Software Requirements Specification (SRS)

Project Information(PatientMonitoring System)

This mobile App is for the pathient party, Head office and Nurse/ caregivers three differect part in this app. patient party can see the update of their patients Head office can see the update of patient and nurse or caregiver and can controll the nurse task and to do list, and caregiver can give the uodate of patiend their to do list medicine update in every hour.

1. Introduction

In today's fast-paced world, healthcare systems require efficient monitoring and communication among patients, caregivers, and administrative staff. The **Patient Monitoring System** is a mobile-based application developed to bridge the communication gap between these parties and ensure better patient care through real-time data sharing.

Team no 9:
MD:Swopon Ali
ID:231071003
Al Abdullah
ID:231061001

1.1 Purpose

The main purpose of the Patient Monitoring System mobile application is to provide a centralized and efficient platform for managing patient care, ensuring smooth communication between patients' families, caregivers, and healthcare administrators.
This application aims to solve the problem of poor communication and lack of real-time updates in patient monitoring. Traditionally, family members often remain unaware of the patient's hourly condition, medication updates, or the caregiver's performance, leading to stress, confusion, and potential health risks. Similarly, healthcare management teams face difficulties in tracking caregiver activities and ensuring timely care for all patients.
The Patient Monitoring System addresses these challenges by allowing:
Patient parties to easily check their patient's health status, medication records, and daily care updates.
Caregivers/Nurses to record and share patient data in real time, including medicine intake, vital signs, and to-do tasks.
Head office administrators to monitor caregiver performance, assign tasks, and ensure proper coordination in patient management.
This app is useful because it improves transparency, accountability, and efficiency in healthcare services. It helps reduce communication delays, ensures that patients receive timely attention, and provides peace of mind to families by keeping them connected and informed at all times.

1.2 Scope

Main Features and Goals:

The app allows caregivers/nurses to update patient information such as medicine records, vital signs, and daily care tasks.

The app enables the **head office** to assign tasks, monitor caregiver activities, and manage patient reports.

The app allows **patient parties** to view real-time updates on their patient's condition, medication schedule, and caregiver performance.

The app includes a **notification and reminder system** for medicine times and important updates.

The app maintains **secure, role-based access** for different users (Patient Party, Head Office, Caregiver).

The goal of the app is to **improve communication, transparency, and patient care efficiency** through real-time monitoring.

The System Will:

Provide a mobile dashboard for all users with relevant information.

Store and update patient data in real time.

Send alerts and reminders for medication and task deadlines.

Allow the head office to control and manage caregiver activities.

The System Will Not:

Include a web or desktop version in the current phase.

Allow **direct medical diagnosis or treatment decisions** (it only records and tracks data).

Handle **payment or billing** features at this stage.

1.3 Objectives

- To design and develop a **user-friendly mobile application** for patient monitoring using **Flutter**.
- To implement **Firebase authentication and database** for secure login and real-time data storage.
- To enable **real-time updates** of patient information between caregivers, head office, and patient parties.

- To create a **task management system** for caregivers to follow their assigned duties and medicine schedules.

- To allow **head office administrators** to monitor caregiver performance and control task lists.

- To provide **instant notifications and reminders** for medicines, vital checks, and important updates.

- To ensure **data synchronization and reliability** across all user roles.

- To maintain **data privacy and security** for all patient records and activities.

- To develop a **scalable system architecture** that can support future expansion, such as a web version or analytics module.

1.4 • Target Users

The **Patient Monitoring System** mobile application is designed for three primary user groups who play different roles in the healthcare monitoring process. Each user has specific responsibilities and access levels within the system.

1. Patient Party

Family members or guardians of the patient.

They can view real-time updates about the patient's health status, medicines, and caregiver activities.

Receive notifications about medicine times, emergencies, and important reports.

Their main goal is to stay informed and reassured about the patient's condition without needing to visit physically.

2. Head Office / Admin

The healthcare organization or management team responsible for overseeing caregivers and patient care.

They can monitor all patients and caregivers from a central dashboard.

Manage to-do lists, assign tasks, and check daily reports.

Ensure that each patient receives proper care and that caregivers perform their duties efficiently.

2. Nurse / Caregiver

The person directly responsible for the daily care of patients.

They update patient information, medicine intake, vital signs, and other observations in real time.

Follow assigned tasks from the head office and mark them as completed.

Their role ensures accurate, timely, and continuous care for each patient.

2.1 Definitions, Acronyms, and Abbreviations

Term / Acronym	Definition / Description
App	Short form of "Application," referring to the Patient Monitoring System mobile application.
Patient Party	The family members or guardians of the patient who can monitor patient updates through the app.
Head Office / Admin	The management or supervisory body that oversees caregivers and manages patient data and task lists.
Caregiver / Nurse	The healthcare worker responsible for providing daily care to patients and updating their status in the app.
Task List / To-Do List	A list of assigned duties or actions that the caregiver must complete for each patient.
Vitals	Key health indicators such as heart rate, temperature, blood pressure, etc., that are monitored and recorded.
Firebase	A backend platform by Google used in the app for real-time database, authentication, and cloud storage.
Authentication	The process of verifying user identity before allowing access to the system.
Dashboard	The main screen in the app where users can view updates, tasks, and reports based on their role.
Notification	An alert or message sent by the app to inform users about important updates, tasks, or medicine times.
UI (User Interface)	The visual part of the app through which users interact with the system.
UX (User Experience)	The overall experience and satisfaction users feel while using the app.

3. Overall Description

The **Patient Monitoring System** is a mobile-based healthcare management application designed to improve communication, transparency, and efficiency between **patient parties**, **caregivers**, and the **head office (administrators)**. The system provides real-time updates about patients' health, medicine schedules, and caregiver activities, ensuring that all involved users remain connected and informed.

This app focuses on **remote patient monitoring** by allowing caregivers to input data such as medicine updates, vital signs, and task completion. The head office can supervise these activities through a centralized dashboard, while patient families can monitor their loved one's condition anytime, anywhere.

The system uses a **role-based structure** to ensure that each user type has specific permissions and access:

The **Caregiver/Nurse** enters and updates patient information in real time.

The **Head Office** monitors, manages, and assigns tasks or care duties.

The **Patient Party** views the patient's updates and receives notifications about important events or activities.

By integrating **Flutter** for cross-platform development and **Firebase** for real-time database and authentication, the system ensures a secure, scalable, and reliable performance.

2.1 Goals of the System

To simplify and automate the patient monitoring process.

To improve communication between caregivers, head office, and patient parties.

To provide real-time and transparent updates on patient conditions.

To enhance efficiency and accountability in healthcare services.

2.2 System Environment

Platform: Mobile application (Android & iOS using Flutter).

Backend: Firebase (Database, Authentication, Cloud Storage).

Users: Patient Party, Head Office, Nurse/Caregiver.

Network Requirement: Internet connection for real-time synchronization.

3.1 Product Perspective

The **Patient Monitoring System** is a **standalone mobile application** developed using **Flutter**, designed to operate as an independent platform for managing patient care and monitoring activities. However, it integrates with **Firebase** services to handle essential backend operations such as **authentication, real-time database, and cloud storage**.

The app does not rely on any external hospital management systems or third-party APIs in its initial phase. Instead, it provides a self-contained environment where all user interactions—such as patient updates, task management, and notifications—are processed through Firebase's cloud infrastructure.

The use of Firebase ensures **real-time synchronization** between the three main user groups (Patient Party, Head Office, and Caregiver) and provides **secure data handling** through built-in authentication and database management.

Overall, the app functions as a **mobile-first solution** that connects users through the internet, enabling efficient communication and transparent patient monitoring within a healthcare ecosystem.

.

3.2 Product Functions

The **Patient Monitoring System** mobile application provides several key functions to support patient care, task management, and real-time monitoring. Each function is designed to meet the specific needs of the three user roles: **Patient Party**, **Head Office**, and **Caregiver/Nurse**.

Main Functionalities:

User Registration and Login:
Secure authentication system using Firebase for all user types (Patient Party, Head Office, Caregiver).

Profile Management:
Users can manage and update their profile information according to their role.

Patient Information Update:
Caregivers can record and update patient data, including vital signs, medicine intake, and daily reports.

Task/To-Do Management:
The Head Office can create, assign, and monitor caregiver tasks, while caregivers can mark them as completed.

Patient Monitoring Dashboard:
The Head Office and Patient Party can view real-time updates of patient conditions, progress reports, and caregiver activities.

Medicine and Schedule Reminders:
Automatic notifications for medicine times and other important activities.

Real-Time Data Synchronization:
All data and updates are instantly synchronized through Firebase to ensure all users see the latest information.

Reports and Logs:
The system stores historical patient reports, caregiver logs, and completed tasks for review.

Notifications and Alerts:
Push notifications keep all users informed about updates, emergencies, or task deadlines.

Role-Based Access Control:
Each user has specific permissions based on their role to ensure data privacy and security.

3.3 User Classes and Characteristics

The **Patient Monitoring System** is designed for three primary user classes, each with distinct roles, responsibilities, and technical proficiency requirements.

1. Patient Party

Role: Family members or guardians of the patient.

Responsibilities: View patient updates, track medicine schedules, and receive notifications about patient care.

Technical Characteristics: Basic mobile app usage skills; ability to read updates and respond to notifications.

Access Level: Read-only access to patient information and notifications; cannot modify patient or caregiver data.

2. Head Office / Admin

Role: Supervisors or administrators who oversee patient care and caregiver activities.

Responsibilities: Assign tasks, monitor caregiver performance, manage patient data, and generate reports.

Technical Characteristics: Comfortable with mobile apps and dashboards; basic knowledge of task and report management.

Access Level: Full access to patient records, caregiver tasks, and administrative functions; can add, edit, or delete tasks.

3. Nurse / Caregiver

Role: Healthcare professional responsible for daily patient care.

Responsibilities: Update patient status, record vitals and medicines, complete assigned tasks, and communicate with head office.

Technical Characteristics: Familiar with mobile app usage; able to log data and follow task lists regularly.

Access Level: Can update patient data, mark tasks as completed, and view assigned tasks; cannot access other caregivers' private data.

3.4 Operating Environment

The **Patient Monitoring System** mobile application is developed to operate in a modern mobile environment, using cross-platform technologies and cloud services for backend management. The operating environment includes the following components:

Platform:

Android (primary target)

iOS (optional / future support)

Framework:

Flutter – for cross-platform mobile app development with a single codebase.

Backend / Cloud Services:

Firebase – for real-time database, authentication, cloud storage, and notifications.

Development Environment / IDE:

Android Studio – for app development and emulator testing

VS Code – alternative lightweight code editor for Flutter development

Database:

Firebase Realtime Database / Firestore – stores patient data, caregiver logs, and task lists.

Network Requirements:

Internet connection is required for real-time synchronization and notifications.

Offline caching can be supported for limited local data access if needed.

Device Requirements:

Smartphone or tablet with Android 8.0 or higher (or iOS 12.0 or higher).

Minimum 2GB RAM recommended for smooth app performance

3.5 Design and Implementation Constraints

The development of the **Patient Monitoring System** mobile application is subject to the following constraints and limitations:

Framework Constraint:

The app must be developed using **Flutter** to ensure cross-platform compatibility.

Platform Constraint:

The app must run on **Android 10 or above** (iOS support may be considered in future phases).

Backend Constraint:

All real-time data, authentication, and storage must use **Firebase** services.

Time and Resource Constraint:

Limited development timeline and team size may restrict the addition of advanced features such as analytics dashboards or offline-first capabilities in the initial release.

Connectivity Constraint:

Real-time updates and notifications require a **stable internet connection**. Offline functionality is limited to locally cached data.

Security Constraint:

User data privacy and role-based access must be maintained according to project scope; integration with external hospital systems is not included in this phase.

Device Constraint:

Minimum device specifications are **2GB RAM** and **Android 10+**, which may limit accessibility for older devices.

3.6 Assumptions and Dependencies

The development and operation of the **Patient Monitoring System** mobile application rely on certain assumptions and external dependencies, as outlined below:

Assumptions:

Users have **basic familiarity with smartphones** and can operate a mobile application.

Patient parties, caregivers, and head office staff will provide **accurate and timely data** for updates to be meaningful.

During registration, users are assumed to provide **valid email addresses** and necessary personal details.

Caregivers and head office staff will follow assigned tasks and update the app regularly.

Dependencies:

Internet Connection: Required for real-time data synchronization, notifications, and accessing cloud-stored information.

Firebase Services: The app depends on Firebase for authentication, database management, and cloud storage. Firebase servers must be operational at all times.

Mobile Platform: The app depends on Android (minimum version 10) and, optionally, iOS devices with supported versions.

Third-Party Services: Any push notifications, authentication verification, or cloud storage are dependent on external services integrated via Firebase.

These assumptions and dependencies ensure that the system functions as intended under normal operating conditions and provides real-time, reliable patient monitoring.

4. Specific Requirements

4.1 Functional Requirements

ID	Requirement Description	Priority
FR1	The system shall allow users to register using email and password.	High
FR2	The system shall allow users to log in based on their role (Patient Party, Head Office, Caregiver).	High
FR3	Caregivers shall be able to update patient information including vitals, medicine intake, and daily reports.	High
FR4	The Head Office shall be able to assign, edit, and monitor caregiver tasks and to-do lists.	High
FR5	Patient Parties shall be able to view real-time updates about their patient's health and caregiver activities.	High
FR6	The system shall send notifications for medicine schedules, task reminders, and important updates.	Medium
FR7	The system shall maintain logs of completed tasks and historical patient data for reporting.	Medium
FR8	Users shall be able to edit their profile information.	Low
FR9	The system shall implement role-based access control to restrict actions based on user type.	High
FR10	The system shall allow head office to generate daily or weekly reports of patient and caregiver activities.	Medium

4.2 Non-Functional Requirements

ID	Requirement Description	Type
NFR1	The app should load within 3 seconds.	Performance
NFR2	The system shall maintain data security using Firebase authentication.	Security
NFR3	The system shall synchronize data in real-time between all users.	Reliability

ID	Requirement Description	Type
NFR4	The app shall be responsive and usable on smartphones with at least 2GB RAM.	Usability
NFR5	The system shall provide notifications within 1 minute of a scheduled event.	Performance
NFR6	The app shall operate under Android 10+ and optionally iOS 12+.	Platform Compatibility
NFR7	The app interface shall be intuitive and user-friendly for all user classes.	Usability

4.3 External Interface Requirements

The **Patient Monitoring System** interacts with users, devices, and software platforms through well-defined interfaces to ensure seamless operation, data exchange, and usability.

1. User Interface (UI) Requirements

The app shall have a **role-based dashboard** for Patient Party, Head Office, and Caregiver, showing only relevant information.

Navigation shall be **intuitive**, using tabs, buttons, and menus appropriate for mobile devices.

Forms for registration, login, patient updates, and task management shall include **validation** for required fields and correct data formats (e.g., valid email addresses).

Notifications shall appear as **push messages** on the device, with clear information about medicine times, tasks, or updates.

Reports and patient history shall be **accessible through scrollable lists or cards** for easy viewing on small screens.

2. Hardware Interface Requirements

The application shall run on **smartphones or tablets** with a minimum of 2GB RAM.

The device must have **internet connectivity** (Wi-Fi or mobile data) for real-time updates and notifications.

The app shall use the device's **local storage** minimally for caching offline data temporarily.

The device must support **push notifications** for timely alerts.

3. Software Interface Requirements

The app shall integrate with **Firebase services** for:

Authentication (email/password login)

Real-time database for storing patient, task, and caregiver information

Cloud storage for reports and logs

Push notifications for reminders and alerts

The app shall run on **Android (10+)** and optionally **iOS (12+)** platforms.

The application shall use **Flutter framework** for cross-platform compatibility.

4. Communication Interface Requirements

The app shall communicate with Firebase over **HTTPS** to ensure secure data transfer.

Real-time updates between users shall be handled via **Firebase Realtime Database or Firestore**, ensuring instantaneous data synchronization.

Push notifications shall use **Firebase Cloud Messaging (FCM)** for delivering timely alerts to all users.

5. System Models (Recommended)

5.1 Use Case Diagram

The use case diagram represents the main interactions between the three user types (Patient Party, Head Office, Caregiver) and the system. It helps visualize how each user accesses specific functionalities of the app.

Actors:

Patient Party

Head Office / Admin

Caregiver / Nurse

Main Use Cases:

User Registration & Login

Profile Management

Update Patient Information (Caregiver)

View Patient Status (Patient Party & Head Office)

Assign and Track Tasks (Head Office)

Receive Notifications & Reminders

Generate Reports (Head Office)

Description:

Patient Party can view patient updates and receive notifications.

Caregiver can update patient information and mark tasks as completed.

Head Office can assign tasks, monitor caregivers, and generate reports.

5.2 Sequence Diagram

The sequence diagram shows the **process flow of a typical task**, such as updating patient information and notifying the patient party.

Example Flow:

Caregiver logs in → system authenticates via Firebase.

Caregiver updates patient information → system stores data in Firebase database.

Firebase triggers notification → Patient Party receives push notification.

Head Office dashboard updates → Head Office sees updated patient status and completed tasks.

Key Components:

Caregiver (Actor)

System / App

Firebase (Backend)

Patient Party (Actor)

Head Office (Actor)

Description:

The diagram shows **chronological order of interactions**, including login, data input, data storage, notification delivery, and dashboard updates.

6. Project Management

6.1 Development Tools

Tool/Technology	Purpose/Use
Flutter	For developing the mobile application's user interface and cross-platform functionality.
Firebase	For backend services such as authentication, real-time database, and cloud

Tool/Technology	Purpose/Use
	storage.
Android Studio	As the main IDE for developing, debugging, and testing the Flutter application.
GitHub	For version control and collaborative development.
Figma	For designing UI/UX mockups and layout planning.

7. Timeline

Phase	Task	Duration
Phase 1: Planning & Research	Identifying project idea, conducting background research, and requirement analysis.	1 week
Phase 2: Design	Creating wireframes, user interface design, and system architecture.	1 week
Phase 3: Development	Implementing frontend using Flutter and integrating backend with Firebase.	3 weeks
Phase 4: Testing & Debugging	Testing the app for bugs, fixing issues, and improving performance.	1 week
Phase 5: Final Review & Submission	Final documentation, report writing, and project submission.	1 week

8. Appendix

Type	Source/Reference	Purpose/Usage
Official Documentation	Flutter Documentation	For guidance on Flutter widgets, navigation, and state management.
Official Documentation	Firebase Documentation	For integrating authentication, database, and cloud storage.
API Reference	Google Maps API	For location tracking and map integration (if applicable).
Video Tutorial	YouTube – *Flutter & Firebase App Development Tutorials* by The Net Ninja	For learning Flutter-Firebase integration and project structure.
Blog/Article	Medium – *Best Practices for Flutter UI Design*	For improving UI/UX design quality.
GitHub Repository	Sample Flutter projects on GitHub	For referencing code structure and project organization.