Technical Assessment: Patient Risk Monitoring System

Submission Deadline: February 20 (08:00 P.M.)

Submission Format: GitHub Repository

Repository Access Required For: lisonsabu@gmail.com

---

1. Project Objective

The objective of this assignment is to design and implement a **Patient Risk Monitoring System**. The application must capture patient data, calculate clinical risk using a deterministic rule-based scoring engine, and maintain a complete audit history of all parameter changes.

Core Constraint: Risk levels must be system-calculated based strictly on input parameters. Users are prohibited from manually overriding risk scores or labels. Any modification to patient parameters must trigger an automatic recalculation of the risk profile.

2. Technical Scope

- **Platform:** Candidates may choose to build either a **Web Application** or a **Mobile Application** (Android/iOS/Cross-platform).
- **Technology Stack:** Flexible/Open. Candidates are encouraged to use frameworks they are most proficient in.

---

3. Functional Requirements

3.1. Patient Data Collection

The system must support two methods of data entry:

A. Manual Form Entry

Users must be able to manually input the following data points:

- **Demographics:** Full Name, Date of Birth/Age, Gender, Contact Details, Admission Date.
- **Clinical Parameters:** Heart Rate, Systolic Blood Pressure, Oxygen Saturation (SpO2), Temperature, Respiratory Rate.
- **Medical History:** Chronic Conditions (Diabetes, COPD, Cardiac Disease), ER Visits (last 30 days).
- **Lab Indicators:** WBC, Creatinine, CRP.
- **Notes:** Free text field for clinical observations.

B. Document Parsing (PDF Upload)

- The user allows the upload of a medical document (e.g., lab report or discharge summary).
- The system must extract relevant patient data from the PDF and auto-populate the corresponding fields in the form.
- Users must be permitted to review and manually correct extracted data before creating the record.
- *Note: Complex AI implementation is not required; rule-based text extraction logic is sufficient.*

3.2. Risk Calculation Engine

The system must automatically compute a risk score based on the following logic.

Scoring Rules

Category	**Criteria**	**Points**
Demographics	Age 60–75	+1
	Age >75	+2
Vitals	Heart Rate 100–120 bpm	+1
	Heart Rate >120 bpm	+2
	Systolic BP <90 mmHg	+2
	Oxygen Saturation 90–93%	+1
	Oxygen Saturation <90%	+2
	Temperature 38–39°C	+1

	Temperature >39°C	+2
	Respiratory Rate >24/min	+1
Clinical History	Chronic Condition (Diabetes, COPD, Cardiac)	+1 (per condition)
	ER Visits (Last 30 days): 2–3	+1
	ER Visits (Last 30 days): >3	+2
Lab Indicators	Elevated WBC, High Creatinine, or High CRP	+1 (per indicator)

Risk Classification

- **Score 0–2: LOW** (Green)
- **Score 3–5: MEDIUM** (Yellow/Orange)
- **Score 6+: HIGH** (Red)

Critical Escalation Protocol

Regardless of the calculated total score, the patient must be immediately classified as **HIGH** risk if **any** of the following criteria are met:

- Oxygen Saturation <85%
- Systolic BP <80 mmHg
- Heart Rate >140 bpm

4. User Interface & Workflow

The application should feature, at minimum, the following distinct views or screens:

4.1. Analytics Dashboard (Home)

A high-level overview for clinical supervisors.

- **Key Metrics:** Total Patients, Patients at High Risk, Recent Admissions.
- **Visualizations:** Risk Distribution Chart (Low vs. Medium vs. High), Risk Trends over the last 7 days.

4.2. Patient List

The primary operational screen.

- **List View:** Display key columns: Name, Age, Admission Date, Last Updated.
- **Risk Indicator:** Each row must clearly indicate the current risk level (Color-coded badge/icon).
- **Interactions:** Ability to expand a row to see a "Quick View" of vitals, or click to enter the detailed edit mode.

4.3. Patient Entry & Details (Edit Mode)

A unified interface for data management.

- **Input Form:** Clean, validated fields for entering Demographics and Clinical Parameters.
- **PDF Upload Zone:** Drag-and-drop area for document parsing.
- **Dynamic Risk Display:** As parameters are edited (e.g., HR changes from 80 to 130), the calculated risk score and label on the screen should update in real-time or upon save.

4.4. Audit Log (History)

A dedicated section or modal within the Patient Details view.

- **Timeline:** A chronological list of all updates made to a patient's record.
- **Diff View:** Explicitly show "Previous Value" vs. "New Value" for changed parameters.
- **Risk Trace:** Show how the risk score changed with each update (e.g., "Risk escalated from Low to High due to SpO2 drop").

5. Sample Scoring Scenario

To ensure the scoring logic is understood, refer to the following example.

Patient: Sarah Jenkins (Age 72)

Input Data:

- **Vitals:** HR 102, BP 110/70, SpO2 91%, Temp 36.8°C, Resp 20.
- **History:** Diabetes, COPD. 1 ER visit in last 30 days.
- **Labs:** Elevated WBC.

Calculation:

- Age 72 (60-75): **+1**
- HR 102 (100-120): **+1**
- SpO2 91% (90-93%): **+1**
- History (Diabetes + COPD): **+2**
- Labs (Elevated WBC): **+1**
- **Total Score: 6**

Result:

- **Risk Level: HIGH**

6. Architectural & Data Requirements

- **Persistence:** Must use a persistent database (SQL or NoSQL).
- **Architecture:** The risk calculation logic must be decoupled from the UI layer to ensure testability.
- **Data Integrity:** Proper data versioning must be implemented to support the audit history requirements.

7. Submission Instructions

Repository:
Upload the complete source code to a public or private GitHub repository.

If you are building a **mobile app**, you must also include a **self-signed APK file** inside the repository so that the application can be directly installed and tested without requiring a build process.

Documentation:
Include a comprehensive `README.md` file containing:

- Project setup and installation instructions.
- Database configuration steps.
- Commands to run the application.
- Information regarding any seed/sample data provided.
- A clear list of features that were fully implemented.
- If you are unable to complete all the requirements, explicitly mention in the README which requirements were not completed .

Development Scope:
You are encouraged to build and implement as much of the system as possible. If all requirements cannot be completed, partial implementations are acceptable, provided they are clearly documented in the README.

You may also add extra functionalities or enhancements that you believe improve or suitably extend the system.

Access:
Grant repository access to lisonsabu@gmail.com