Alert Generation System

This UML diagram represents a system for generating and managing alerts based on patient data. The classes handle the evaluation of patient data, creation of alerts, and the sending of alerts to the user. The central component is the AlertGenerator class, which evaluates patient data and triggers alerts based on predefined conditions, given by the AlertThreshold. It fetches historical patient data for more complex condition evaluations from the DataStorage class and retrieves patient details based on the patient ID from the Patient Identification System. It receives live data from the PatientMonitor, which also generates that data (PatientData) and sends it to the AlertGenerator. Each alert threshold contains the observed metric's name and interval. Whenever the threshold is met (AlertThreshold), the AlertGenerator, with multiplicity 1 to 0 or more (one AlertGenerator can generate 0 or more alerts), instantiates alerts, which are instances of the Alert class. The Alert class contains information such as the patient ID, condition, and timestamp. The alerts are then sent to the AlertManager class, which manages the process of sending the alerts to the staff. This is performed using the OutputStrategy interface, which implements different types of output strategies, such as ConsoleOutputStrategy and FileOutputStrategy. The whole AlertGenerationSystem consists of 9 components, however it does communicate with 3 other components, from the DataStorageSystem and PatientIdentificationSystem.

Data Storage System

The UML diagram represents a system for storing and managing historical patient data, which provides a foundation for real-time monitoring and retrospective analysis. The DataStorage class is the core of this diagram, responsible for storing, retrieving, and deleting patient data. It communicates with the PatientIdentificationSystem, by validating patient identities, from the PatientIdentifier class, that is in order to associate incoming data with the correct patient records, before any data retrieval or storage operations are performed. DataStorage stores the incoming information from the PatientData class, that being the ID of the patient (patientId), the map that contains the metrics' names and values (metrics) and the timestamp in a date-time format. The PatientData associates with the DataRetriever class, as the patients' details are needed for the data retrieval. Whenever the data needs to be retrieved, the DataRetriever component fetches patient data from the DataStorage, requiring both a staff ID and a patient ID to proceed. This ensures that only authorized personnel can access sensitive patient information. The authentication for the medical personnel is performed by the Staff class, which verifies access to patient data by checking the staff's credentials, such as their ID (staffId) and password (staffPassword). This ensures that patient data is only accessed by authorized personnel.

Patient Identification System

The PatientIdentificationSystem UML represents a system designed for matching incoming patient data with existing patient records. The main component is the PatientIdentifier class, which ensures that all data is accurately attributed to the correct patient. It does that by retrieving data from the PatientRecord, containing information such as: the ID of the patient (patientId), the name of the patient (name), the date of birth (dateOfBirth), and the medical history of the patient (medicalHistory). After fetching the data, the IdentityManager makes sure that integrity is maintained, by handling discrepancies or anomalies based one the patient's ID. Overall, this system communicates with other systems by interfacing the DataStorageSystem, which can be observed in the 2nd UML diagram, and by allowing the AlertGenerationSystem to fetch data from it, which is represented in the 1st UML diagram.

Data Access Layer

This UML diagram illustrates a data access layer, which is designed to make a connection between the hospital's database and external data sources, such as the PatientMonitor. The connection is made though the DataListener interface, which implements three types of data listeners, these being the TCPDataListener, the WebSocketDataListener, and the FileDataListener. This ensures that the system can retrieve the data from the monitors, no matter its type. After the retrieval, the data is sent to the DataParser, which ensures the conversion of the data given by the PatientMonitor, in order to allow its usage in the hospital's database. Once the data is converted into PatientData type, it is processed by the DataSourceAdapter class, which makes it possible to efficiently store that data into the DataStorageSystem.

UML Sequence Diagram based on AlertGenerationSystem

The given UML sequence diagram is based on the AlertGenerationSystem and contains five main objects: AlertGenerator, DataStorage, AlertManager, OutputStrategy and Alert; and an agent, that is the Doctor. The AlertGenerator communicates with the DataStorage to get the patients' historic data, based on their ID (getHistoricData(patientId)). After fetching the data, the AlertGenerator verifies whether the alert threshold criteria are met or not, if yes it sends the generated alert (instance of Alert) to the AlertManager, also specifying the output strategy. The AlertManager, based on the received data, manages the alert and sends it to the corresponding output strategy, specifying the following parameters: patientId, timestamp, label and data. The OutputStrategy displays the alert to the staff in the corresponding format, and the staff, in this case the doctor, acknowledges the given alert.

UML State Diragram – AlertLifecycle

AlertLifecycle is the UML state diagram that represents the states which an alert undergoes until it reaches its end, in other words, until its resolved. Starting the diagram, there is no alert until the data meets the predefined threshold criteria, that is the cause of the 'Alert Generated' state, when the alert is generated based on the data. Once the alert is generated, a notification is sent to the hospital's staff members, and the alert's state updates to 'Alert Sent'. Upon the alert's receival, it is acknowledged by the personnel and its state becomes 'Alert Acknowledged', meaning that the staff saw the alert. There are two different ways to resolve an alert, that's why the 'main' branch of the state diagram splits into two different actions, one being when the alert is resolved automatically, that is when the subsequent patient data is showing normal readings, or manually, when the staff must resolve it. After any of these two actions, the state of the alert becomes ‘Alert Resolved’, which depicts that the alert has been resolved and that there’s no need for that specific alert, therefore the diagram comes to an end.