Software Requirement Specifications

Patient Vitals Management System

Project Team	Aiman Rizwan – ar08513 Hiba Shahid – hs08036 Fatima Hasan – fh08026
Submission Date	28th April ‘25

Document Sign-Off

Version	Sign-off Authority	Project Role	Sign-off Date
1	Ateeb Ahmed	Supervisor	28th Apr'25

Table of Contents

1. Introduction

1.1. Purpose of Document

The purpose of this document is to outline the requirements for the development of the Patient Vitals Monitoring System. It details the functional and non-functional requirements, system constraints, interface specifications, design considerations, and other critical factors necessary to ensure the software effectively monitors patient vitals in real-time, alerts healthcare professionals to abnormalities, and maintains historical health data for analysis.

1.2. Intended Audience

This document is intended for healthcare sponsors and stakeholders, including hospital administrators and clinical staff, to help them understand the system's monitoring capabilities and how it aligns with patient care objectives. It serves the IT department by providing technical specifications to evaluate system feasibility and plan for future enhancements, such as integration with electronic health record (EHR) systems. Developers will use this document as a blueprint to implement critical features like real-time vital sign processing, configurable alert thresholds, and secure role-based access controls while maintaining focus on the system's core functionality. Additionally, system administrators can reference this document to ensure proper deployment, ongoing security compliance, and efficient maintenance of the monitoring infrastructure.

1.3. Document Convention

Font: Arial

Size: 10

1.5 Space Gapping

2. Overall System Description

2.1. Project Background

The Patient Vitals Monitoring System is designed to replace manual tracking of patient health metrics with an automated, real-time monitoring solution. Traditional methods of recording vitals (e.g., periodic nurse checks, paper-based logs) are prone to delays, human error, and inefficiencies in critical care scenarios. This project was initiated to address these gaps by providing a centralized platform that continuously captures heart rate, blood pressure, temperature, and oxygen saturation (SpO_2), triggers alerts for abnormal readings, and maintains historical data for medical analysis.

2.2. Project Scope

- Real-time vitals monitoring: Continuously track and display patient health metrics.
- Alert system: Notify healthcare staff when vitals exceed safe thresholds.
- Historical data visualization: Generate trends and reports for clinical decision-making.
- Role-based access: Custom dashboards for doctors, nurses, and administrators.

2.3. Not In Scope

- Integration with external EHR (Electronic Health Record) systems: Future phase.
- Mobile app support: Currently limited to desktop/web interfaces.
- Diagnostic recommendations: Current alerts are based on fixed thresholds only. Future implementations may incorporate personalized thresholds based on individual patient history and baseline vitals patterns.

2.4. Project Objectives

The Patient Vitals Monitoring System aims to significantly improve patient outcomes by reducing response times to critical health events through real-time automated alerts, ensuring timely medical interventions. By leveraging sensor-based data capture, the system enhances data reliability by eliminating errors associated with manual vitals recording. Designed for scalability, it supports simultaneous monitoring of multiple patients with minimal latency, making it suitable for both small clinics and large hospital settings. Additionally, the system prioritizes secure data handling, adhering strictly to healthcare privacy standards such as HIPAA and GDPR to protect sensitive patient information.

2.5. Stakeholders

- Owner

The governing body or executive team (Hospital Administration) funding and approving the project. They are responsible for strategic decisions regarding system implementation across healthcare facilities and ensuring the solution aligns with organizational goals for patient care improvement.

- Clinical End-Users

Doctors: Primary users who monitor patient vitals trends, respond to critical alerts, and make treatment decisions based on system data.

Nurses: Frontline staff who interact continuously with the system, documenting vitals and acknowledging alerts at patient bedsides.

Medical Technicians: Personnel responsible for maintaining and calibrating monitoring equipment integrated with the system

- Project Manager

Oversees the entire development lifecycle, creating timelines and milestones while coordinating between clinical staff, IT teams, and developers. Ensures the project meets healthcare operational needs while staying on schedule and within budget.

- Clinical Informatics Specialist

Acts as the bridge between medical and technical teams. Analyzes clinical workflows to translate healthcare requirements into technical specifications, ensuring the system supports real-world medical scenarios and complies with clinical protocols.

- Developer

The developers are the one who are coding the system or building the logic as well as the interface.

- Quality Assurance people

Validate system accuracy and reliability under clinical conditions.

2.6. Operating Environment

- Backend: Python (Flask).
- Frontend: HTML/CSS for dashboard interfaces.
- Database: SQLite.
- OS: Windows.

2.7. System Constraints

The Patient Vitals Monitoring System requires stable network connectivity between medical sensors and servers to ensure uninterrupted real-time processing of critical patient data. Given the sensitive nature of health information, all patient data transmissions and storage must utilize robust encryption protocols to maintain compliance with healthcare privacy regulations (e.g., HIPAA, GDPR). Additionally, the system depends on compatible medical-grade hardware, including validated vital sign sensors such as pulse oximeters, blood pressure monitors, and thermometers, which must be properly provisioned and calibrated to guarantee accurate data collection. These constraints are essential for maintaining system reliability, data security, and clinical effectiveness.

2.8. Assumptions & Dependencies

The Patient Vitals Monitoring System's functionality is contingent upon several critical software and hardware assumptions. The server and client applications must meet minimum resource requirements, including sufficient memory, storage capacity, and processing power to handle real-time data streams from multiple patients simultaneously. A stable TCP/IP network connection is essential for continuous communication between bedside sensors, central servers, and clinician dashboards.

The system assumes the availability of standardized medical device interfaces to integrate with hospital-grade vital sign monitors such as ECG machines, pulse oximeters, and blood pressure cuffs. For data management, the system relies on Python-based frameworks (e.g., SQLite) for database operations and assumes compatibility with healthcare-specific database for time-series data storage.

3. SOFTWARE DEVELOPMENT METHODOLOGY

An Agile-Incremental hybrid methodology was adopted for this project. This approach was chosen because:

Academic Requirements: The university course structure necessitated iterative deliverables (e.g., prototype, core features, final polish) aligned with semester milestones.

Simulated Stakeholder Feedback: Hypothetical clinical scenarios (based on medical literature) guided prioritization of alert thresholds and dashboard design.

Modular Development: Enabled parallel work streams:

Phase 1: Real-time vitals simulation engine

Phase 2: Threshold-based alert system

Phase 3: Data visualization (graphs/reports)

4. SYSTEM ANALYSIS & REQUIREMENT GATHERING

Requirements were derived through academic research and simulated clinical workflows:

- Python/Flask for backend
- SQLite

5. External Interface Requirements

5.1. Hardware Interfaces

A standard desktop or laptop computer is required to run the application. The system needs sufficient RAM and storage space to handle patient vitals data processing and storage. A functional keyboard and mouse are needed for user interaction. For multi-device testing, a local area network (LAN) connection is recommended to ensure smooth communication between the monitoring dashboard and simulated patient data sources.

5.2. Software Interfaces

The system requires:

- Python (for backend processing)
- SQLite (for database operations)
- Web Browser (Chrome/Firefox for accessing the dashboard)

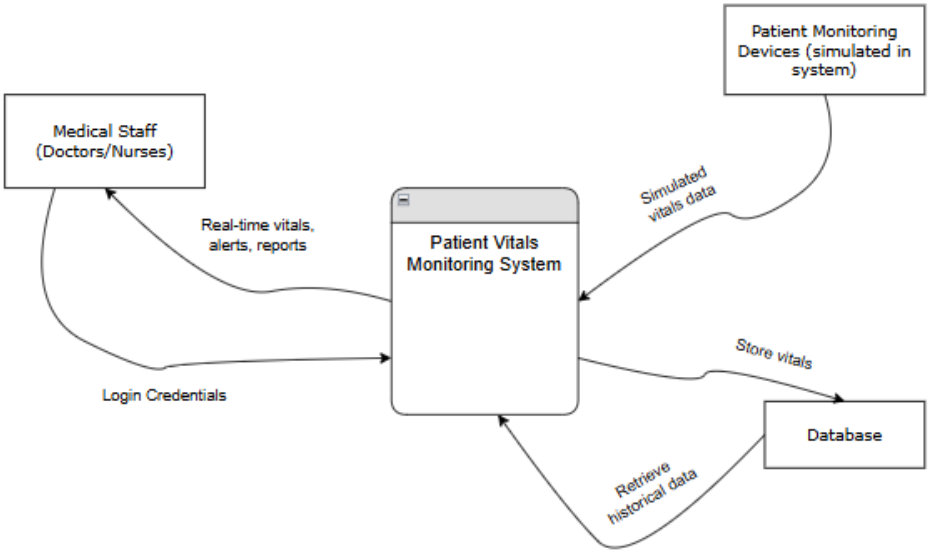
5.3. Communications Interfaces

Not Applicable.

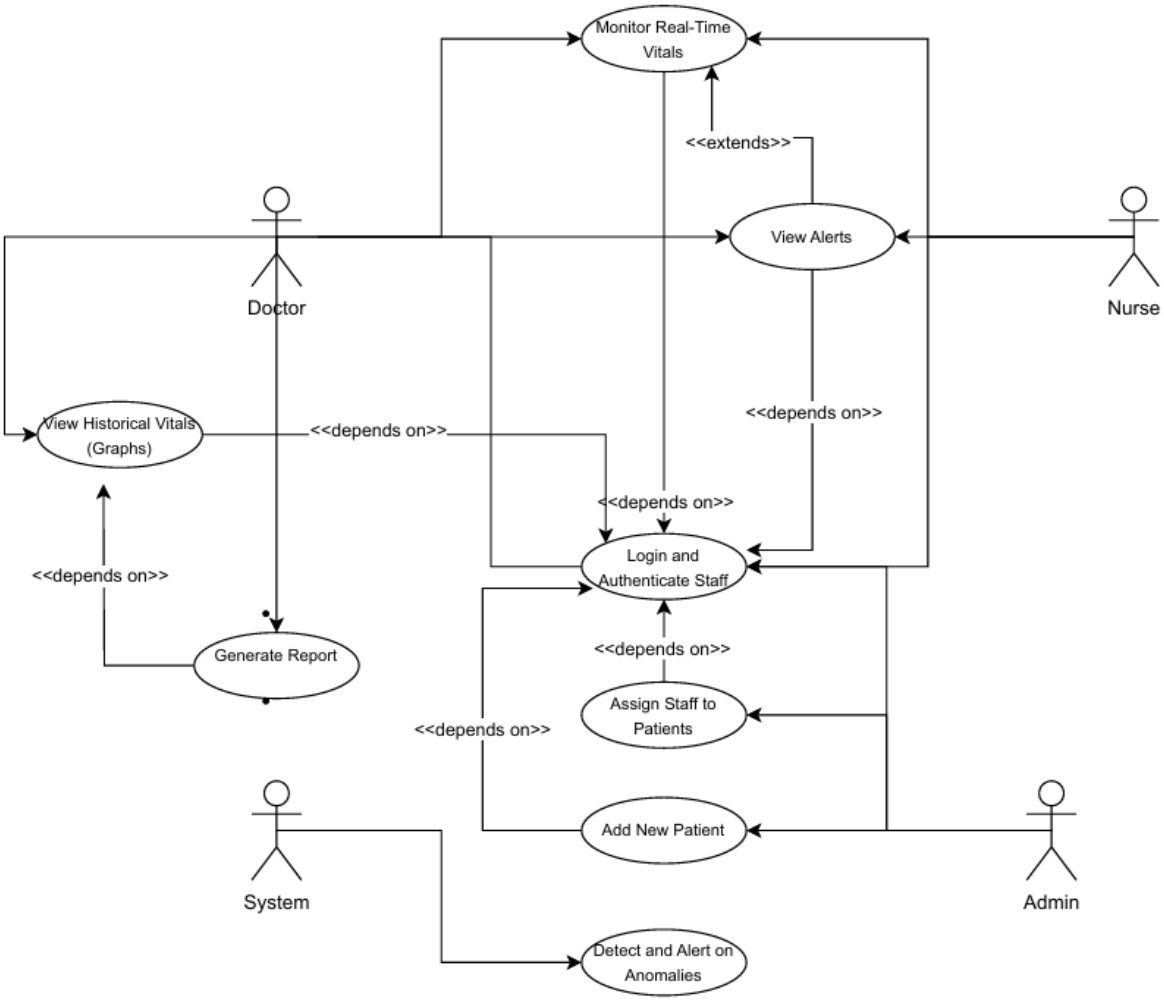
6. Functional Requirements

6.1. Data Flow Diagrams

Context Level DFD:



6.2. Use Cases


Monitor Real-Time Vitals
<<extends>>
Doctor
View Alerts
Nurse
View Historical Vitals (Graphs)
<<depends on>>
<<depends on>>
<<depends on>>
<<depends on>>
Login and Authenticate Staff
<<depends on>>
Generate Report
Assign Staff to Patients
<<depends on>>
Add New Patient
System
Admin
Detect and Alert on Anomalies

This diagram illustrates the basic breakdown of our system in terms of use-cases and actors. The expanded use cases are provided ahead.

Use Case 1: Monitor Real-Time Patient Vitals

Use-Case Name:	Monitor Real-Time Patient Vitals	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-01	
Priority:	High	
Source:	NA	
Primary Business Actor:	Nurse	
Other Participating Actors:	Doctor	
Description:	This use case describes how a nurse logs in and views live updates of vital signs for assigned patients on a dashboard. The system continuously fetches vitals data from the database and updates the screen.	
Precondition:	Vitals are being generated and saved in the database. Nurse has valid login access.	
Trigger:	Nurse logs in to monitor assigned patients.	
Typical course of events	**Actor Action:** Nurse logs in and selects a patient. **System Action:** 1.System fetches the latest vitals from the database. 2.System displays real-time vitals on the dashboard.	
Alternate courses	If the patient is not authorized to the logged-in nurse, access is denied.	
Conclusion	Nurse monitors patient's health in real time.	
Post Condition	Vitals data continues updating. Nurse can intervene if needed.	

Use Case 2: Detect and Alert on Anomalies

Use-Case Name:	Detect and Alert on Anomalies	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-02	
Priority:	High	
Source:	NA	
Primary Business Actor:	System	
Other Participating Actors:	Doctor, Nurse	
Description:	This use case explains how the system automatically checks patient vitals against thresholds and raises alerts if anomalies are detected.	

Precondition:	Vitals exist in the database, and threshold ranges are defined.
Trigger:	The anomaly detection process runs periodically (e.g., every second).
Typical course of events	**Actor Action:** fully automated **System Action:** 1. System fetches the last 10 vitals. 2. Compares each with thresholds. 3. If anomalies found, creates alert.
Alternate courses	If threshold data is missing, skip patient in this cycle.
Conclusion	Alerts for abnormal vitals are generated.
Post Condition	Alerts are saved in the database and visible to staff.

Use Case 3: View Alerts

Use-Case Name:	View Alerts	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-03	
Priority:	High	
Source:	NA	
Primary Business Actor:	Doctor	
Other Participating Actors:	Nurse	
Description:	This use case allows healthcare staff to view alerts associated with patients whose vitals have exceeded safe limits.	
Precondition:	At least one alert exists in the system.	
Trigger:	User opens the alert dashboard.	
Typical course of events	**Actor Action:** Doctor or nurse logs in and opens the alert screen. **System Action:** 1. System fetches alerts from the database. 2. Displays severity and time-stamp for each alert.	
Alternate courses	If no alert exists, display a “No Alerts” message.	
Conclusion	Staff is notified of potential danger.	
Post Condition	Staff can respond to alert or mark as acknowledged.	

Use Case 4: Login and Authenticate Staff

Use-Case Name:	Login and Authenticate Staff	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-04	

Priority:	High	
Source:	NA	
Primary Business Actor:	Staff (Doctor/Nurse)	
Other Participating Actors:	Authentication System	
Description:	This use case outlines how staff members log in with their credentials and gain access based on role (doctor or nurse).	
Precondition:	The user must be registered in the system.	
Trigger:	User enters credentials on the login page.	
Typical course of events	**Actor Action:** User enters username and password.	**System Action:** 1. Verifies credentials. 2. Assigns access based on role. 3. Redirects to appropriate dashboard.
Alternate courses	If credentials fail, the login is rejected and error is shown.	
Conclusion	User is authenticated and directed to their panel.	
Post Condition	User session is active until logout.	

Use Case 5: View Historical Vitals (Graphs)

Use-Case Name:	View Historical Vitals (Graphs)	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-05	
Priority:	Medium	
Source:	NA	
Primary Business Actor:	Doctor	
Other Participating Actors:	System	
Description:	Doctors can view time-series graphs of past patient vitals to assess trends and make decisions.	
Precondition:	Historical vitals must exist for selected patient.	
Trigger:	Doctor opens a patient's graph view.	
Typical course of events	**Actor Action:** Doctor clicks "View Graphs" on patient page.	**System Response:** 1. Queries database for vitals over past N hours/days. 2. Generates graph (via graph_utils.py) and displays.
Alternate courses	If no data exists, inform user with a message.	
Conclusion	Doctor views patient trends.	

Post Condition	Used for diagnosis or planning further treatment.

Use Case 8: Generate Report

Use-Case Name:	Generate Report	**Use-Case Type:** Business Requirements ☑
Use-Case ID:	PVMS-08	
Priority:	Medium	
Source:	NA	
Primary Business Actor:	Doctor	
Other Participating Actors:	None	
Description:	This use case allows doctors to generate downloadable reports summarizing a patient's vital history, alert trends, and overall condition over a selected time period. These reports can be used for clinical analysis or shared with other professionals.	
Precondition:	Vitals data and alert records must already exist for the selected patient.	
Trigger:	Doctor clicks the "Generate Report" button on the patient's profile.	
Typical course of events	**Actor Action:** Doctor selects time range (e.g., last 7 days) and presses "Generate Report."	**System Response:** 1. Fetches vitals and alerts from the database. 2. Formats the data into a readable format (e.g., PDF or CSV). 3. Provides a download link.
Alternate courses	If no data exists in the selected range, display a message: "No data available."	
Conclusion	Doctor receives a downloadable report file.	
Post Condition	Doctor can use or share the report for diagnosis or consultation.	

7. Non-functional Requirements

7.1. Performance Requirements

The system is designed to run efficiently on computers/laptops. While high-end specifications are not mandatory, the computer should have at least 4GB RAM and a modern processor to ensure smooth real-time monitoring of simulated patient vitals. The system will handle multiple simulated patient data streams without significant lag.

7.2. Safety Requirements

As an academic prototype, the system prioritizes data integrity through automatic saving features that protect against data loss during unexpected interruptions. While industrial-level redundancy measures are not implemented, the application includes robust error handling to manage potential issues like sensor disconnections or data flow interruptions. These safety measures ensure continuous operation during classroom demonstrations and project evaluations.

7.3. Security Requirements

The system implements role-based access control to ensure proper data security. Doctors and nurses are granted different levels of authorization within the application - doctors have full access to view all patient vitals and historical data, while nurses can access real-time monitoring but with limited historical data privileges. All medical staff must authenticate using unique username and password credentials before accessing the system. While the patient data is simulated for this academic project, basic encryption methods are employed to demonstrate proper security practices for handling sensitive medical information. These measures ensure that only authorized healthcare professionals can access patient vitals data according to their designated roles.

7.4. User Documentation

A user manual and a self-maintenance guide will be delivered upon software instalment.

8. References

The template for the SRS was given by Mr. Ateeb Ahmed, everything else is original.

9. Appendices

Not Applicable.