alert panel might say:

“Alert fired: Lactate ↑ from 1.2 to 2.8 mmol/L in 6h and MAP ↓ from 80 to 65 mmHg” ⁴.

Testing Recommendations & Folder Structure

- **Folder Layout (src/):**

- `/components/` – shared UI components (Chart, Card, Navbar, etc.)
- `/pages/` – page components (Login, Dashboard, PatientDetail, Alerts, Upload, Settings)
- `/utils/` – helper functions (data simulation, auth helpers)
- `/assets/` – static images or icons if any (logo)
- `/tests/` – test files (parallel to src structure)

- **Testing:**

- **Unit Tests:** Jest + React Testing Library for all components (render, props, accessibility checks). Mock chart data and ensure fallback states render.
- **Integration Tests:** Simulate user flows (e.g. login → navigate → view alert). Use React Testing Library or Cypress (for end-to-end on a dev server).
- **Accessibility:** Automated axe checks in tests or storybook for color contrast, labels.
- **Mock Data:** Provide a fixture file (e.g. `mockPatients.json`) for consistent test data.
- **Styling:** Use Tailwind classes and ensure responsiveness. No external CSS frameworks aside from shadcn.

Mobile App (Alerts-Focused)

- **Scope:** Limited to alert viewing and risk summary (no data upload). Likely implemented as a React Native or responsive web app.

- **Screens:**
- **Alerts List:** Similar to web Alerts screen but optimized for small screens (vertical list, swipe gestures).
- **Alert Detail:** Touch-friendly charts (pinch-zoom optional), full explanation text, and action buttons (“Acknowledge”, “👍 Useful”, “👎 Not Useful”).
- **Risk Dashboard:** A simplified list of patients with their current risk (color-coded). If user is a nurse, show assigned patients only. Possibly show a combined feed of recent alerts.
- **Navigation:** Use a bottom tab bar (e.g. Alerts, Patients) or a drawer.
- **Notifications:** Simulate push notifications by showing a device dialog when a new mock alert arrives during demo (e.g. on a timer, show alert toast).
- **Behavior:**
- Alerts on mobile mirror the web alerts. For instance, tapping a mobile notification opens the alert detail in app.
- The risk summary can use the same risk-calculation logic as web. For example, the app home could display: “Patient Raj – High risk ” and sparkline.
- **Accessibility:** Large touch targets, clear iconography, voice-over labels for charts.

Citations: Alerts are delivered via “push notifications, EHR inbox, or unit dashboards” ¹⁶ ⁴ . Our mobile app will simulate push alerts to the care team (nurse/doctor) devices ⁴ , showing the same explainable trend charts and text as on the web.

Assumptions and Open Questions

- **Data Schema:** We assume numeric vitals/labs and timestamped entries. Exact units and normal ranges are not specified – use consistent units in mock data.
- **Patient Assignment:** We assume each doctor/nurse sees a fixed list of patients. No real authentication means we hardcode which user has which patient for demo.
- **Alerts Frequency:** We will predefine a few alert events. In a real system, alerts fire “every hour” when thresholds cross ¹⁷ ; here we can trigger one on page load.
- **Mobile Platform:** Implementation could be React Native or a responsive React web app. Here, we treat it conceptually.
- **User Feedback:** The UI allows feedback (useful/not) but no back-end model updates. We assume feedback is simply logged locally.
- **No Backend/API:** All data (patients, vitals, alerts) are loaded from static JSON or in-memory variables. Charts and logic do not call servers or AI models.
- **Performance:** We assume small datasets (tens of patients). Chart libraries should be chosen for simplicity over massive scale.
- **Open Questions:**
- What specific charting library to use (Recharts is common, but ensure licensing is OK)?
- Will colorblind users be common? (Use patterns/icons in addition to color for risk levels.)
- Should we support dark mode? (Not required in this demo scope, but Tailwind makes it easy.)
- Actual fields for upload: e.g. will vitals be in a specific CSV format? For demo, we define one sample schema.

Sources: All UI requirements are derived from the project documentation and usage scenarios ¹ ⁴ ³ ⁹ . For example, the docs describe a dashboard with time-series vitals and alerts with explanations ¹ ³ . These form the basis of our screen designs.

1 4 6 7 11 12 13 14 15 16 17 Clinical Intelligence Platform for Longitudinal Patient Risk
Monitoring.pdf

file:///file-ACyGhJt7wzoiFQgEMWzpyr

2 3 8 10 clinical_intelligence_platform_problem_dry_run.md

file:///file-KW2yN8YYdhCpAXAG3m1HjV

5 9 clinical_intelligence_platform_solution_design_ai_strategy.md

file:///file-QA5u6zc4h9jofND5L8xLwi