UNIVERSITY OF CALGARY | Schulich School of Engineering

Term Project II SRS Medical Monitoring System

Prepared for:

Mahmood Moussavi

December 4, 2019

Prepared By:

Group 3: Zixin (James) Chen
Asif Bux
Blair Chau
Vaibhav Jadhav

TABLE OF CONTENTS

List of Figures

List of Tables

Glossary

Term	Definition
Actors	Actor represents a role that a human, hardware device or software can play.
API	An Application Programming Interface (API) is a bridge between a web app (frontend) and a database. In the context of web development, an API is defined as a set of specifications such as HTTP (Hypertext Transfer Protocol)

	requests along with a definition of the structure of response messages usually in XML or JSON format.
HTML and CSS	HTML is a mark-up language designed to be displayed in web browsers and it is often combined with Cascading Style Sheets (CSS) and JavaScript (JS). CSS describes how elements should be rendered on the screen.
JavaScript	JavaScript is a client-side scripting language which is used to add functionality for HTML elements and access HTML Document Object Model (DOM).
JSON	JavaScript Object Notation (JSON) is a light weighted data interchange format used to send and re
MMS	The Medical Monitoring System (MMS) is a platform which can aggregate a patient's vital measurements and facilitates monitoring of a patient's health for hospital staff and caregivers via features such as emergency alerts and medication reminders.
Sequence Diagram	A sequence diagram represents an object interaction arranged in time sequence and it models dynamic aspects of system.
SMS	Short Message Service (SMS) is a text messaging component most commonly associated with mobile device systems.
Twilio	Twilio is a communication platform to make and receive phone calls, send and receive text messages using its web service APIs.
Use Case Diagram	Use Case Diagram is a dialog between actors and system. It shows the relationship between the user and the different use cases in which the user is involved.

1.0 Introduction

1.1 Purpose

The product whose software requirements are specified in this document is a medical monitoring system (MMS). The scope of this system is limited to the core functionality required for the monitoring and management of patient conditions such as heart rate, body temperature, blood pressure, providing reminders to patient caregivers about medicine administration, the ability to record medical condition metrics on the hospital's server, and sending an appropriate emergency notification when the patient's heart rate or blood pressure is at an extreme condition. Functionality such as the processing/filtering of medical sensor data, the system for displaying vital measurements in real-time and any additional handling of emergency situations beyond sending an appropriate notification are all beyond the scope of this product.

1.2 Overview of the Document

1.2.1 Document Conventions

For the document convention following styles were used:

- Heading 1: Sample Text 1.0 Introduction, font size 16 and font type Calibri Light.
- Heading 2: Sample text 1.2 Overview of the Document, font size 13 and type Calibri Light.
- Heading 3: Sample Text 1.2.1 Document Convention, font size 12 and type Calibri Light.
- Normal Text throughout the document is font size 12 and font type Calibri.

1.2.2 Readers and Audience

This document is intended for the reference of developers and testers of this medical monitoring application and any future related applications; as well as medical and hospital personnel such as doctors, nurses, and technicians responsible for maintenance of the medical monitoring system components. The document begins with an overall description of the application, which is followed by a description of each function of the application in greater detail. Following this, a description of the external requirements is provided, after which there is a list of all non-functional requirements and issues regarding quality control. The document concludes with a summary of the tentative release plan for the application.

For managers and hospital technical team of the application, relevant sections first include the product scope of this application (Section 1.4), followed by the overall description (Section 2). Users of a specific system function should reference the section pertaining to that function directly (Section 4). Managers should also consult the external interface requirements (Section

4) and the non-functional and quality requirements (Section 5).

For all other readers of this document, an appropriate point to begin is with the overall description (Section 2). Refer to system features (Section 4) and the user interface requirements (Section 7) for relevant details. Refer to the release plan (Section 9) for overall timeline and deadline requirements. Quality requirements and non-functional requirements (Section 8) are available for system information beyond the core functionality of the application.

1.3 References

- Barry, D. K., & Dick, D. (2013). *Web services, service-oriented architectures, and cloud computing: the savvy managers guide*. San Francisco, CA: Morgan Kaufmann.
- Bittner, Kurt, and Ian Spence. *Use Case Modeling*. Addison-Wesley, 2008.
- HITInfrastructure. *"Healthcare Data Storage Options: On-Premise, Cloud and Hybrid Data Storage."* HITInfrastructure, 9 May 2019, https://hitinfrastructure.com/features/healthcare-data-storage-options-on-premise-cloud-and-hybrid-data-storage.
- *"Oracle Healthcare."* Healthcare | Oracle Canada, https://www.oracle.com/ca-en/industries/healthcare/.
- *Strategies and Technologies for Healthcare Information: Theory into Practice*. Springer-Verlag New York, 2013.

1.4 Product Scope

The purpose of the medical monitoring application is to provide a platform which can aggregate a patient's vital measurements to make analysis of these measurements easier, provide instant notification to medical staff if the patient's condition worsens, and provide reminders to the patient to take medication on time.

A visual representation of the application is shown in Figure 1.

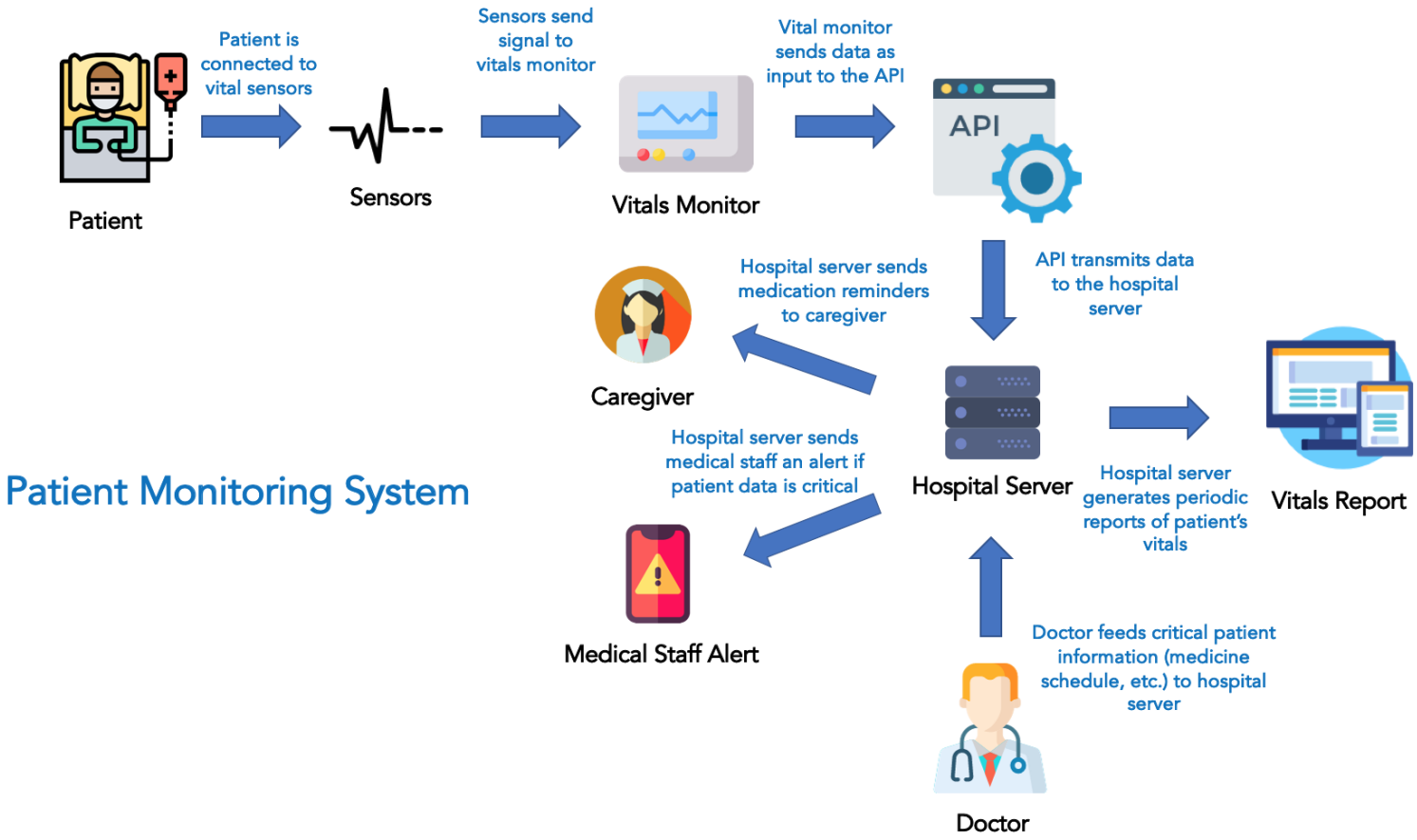

Figure 1: Product Scope for Medical Monitoring System

As shown in the figure, the process for the application is as follows:

- The patient's vitals are read by various sensors (heart rate, blood pressure, and temperature).
- The system displays the measured vitals on the screen of the vital signs monitor.
- The vitals monitor sends the data as input to a database API, which then transmits the data to the on-premise hospital server.
- The server analyses the data and can generate periodic reports of the patient's vitals (occurrence of report generation is decided by medical staff on demand).
- If the vital measurements exceed a standard stable condition, the server sends the medical staff an alert that the patient data is critical via an API (not shown in diagram)
- During the patient intake/registration process, the patient's doctor enters critical patient information such as medication dosage and schedule (if applicable) to the hospital server via an application on hospital computers to be saved in the hospital database.

- The application's server sends the patient's caregiver notifications about medication schedule and dosage at appropriate times via our mobile application.

The use case diagram for the system is shown in Figure 2.

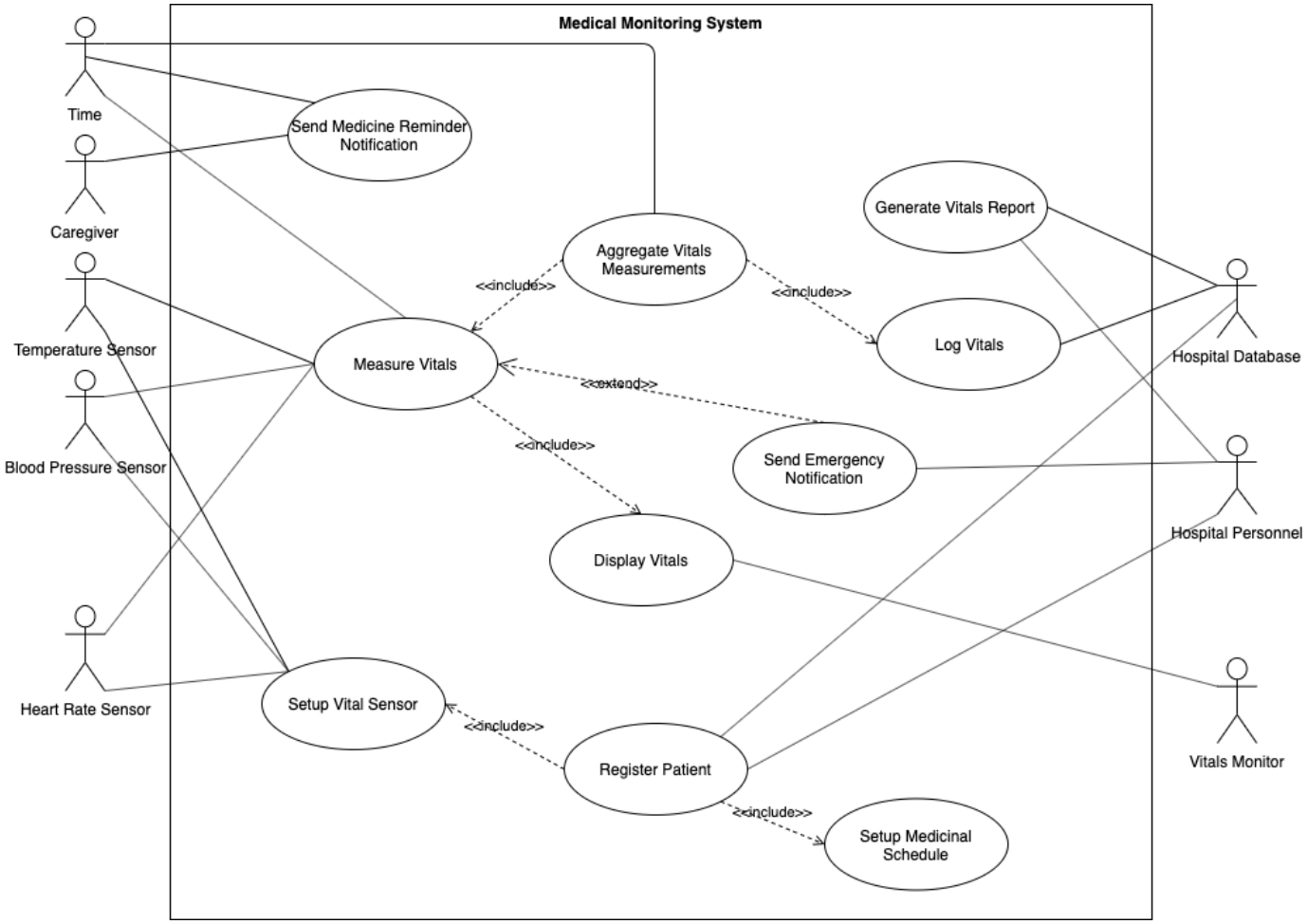

Figure 2: Use case Diagram

A brief overview of the various use cases in this diagram is listed here, where the use cases are decomposed and described in greater detail in Section 4.

- The time actor will trigger a notification for medication reminders at preconfigured times. The time actor also periodically triggers the processes of measuring and of aggregating the patient's vitals measurements to display on the vitals monitor and store in the hospital database respectively.
- The caregiver actor will receive the medication notification reminders on their mobile application.
- The temperature, blood pressure, and heart rate sensor actors measure the patient's sensor to be recorded, displayed and monitored by the system.

- The hospital database actor logs the vitals received from the vitals machine and can generate a periodic report of the patient's vitals/condition upon request.
- The hospital personnel actor receives emergency notifications if the patient's vitals exceed a standard stable threshold in order to respond to emergency situations. The hospital personnel actor also submits the requests for vitals reports generated from data stored in the hospital database.
- The vital signs monitor actor displays the aggregated patient vital signs data

It is important to note that this product contains both hardware and software components. The hardware components consist of all vitals sensors, the vital signs monitor, and the vitals machine. The software components consist of an on-premise application (physically located within the hospital), which has read and write access to the hospital server and database, as well as a mobile application, which only has read access to the hospital database and server since it is not located on-premise. Only authorized medical personnel have access to the on-premise application, and it is used for registering patients, inputting patient medication information, and accessing vitals reports. The mobile application is primarily used by caregivers to attend to patients at their place of residence and consists of the medication notification and reminder features.

2.0 Overall Description

2.1 Product Perspective

This system will be integrated with the existing hospital infrastructure, namely the hospital server and patient database. Our application will require access to the hospital server and database in order to function, as shown in Figure 3. The external interface relating our application the existing hospital infrastructure includes an API that will execute the user's tasks - such as reading or writing to the database - and return results to the user. It also includes access to an existing API that can read data from the hospital database as well as write to the hospital's network system to record patient data reports. For example, in Alberta this application will write to Alberta NetCare, which is the central patient record repository. The doctor will have access to vital reports as well as all past patient medical history.

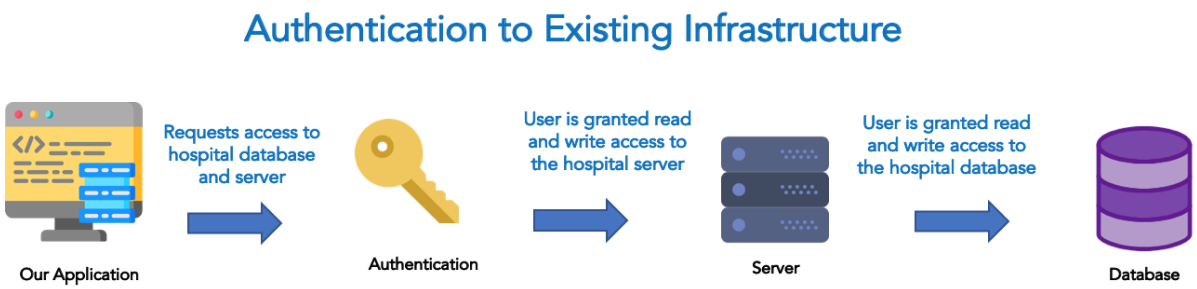

Figure 3: Authentication to Existing Infrastructure

2.2 Operating Environment

The on-premise application will consist of a desktop application on a Windows operating system. The system's software can be divided into front-end and back-end components, where the front-end will consist of an application written in Java and C++, and the back-end will consist of the on-premise hospital servers and databases, which it will require access to as discussed in section 2.3.

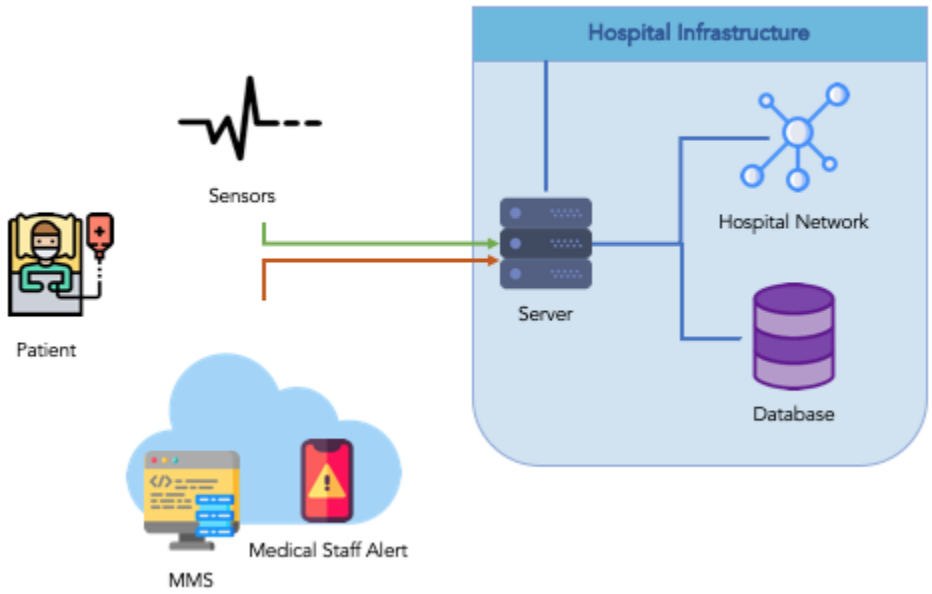

Figure 4: Operating Environment of MMS

Since the application will be hosted on-premise, it will require access to the hospital's physical servers and databases as well as the hospital's local network. Figure 4 depicts how the vitals sensors connect to the hospital server, which writes data to the hospital network and database.

The MMS and Medical Staff Alert system also connect to the hospital server, which provides read access to the hospital network and database.

The operating environment for users to run the on-premise application is any Windows computer. The caregiver mobile application will require Android version Lollipop – 5+ or iOS version 11+.

2.3 System Constraints

Since the application is hosted on-premise, it requires hospital technical staff to constantly maintain, monitor and expand the servers and databases. The application is thus dependent on the hospital staff's technical ability. This could result in the application's functionality being negatively affected if the physical hospital hardware has downtime or is not properly maintained. It would also require extensive licensing agreements with the hospital as it must agree to these maintenance and expansion cost terms. Upon completion of the application, the hospital technical team will be responsible for updating and maintaining the application. These developers must be skilled in the languages we will use to build the application, and we must provide extensive documentation to support further development. Furthermore, the hospital technical team will be responsible for the maintenance and updates of the hardware components of our application, such as the vital signs monitor and machine as well as all associated sensors. They must be skilled in maintaining these as well. In terms of security, the application would be dependent on the security implementation by the hospital technical staff, and if security is not upheld to the highest standard, a data leak may result through our application which could hold high consequences as the data involved is very sensitive. Local disasters are also a risk factor in on-premise application hosting.

2.4 Assumptions and Dependencies

This application assumes granted access to the hospital database and server for it to function as specified. It also assumes granted access to the hospital network, such as Alberta NetCare, in order to log patient vitals reports. The mobile reminder system is beyond the scope of this SRS document; however, a sample user interface is shown in Section 7. The main assumption involving the medical emergency notifications to the medical staff is that the notification will come in various forms, such as phone calls, Short Messaging Service (SMS), a notification on the application, or a physical hardware that alerts medical staff in the hospital.

3.0 User Classes and Characteristics

User classes are defined into five categories: caregiver, hospital personnel, hospital database, vitals sensors, and vital signs monitor. Although time is used as a trigger for some events, we are not considering it as a user class as it is an automated system feature. Patients are admitted

to the hospital by the hospital personnel, and their vital signs data (temperature, blood pressure, and heart rate) will be collected through the sensors. The vital signs monitor will display the sensor data in real time. Hospital personnel include nurses, and doctors who have medical expertise and are attending to the patient. The hospital database collects data from the sensors and patient information. The caregiver administers medication when reminded to do so by the system.

1. ***Caregiver***
    - Receives notification to administer specific drugs in specific dosages at specific times from the system.

2. ***Hospital Personnel***
    - Set up the vitals sensors to the system.
    - Monitor vital signs data collected from the patient sensors stored in the hospital database.
    - Receive emergency notification in case of abnormal vital signs or loss of connection from the sensors.

3. ***Sensors***
    - Initialized by hospital personnel to start collecting vital signs data from the patient.
    - Collect information from the patient on a regularly timed basis (every 0.5 seconds, 1 second, etc.)

4. ***Vital Signs Monitor***
    - Receives vital signs data from the system.
    - Displays vital signs data to the patient's room.

5. ***Hospital Database***
    - Receives patient information and vital signs sensor information when patient is first admitted into the hospital and creates a log file to prepare receive vital signs data.
    - Receives regularly scheduled logs of aggregated vital signs data from all the sensors (temperature, blood pressure, heart rate, at every 0.5 seconds, 1 second, etc.) and stores that data.
    - Generates a vital signs report for the hospital personnel based on their requirements (time frame, specific sensors, auto-refresh for active monitoring, etc.).

4.0 System Features and Use Cases

4.1 Use Case 1 (UC1) - Patient Registration

4.1.1 Description of Use Case

This process allows hospital staff to register the patient with the hospital database. It will also set up the vitals sensors for monitoring and create a medicinal schedule for the patient if applicable. The sequence diagrams for this feature are shown in Figure 5.

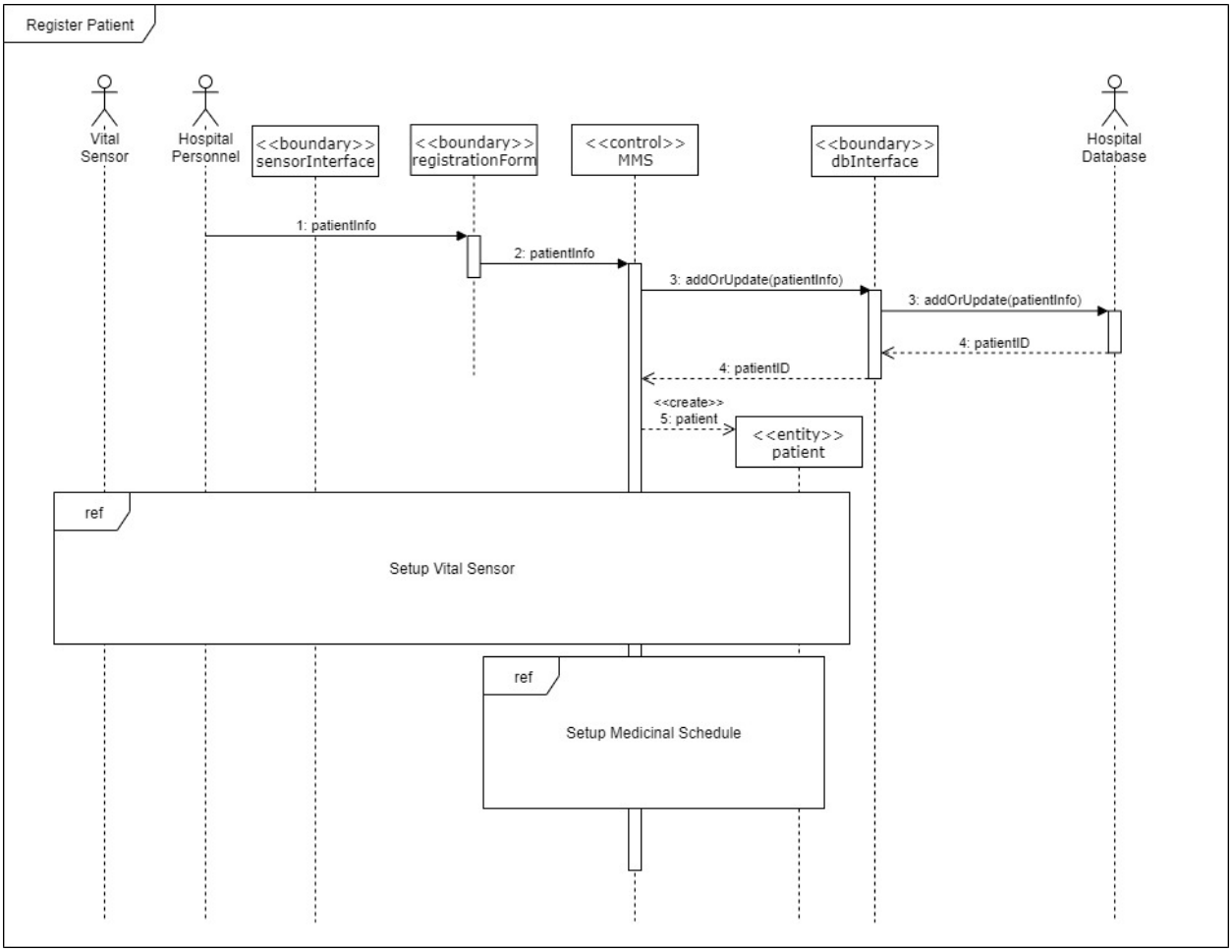

Figure 5: UC1: Patient Registration Sequence Diagram

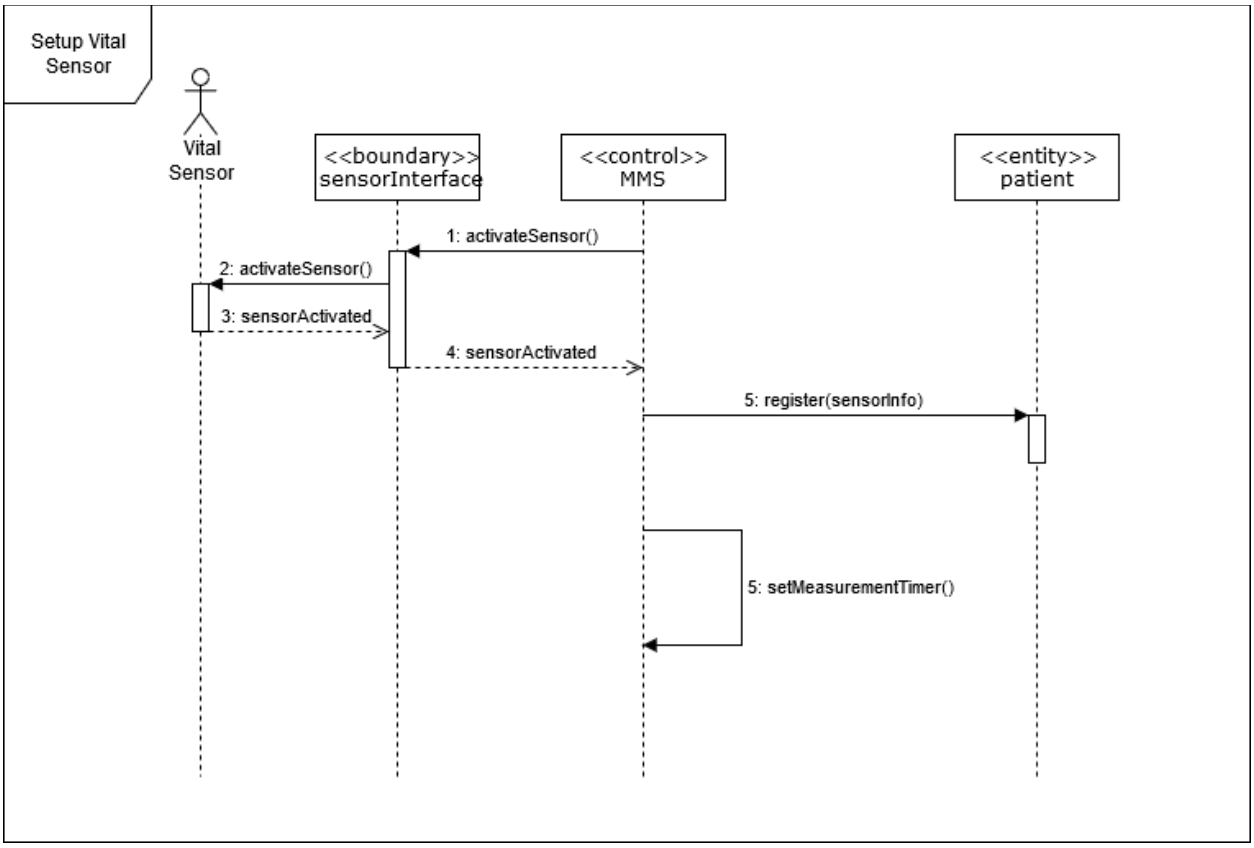

Figure 6: UC1: Vital Sensor Setup Sequence Diagram

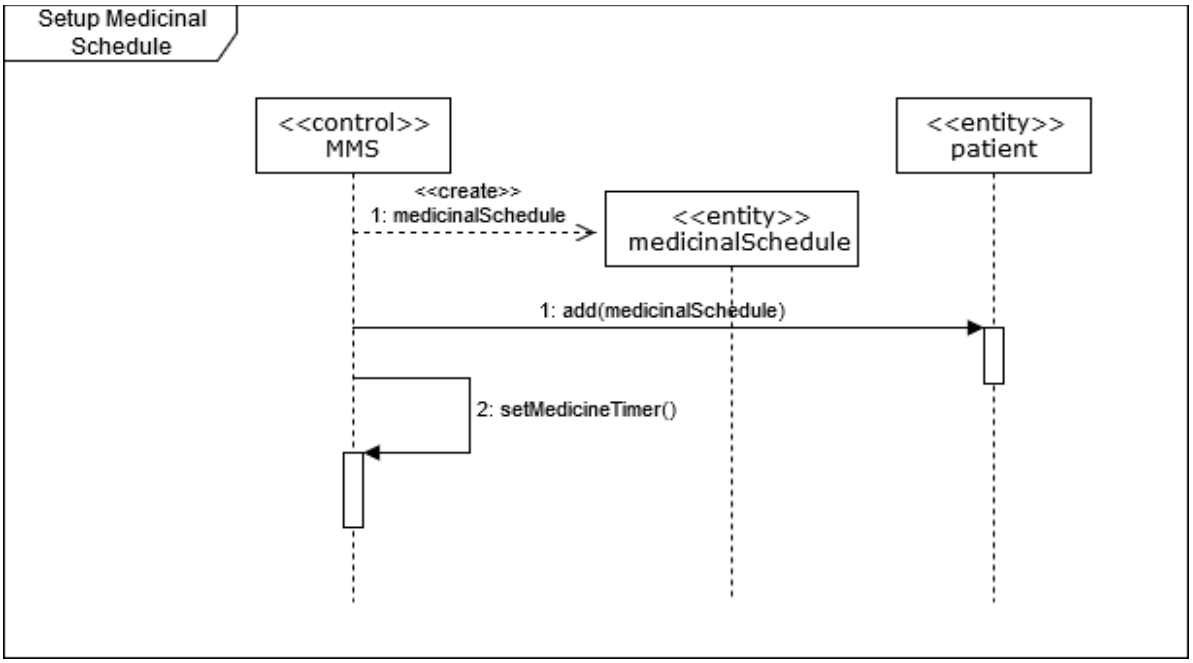

Figure 7: UC1: Medicinal Schedule Setup Sequence Diagram

The SensorInterface boundary class provides the interface for all communication and control of the vitals sensors, including activation and conversion of measured signals into numerical vitals readings. The RegistrationForm boundary class provides the user interface for hospital personnel to enter patient information when initially registering a patient. The DBInterface boundary class provides the interface for all reading and writing of data from and to the hospital database. The MMS control class is the main medical monitoring system controller directing the workflow of the software system.

4.1.2 Preconditions and Promises

Preconditions:

- A set of patient information (name, address, Health ID, sex, medical history, emergency contact information).
- Medicinal scheduling information (name of drug, time to consume, amount to consume).

Promises:

- Saves patient information as well as hospital admittance information (room number, bed number, sensor information) onto the hospital database.
- Activates vital sensors and starts the monitoring process.
- Initializes the medicinal schedule and timers to send alerts for medicine consumption.

4.2 Use Case 2 (UC2) - Vitals Measurements Aggregation

4.2.1 Description of Function

This process allows data to be regularly collected from the vitals sensors attached to the patient and displays the current vitals on the vitals monitor. It will also send an emergency notification if vital measurements move beyond a safe range. The process clusters measured sensor readings over a set time interval to log to the database periodically as well. The sequence diagrams for this feature are shown in Figure 8.

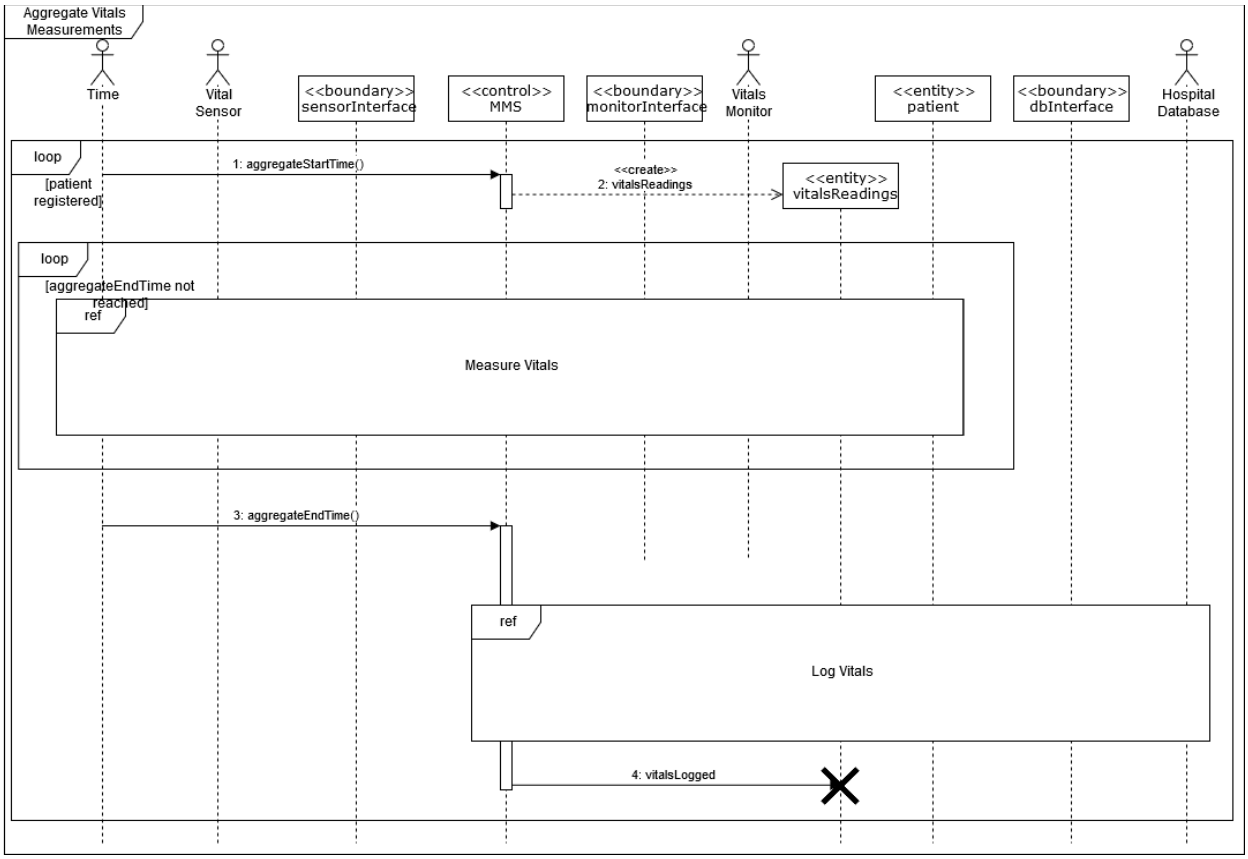

Figure 8: UC2: Vitals Measurements Aggregation Sequence Diagram

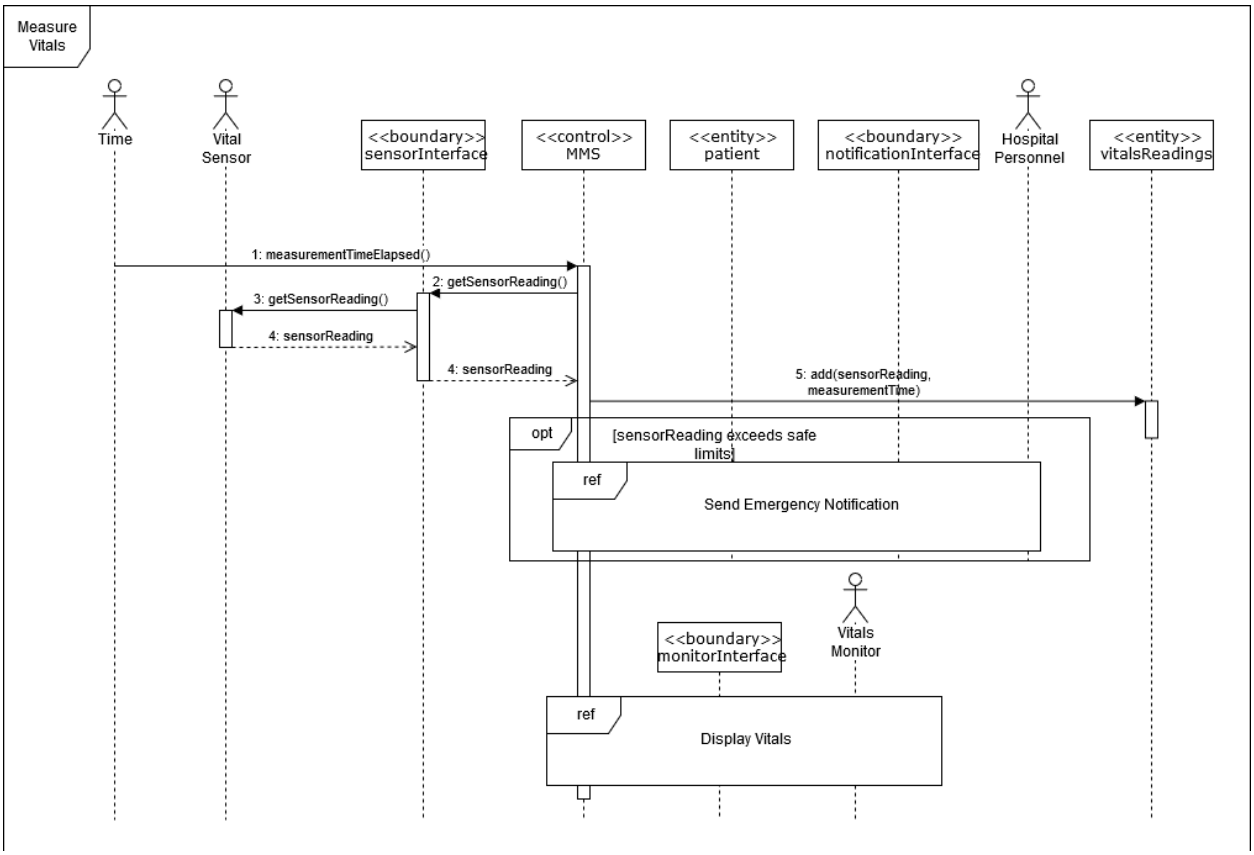

Figure 9: UC2: Vitals Measurement Sequence Diagram

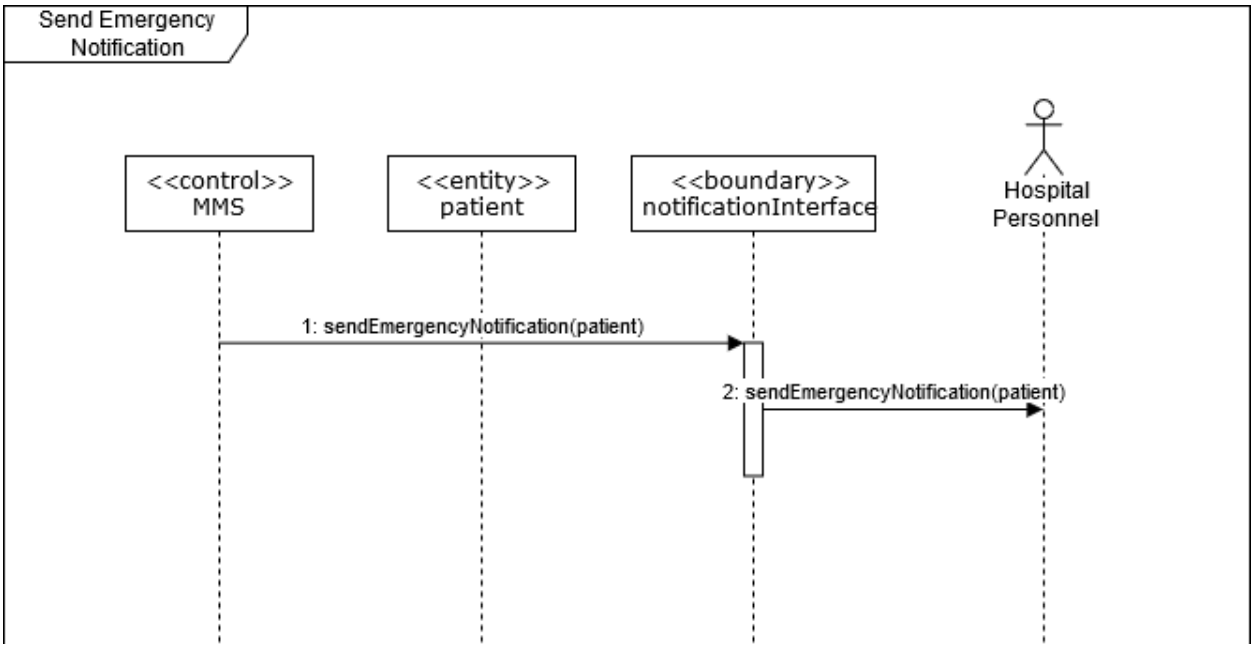

Figure 10: UC2: Emergency Notification Sequence Diagram

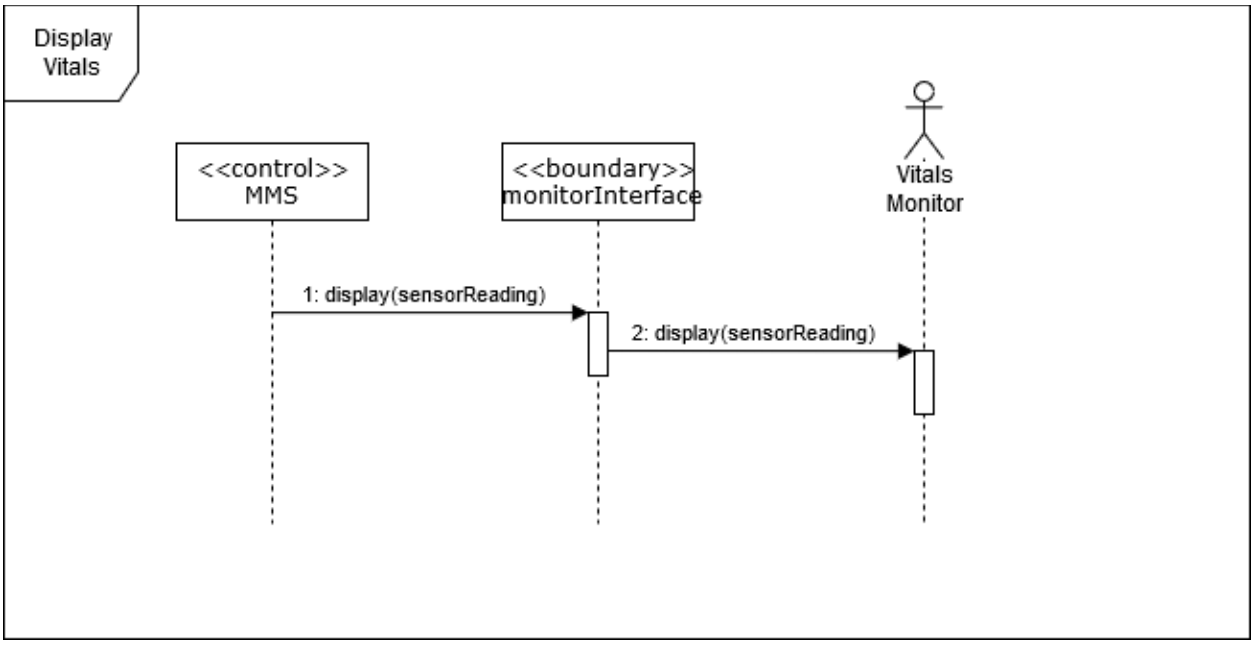

Figure 11: UC2: Vitals Display Sequence Diagram

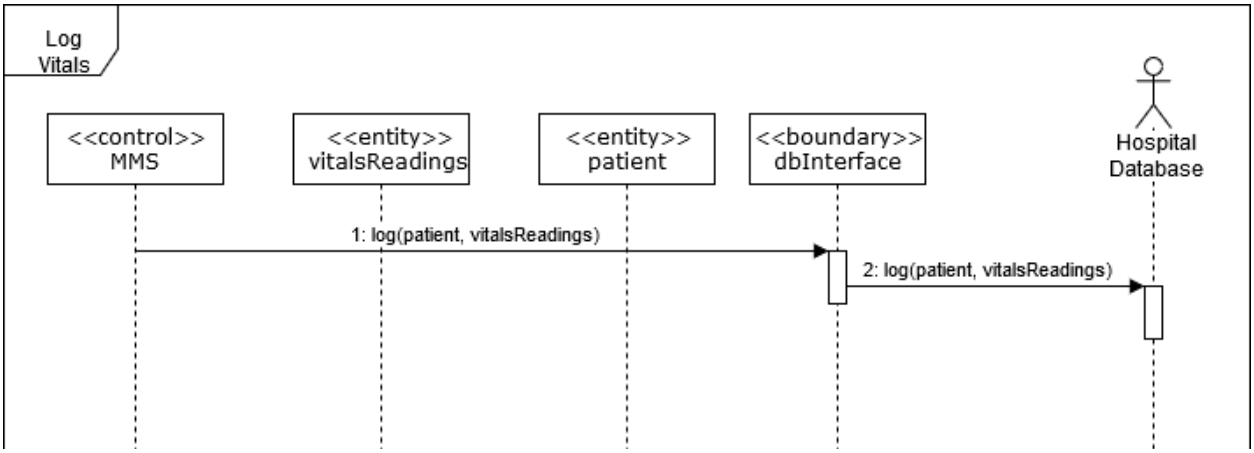

Figure 12: UC2: Vitals Logging Sequence Diagram

The MonitorInterface boundary class provides the interface for displaying live sensor data to the vitals monitor. The NotificationInterface boundary class provides the interface for sending both scheduled medicinal notifications to caregivers and emergency notifications to hospital personnel.

4.2.2 Preconditions and Promises

Preconditions:

- A registered patient.
- Activated vitals sensors.

Promises:

- Continuously obtains and displays vital sensor readings on the vitals monitor.
- Sends an emergency notification if vital measurements are outside a safe range.
- Periodically logs batches of vital sensor readings to the hospital database.

4.3 Use Case 3 (UC3) - Medicinal Notification

4.3.1 Description of Function

This process sends a notification to the caregiver at scheduled times to remind them to administer medication to the patient. The sequence diagram for this feature is shown in Figure 13.

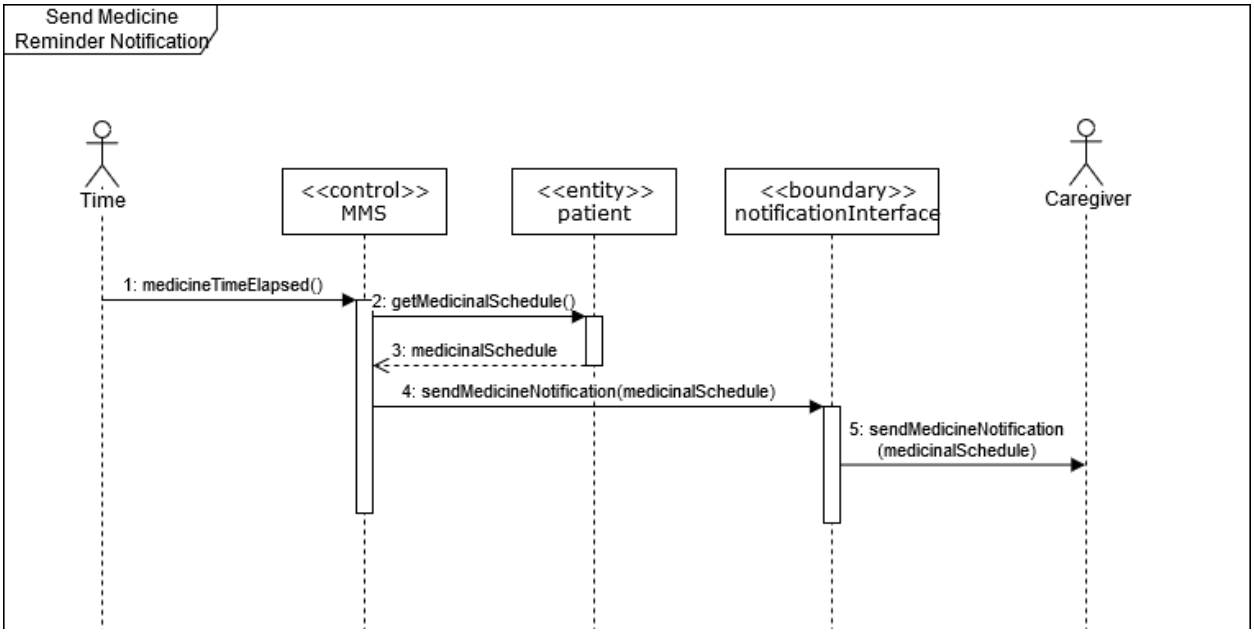

Figure 13: UC3: Medicinal Notification Sequence Diagram

4.3.2 Preconditions and Promises

Preconditions

- Medicinal scheduling information assigned to a registered patient.

Promises

- Sends a notification to the patient's caregiver to administer medication, including the medication name and dosage.

4.4 Use Case 4 (UC4) - Vitals Report Generation

4.4.1 Description of Function

This process allows hospital personnel to generate and view reports of a patient's historical vital measurements. The sequence diagram for this feature is shown in Figure 14.

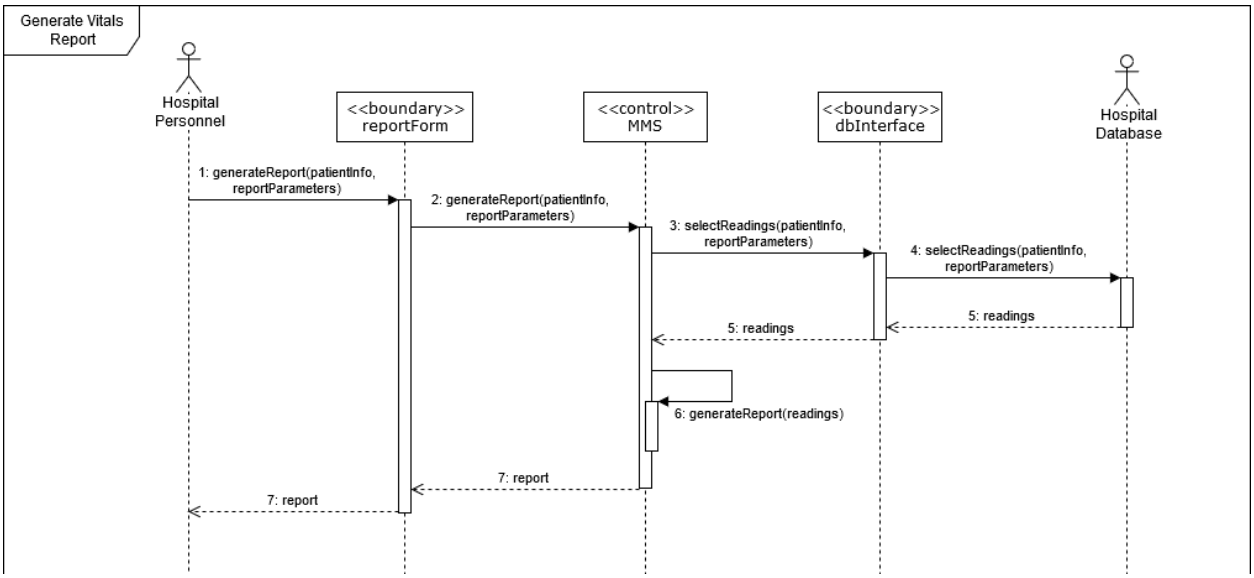

Figure 14: UC4: Vitals Report Generation Sequence Diagram

The ReportForm boundary class provides the user interface for hospital personnel to enter patient information and report parameters (start and end times, measurement types, etc.) when generating reports.

4.4.2 Preconditions and Promises

Preconditions

- A registered patient.
- A valid report request from hospital personnel with valid parameters (time frame, specific sensors, etc.)

Promises

- Generates a report based on the provided request.

5.0 Summary of Functional Requirements

Patient Monitoring system consists of a Vital monitor which is connected to a patient through sensors and a web application where doctor and caregiver can track and monitor patient's vital information. Functional requirements overall system is given below:

Table 1: List of Functional Requirements

RE #	Description
RE 1	Vital monitor should be able to measure and display the patient's heart rate, blood pressure and temperature through HR, BP and temperature sensor.
RE 2	Vital monitor should transfer patient's data to the hospital's server and database through APIs.
RE 3	Doctor must register and login to the web application in order to monitor patient's vital information.
RE 4	Doctor should be able to configure contact information of caregivers on the application and feed patient's data.
RE 5	Doctor should be able to find patient by entering the patient's name and number/id.
RE 6	Once patient is found, the doctor should be able to track and monitor patient's vital information such as heart rate, blood pressure and temperature.
RE 7	Vital monitor should continuously transfer patient's data to the hospital's server and if heart rate or blood pressure goes beyond range then application should send alerts (calls and text messages) to doctor and caregiver.
RE 8	Hospital server should be able to send medication reminders to caregiver.
RE 9	Periodic reports of patients should be generated automatically by web application.

6.0 Product Conceptual Model

A conceptual model of the system and its components is shown in Figure 15. For definitions of example attributes please see Appendix A.

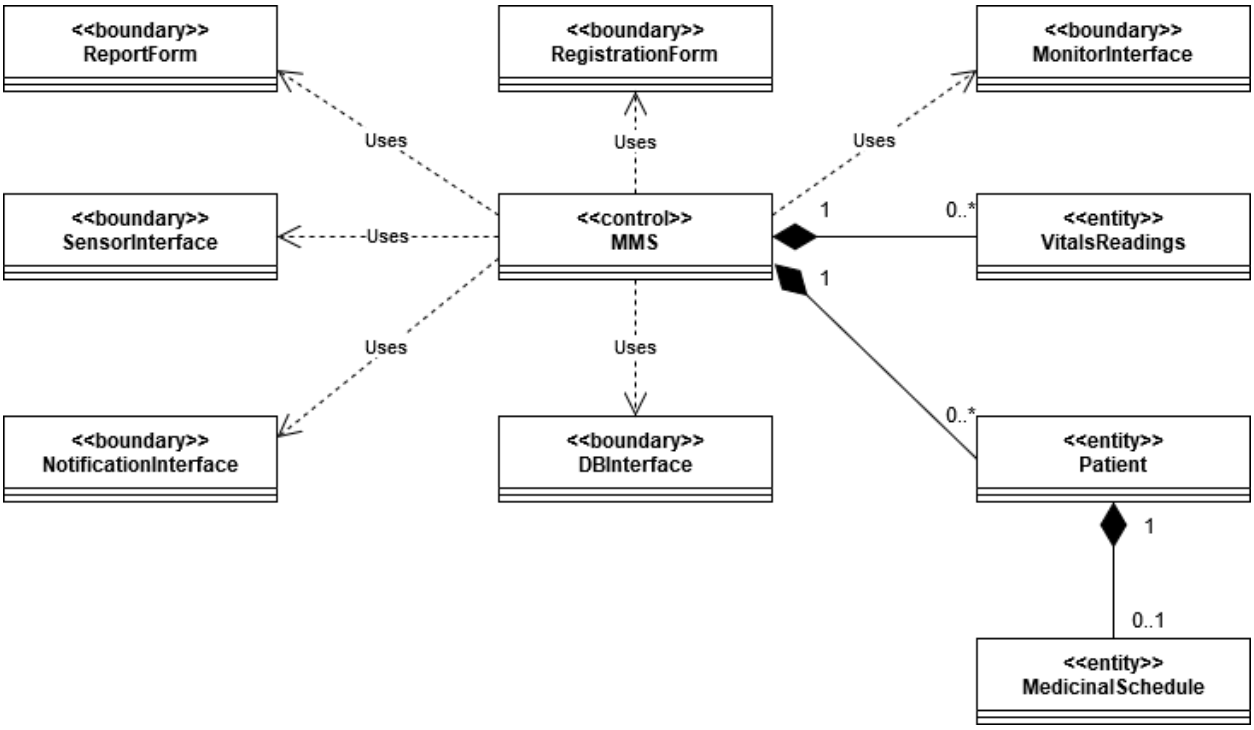

Figure 15: Medical Monitoring System Conceptual Class Diagram

7.0 External Interface Requirements

7.1 User Interfaces (UI)

The interfaces for this Medical Monitoring System consist of a vital monitor connected to a patient and a web application where the patient's doctor or medical team member can login to monitor the patient's heart rate, blood pressure and other data. In the event of emergency situations, the doctor and/or caregiver receives an alert on their mobile device. This alert is triggered from the hospital server.

7.1.1. Emergency Alerts

Emergency alerts are received by doctors and caregivers in the event that a patient's vitals ever enter a critical state. Alerts can be configured in various forms, such as an alert on the medical personnel's portal on the web application, an automated call or a text message from the hospital. Alerts will contain information including the patient's condition, name, Alberta health id, patient number, the caregiver's name and contact information.

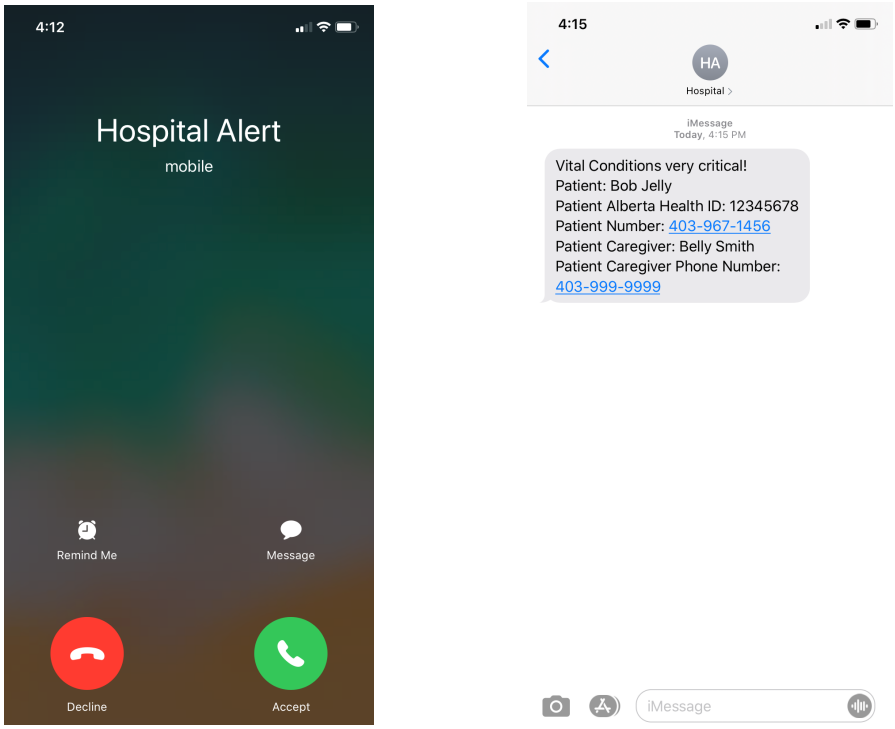

Figure 16: Emergency Alerts UI

7.1.2. Medication Reminders to Caregiver

The hospital server sends out medication reminders to caregivers to administer medication to patients at scheduled times. This medication reminder is sent in the form of a text SMS and contains information including the patient's name, the medication name/type, dosage and scheduled administration time.

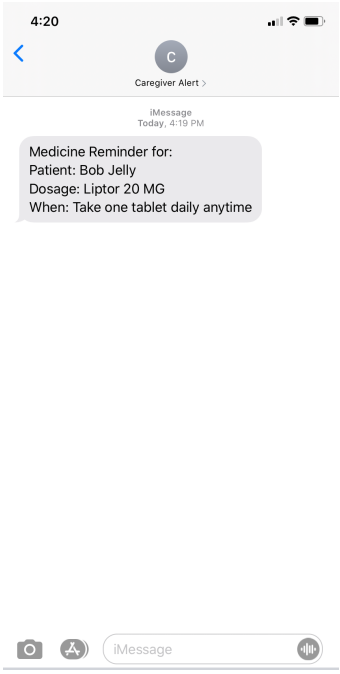

Figure 17: Medication Reminders UI

7.1.3. Web application

The web application is the main application for medical staff to track and monitor the patient's vital information such as blood pressure, heart rate and temperature. Data is transmitted from the vitals machine to hospital web server via appropriate APIs, which is later retrieved by this web application.

Doctors can register and login to this application and find patients by their name and patient number, in order to access and monitor any desired patient data. They have access to configure caregivers contact information so that caregivers receive medication reminders and emergency alerts, and to enter in any relevant patient data manually. Doctors can also generate periodic reports from this web application.

Figure 18 below displays a sample interface for the vital information of a patient. This patient information screen is where doctors can monitor patient's heart rate, blood pressure and temperature. The application also displays basic information of the patient such as contact information, address, age, weight and height.

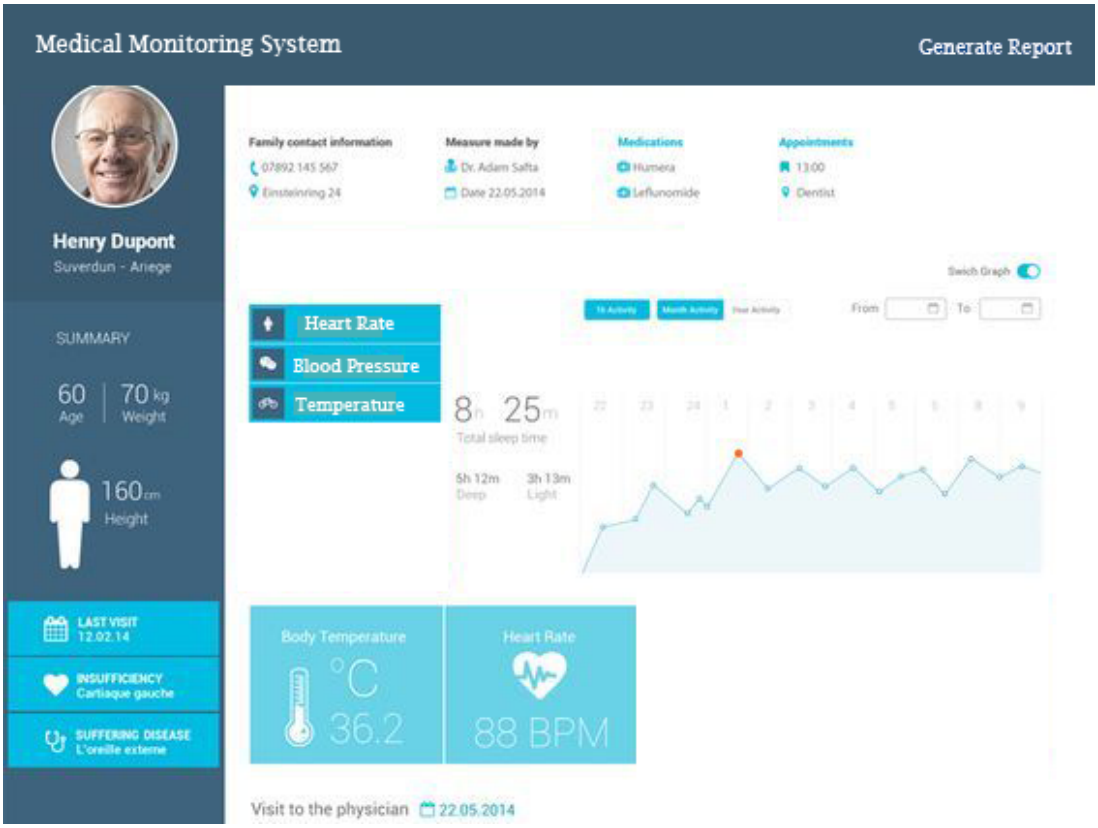

Figure 18: Web application UI

7.2 Hardware Interfaces

The system uses temperature, blood pressure and heart rate sensors which each provide their own hardware interface. Each of these sensors is connected to and mounted on the vitals machine.

The web application will make use of patient's vital information obtained directly from the vitals machine connected to the patient. The vital machine sends data to the hospital server via appropriate APIs, and displays the vitals in real-time on the vital monitor.

Figure 19: Vital Monitor

7.3 Software Interfaces

For the front-end components, HTML 5, CSS 3 and JavaScript version 2018 will be used. For the back-end components, Java Spring Boot framework will be used, including a MySQL database for the hospital's database. The front-end and back-end components communicate with each other over HTTP. The application will also use vital information from API services so that doctors can monitor patient's data on web application. Most of the data that will be shared across software components will be in the JSON format and will be pertaining to properties.

7.4 Communications Interfaces

The application will use Calling and SMS services to notify doctors and caregivers, which can be accomplished by the third-party Twilio notification services. The application relies on web browsers such as Google Chrome, IE and Mozilla Firefox to run. In order to ensure protection of patients' privacy, the application encrypts data prior to saving it in the database or transmitting it between the front-end and back-end components.

8.0 Non-Functional Requirements/Quality Requirements

Since this is an application designed to be used by primarily healthcare professionals (doctors, nurses, and professional caregivers), the UI design and overall flow of use must be clean and unobtrusive. The application must be responsive, focusing on fast and smooth interactions when generating reports and storing data rather than aesthetic appeasement of the application.

Data in the generated reports must be clearly legible and must utilize the maximum allocated screen space. All graphs displayed must be numerically scaled to avoid misunderstandings when reading reports. All reports must contain the most recent numerical representation of the vital signs data (temperature, blood pressure, and heart rate). Emergency alerts must be relayed as fast as possible to hospital personnel's communication devices (smartphones and/or pagers) and the information displayed must be concise, while containing all the essential information needed to identify abnormal vital signs and location of the patient. Reminders sent to the caregiver's communication devices for the administration of medicine must also be concise, while containing information about the drug: name of the drug, the amount of the drug, and any other important conditions to consider before drug administration and to look for after drug administration. Reminders also must contain the patient's information (name and location) as he/she may not be the only patient under the caregiver's care.

Security of data transfers during setup, vital signs report generation, aggregated vital signs data logging, emergency notifications, and medicinal administration reminders are of utmost importance as it pertains to physician-patient privileges. Hospital personnel must be securely logged in to the system in order to set up sensors and request reports. Hospital personnel's contact information (phone number and/or pager number) must be securely logged into the database as part of the patient's file. Emergency vital signs notifications must only be sent to the information on file. Caregiver's phone number must also be securely logged in the same patient's file, and medicinal administration reminders are to be sent to that number only.

9.0 Prioritization and Release Plan

9.1 Functional Requirement Prioritization Method

In order to prioritize the functional requirements of this Medical Monitoring system, a simple stakeholders' voting approach method is used, where the priority of each use case is defined as either high, medium or low based on its business importance. This approach is also combined with consideration of the mutual dependencies of each use case, as many of the use cases described in Section 4 have preconditions that depend on the working functionality of other use cases.

9.2 Feature Priorities

The priority of each functional requirement is listed in Table 1.

Table 2: Functional Requirement Priority

Use Case	Priority Level	Release Date	Remarks
1 - Patient Registration	High	April 30, 2020	The features presented in the patient registration use case is a high priority as it provides the initial set up and patient registration. Without this feature access to the database would not be possible and thus, storing information, which is required for features in the Vitals Report Generation use case, would not be possible.
2 - Vital Measurements Aggregation	High	March 31, 2020	The vital measurements aggregation is a core use case of this system. All users except for the caregiver receive information generated by this use case. Features in this use case must be tested extensively for reliability, data accuracy, and security. The emergency notification feature must also be tested extensively for false positives and false negatives.
3 - Medicinal Notification	Medium	May 20, 2020	Although the Medicinal Notification use case contains the core feature of the system to remind caregivers to administer medicine to the patients, it is a use case independent to Vital Measurements Aggregation, which is the most important use case. Temporary workarounds such as written down schedules and manually set up reminders can be used until a working use case has been developed.
4 - Vitals Report Generation	Medium	June 20, 2020	The Vitals Report Generation use case is a medium priority use case as it is possible to have the system function without it. It is an important system use case as it is a functional requirement for hospital personnel to be able to actively monitor patients remotely

			(without visiting the patient's room). However, a temporary workaround to have scheduled visits to the patient's room to actively monitor patient vital signs can still be used and would not cause significant safety issues as emergency abnormalities in the patient's vital signs are still reported directly to hospital personnel communication devices through a feature in the Vitals Measurement Aggregation use case.

Appendix A: Conceptual Model - Example Attribute Definitions:

Table 3: Conceptual Model - Example Attribute Definitions

Class Name	Example Attribute	Definition
<<entity>> Patient	GeneralInformation	FirstName: String LastName: String Address: String Age: int Sex: char DOB (date of birth): Date HealthCardID: String
	MedicalHistory	MajorOperations: String [] Allergies: String [] HistoricalConditions: String [] CurrentDrugUse: String []
	CurrentDiagnosis	ConditionName: String ConditionDescription: String
	AttendingPersonnel []	PersonnelName: String PersonnelContact: String
	CurrentLocation	RoomNumber: int BuildingNumber: String HospitalID: int HospitalName: String AdmittanceDate: Date
<<entity>> MedicinalSchedule	ScheduledDrug []	DrugName: String DrugAmount: String DrugSchedule: Calendar BeforeDrugWarnings: String AfterDrugObservations: String
<<entity>> VitalsReadings	TemperatureReading	Temperature: int ReadingsUnit: char ReadingsTime: Time ReadingsDate: Date
	HRReading	HeartRate: int ReadingsUnit: String ReadingsTime: Time ReadingsDate: Date

	BPReading	BloodPressure: int ReadingsUnit: String ReadingsTime: Time ReadingsDate: Date