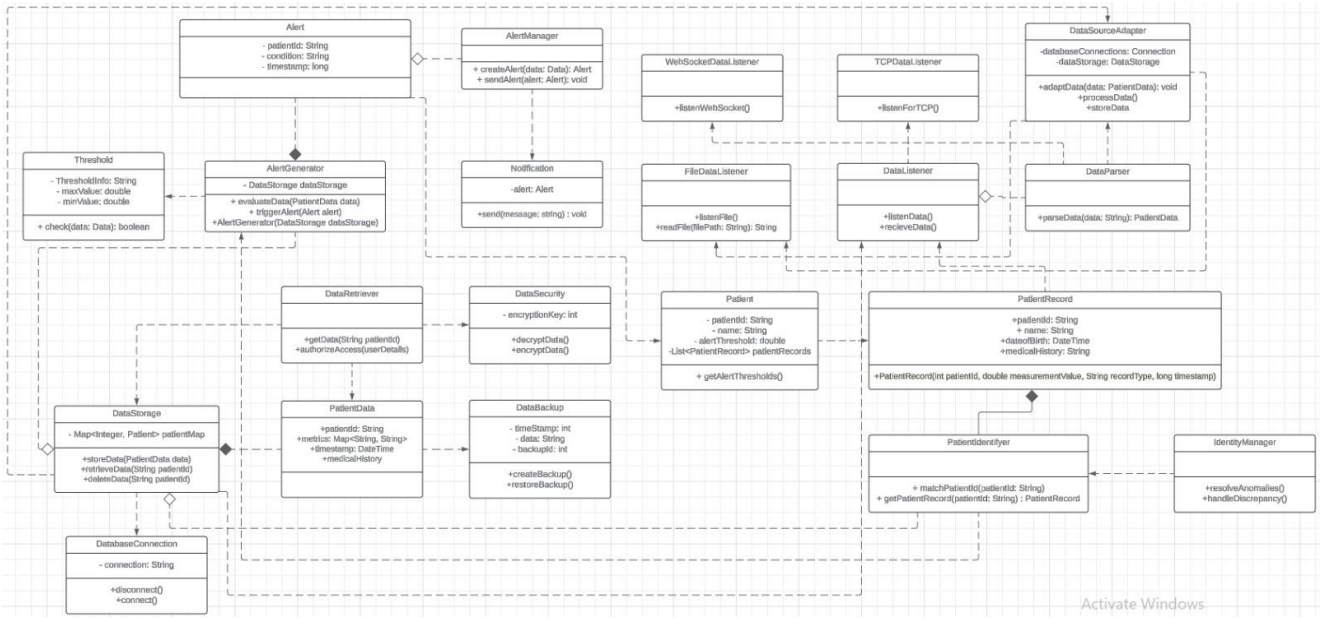
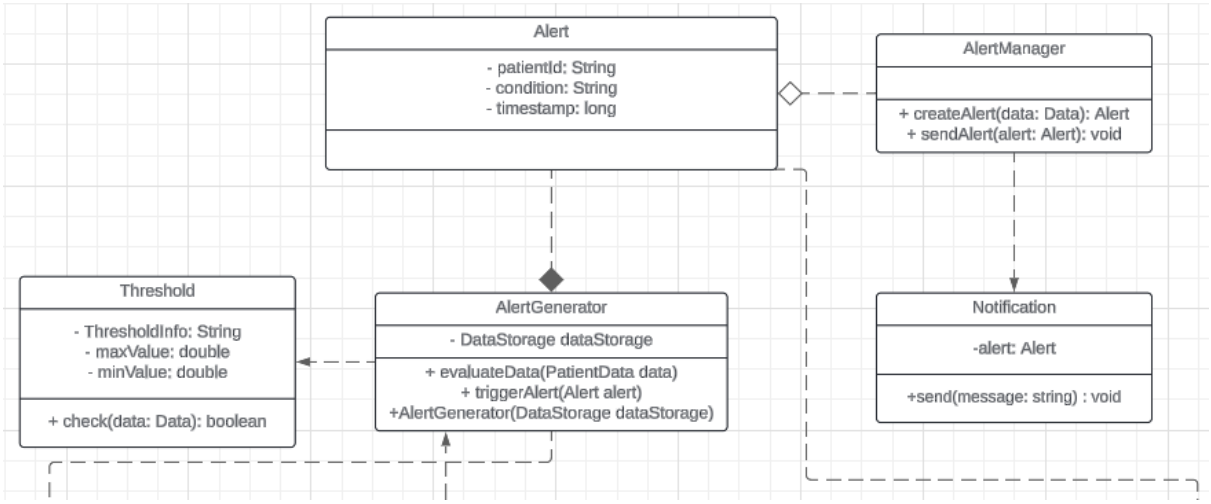


1. Alert Generator System



In this segment we have the following classes:

- Alert
- AlertGenerator
- AlertManager
- Notification
- Threshold

Objective: To monitor patient data in real-time and generate alerts for specific conditions that indicate a patient is in distress or requires immediate medical attention.

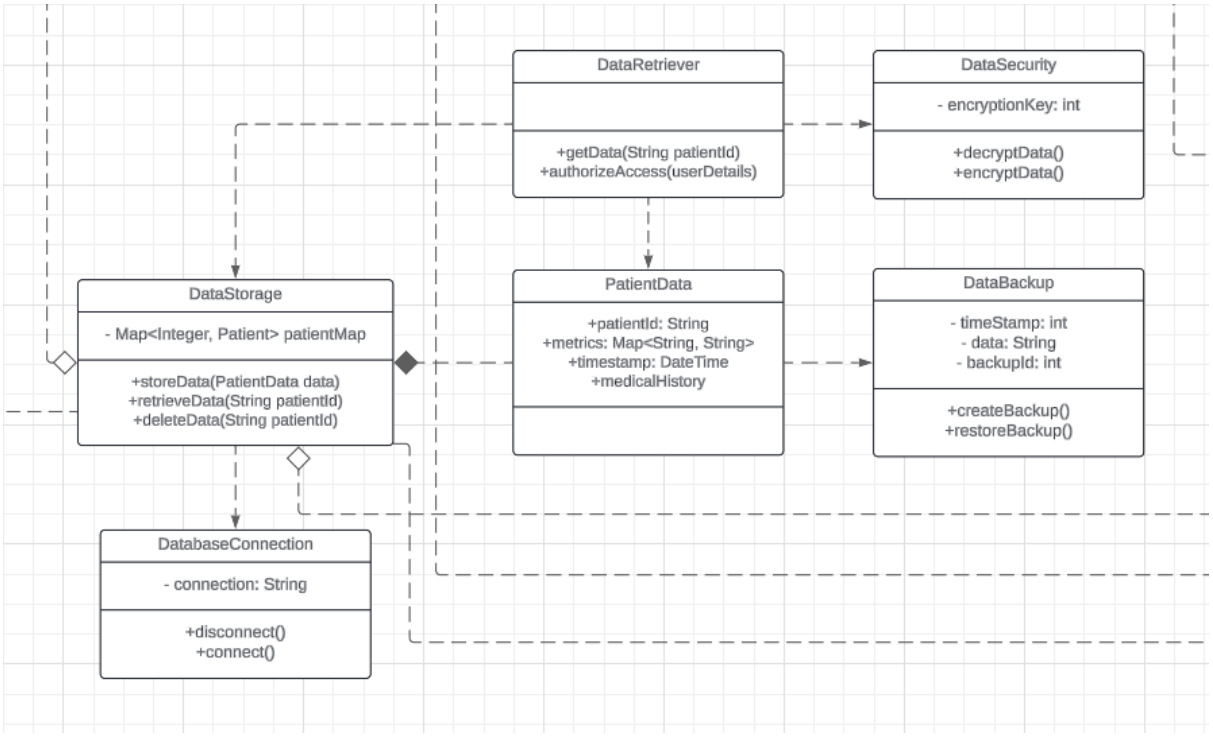
- The Alert class is in charge of handling the patient's information such as their ID, their condition and the timestamp of when they were checked into the hospital. This class acts as a fundamental data structure for this system.

- AlertGenerator class is in charge of generating alerts based on the data provided by the Alert class. It then creates specific alerts to fit that specific patients' needs such as their medical condition to ensure that the alerts are accurate and relevant.
- AlertManager is the class which is in charge of creating and overseeing the process of determining when an alert should be sent out. This also includes prioritizing certain alerts and making sure they are sent out properly.
- Threshold class is a very important part of this system as it “monitors” patient health. This is done by checking the patients medical records and “comparing” their current data to find differences or significant deviations (critical conditions) that would need an alert.
- Notification is in charge of actually sending the alert with a message attached to make sure it reaches the necessary staff.

- Patient class (which isn’t shown here but is still part of this system) is responsible for providing the relevant patient information and retrieving any alerts which are related to the patient. This class serves as the base repository for a patient's personal/medical information thus ensuring that any operations involving patient data are accurate and consistent.

(274 words)

2. Data Storage System



In this section we have the following classes:

- DatabaseConnection
- DataStorage
- PatientData

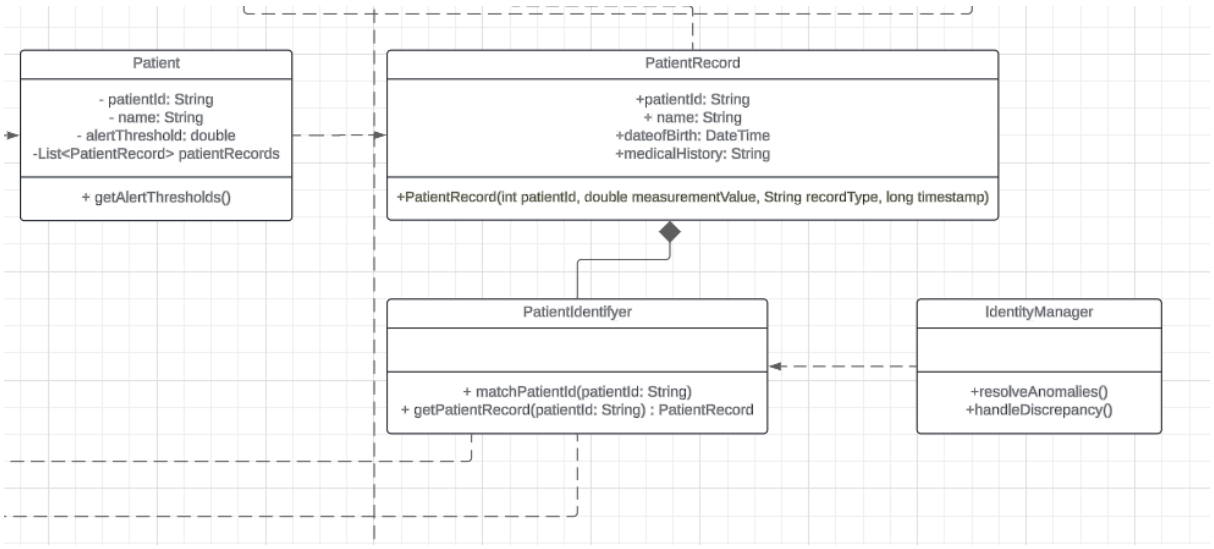
- DataBackup
- DataRetriever
- DataSecurity

Objective: To store and manage historical patient data, providing a foundation for both real-time monitoring and retrospective analysis.

- DatabaseConnection is the class that is responsible to set up and maintaining a connection to the database. It acts as a relay station for all interactions within the database system.
- DataStorage is in charge of handling functionalities such as data retrieval, deletion and storage. It also initializes a Hashmap to store the retrieved data this organizes it in an way that makes it efficient and easier to manipulate.
- PatientData class is in charge of taking the patient data from the DataStorage class, it acts as an simplify to access patient information, making it readily available for other systems which need it.
- DataBackup is in charge of making sure that no data is lost this is done by creating backups of patient data. This ensures that all the information is safely stored and can be recovered in case of any failures/data loses.
- DataRetriever is in charge of managing and retrieving the data that controls access permissions. This ensures that only an authorized person is able to access private information of the patient data. maintaining strict access control.
- DataSecurity is made to encrypt and decrypt the patient data, by applying robust methods of encryptions this helps protect data integrity and privacy. This ensures the patient information remains secure from being accessed by unwanted people.

(244 words)

3. Patient Identification System



In this section we have the following classes:

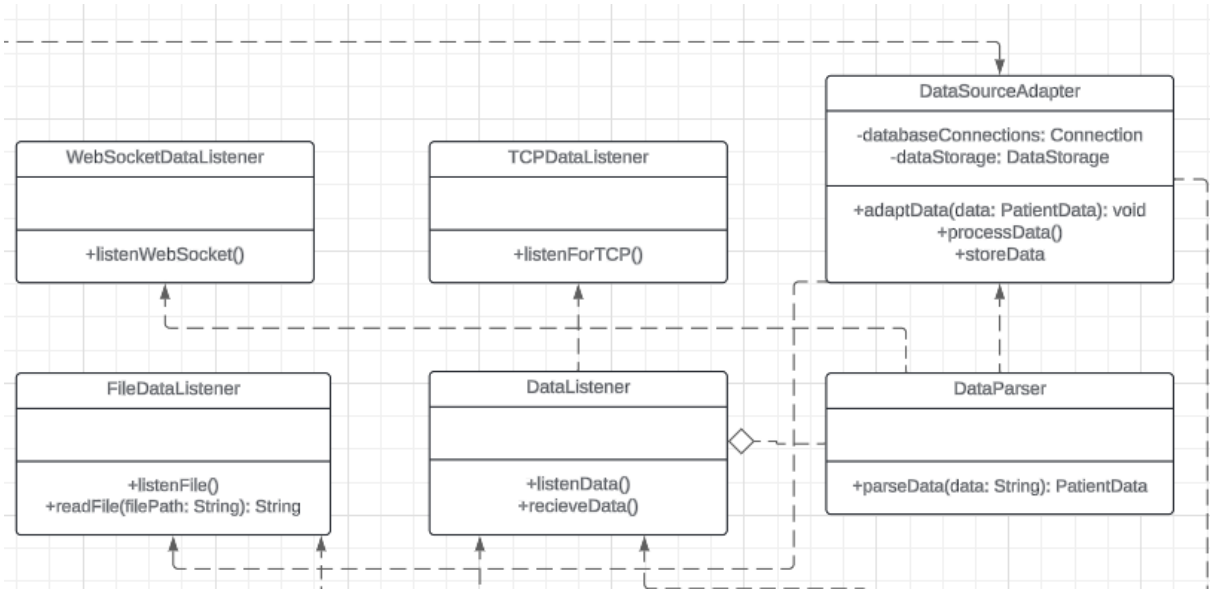
- Patient
- PatientRecord
- PatientIdentifyer
- IdentityManager

Objective: To match incoming patient data with existing patient records, ensuring that all data is accurately attributed to the correct patient.

- Patient class is responsible for providing the relevant patient information and retrieving any alerts which are related to the patient. This class serves as the base repository for a patient's personal/medical information thus ensuring that any operations involving patient data are accurate and consistent.
- PatientRecord is in charge for managing all the records of the patients handling things such as retrieval of all historical medical records and all current records to do with that patient. This class ensures that all patient records are organized and can be accessed efficiently.
- PatientIdentifyer is in charge of identifying the patient based on their ID and retrieving the corresponding records from the database. This class plays an important role to make sure that the right data is assigned to the correct patient by using identification methods to match records accurately.
- IdentifyingManager is responsible for error handling related to patient records and the identification process. It manages any issues or discrepancies that may arise during the identification process. Doing so ensures data integrity and any identification errors are dealt with.

(208 words)

4. Data Processing System



In this section we have the following classes:

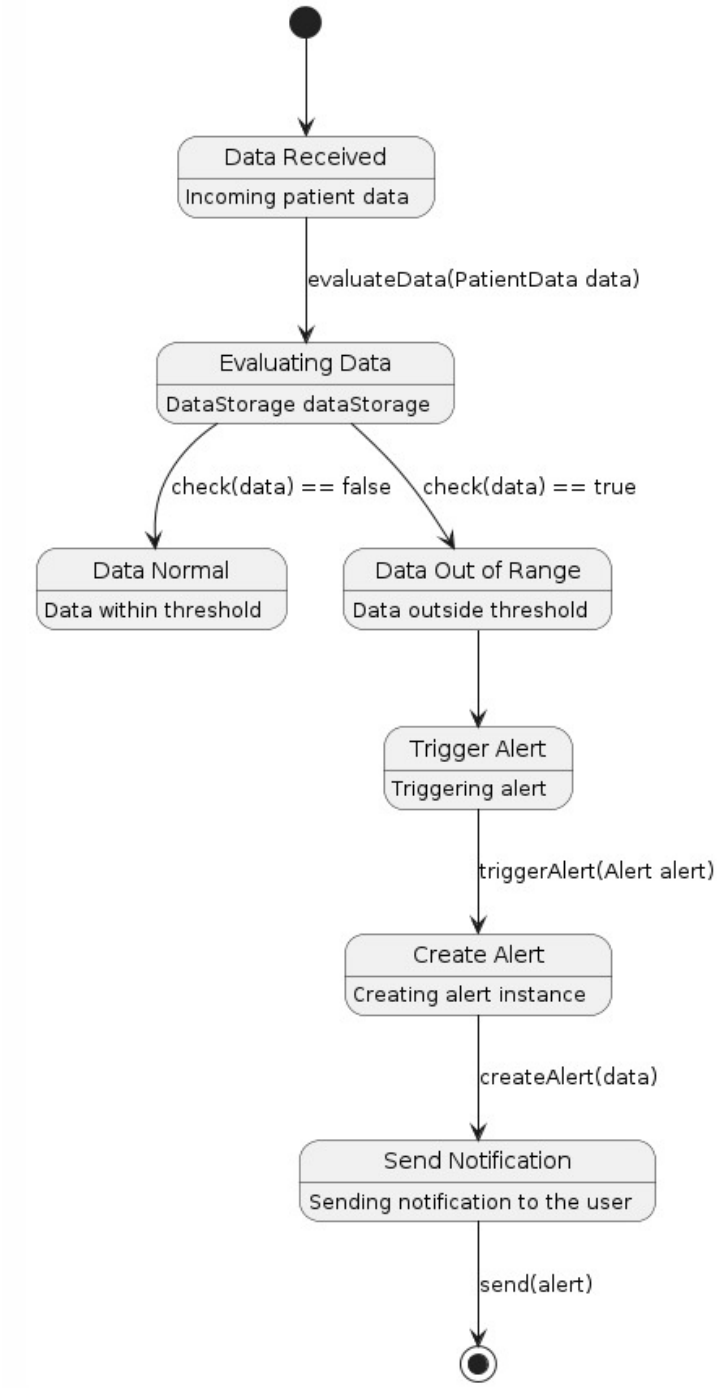
- WebSocketDataListener
- TCPDataListener
- DataSourceAdapter
- FileDataListener
- DataListener
- DataParser

Objective: To process the data retrieved from the database and process the data such that it can also be used.

- WebSocketDataListener is in charge of listening to data from a WebSocket connection. This class also enables the system to handle real-time data coming from WebSocket sources which ensures the incoming data is caught and processed accordingly.
- TCPDataListener is in charge of listening to data from a TCP connection. Similar to the WebSocketListener this class makes sure that the data transmitted over TCP is received and processed in a correct manner. This allows the system to integrate data from TCP-based sources.
- DataSourceAdapter is responsible for adapting, processing and storing patient data it also is responsible for managing database connections and handles data storage, making sure that all the processed data is correctly stored in the database and can be retrieved whenever they are needed.
- FileDataListener is in charge of listening to data from files and reading data from specified file paths. This class allows the system to process the data stored in the files, ensuring that file based data is used in the overall data processing workflow.
- DataListener is in charge of defining the basic methods for listening to and receiving data from various sources. This class provides a standard interface that all other listeners classes must implement, ensuring consistency in how the data is handled across all the different sources.
- DataParser is in charge of parsing raw string data into structured PatientData objects. This class converts raw data into a form that can be easily processed and stored making sure all the incoming data is structured and up to a certain standard.

(286 words)

1. State Diagram for Alert Generator system



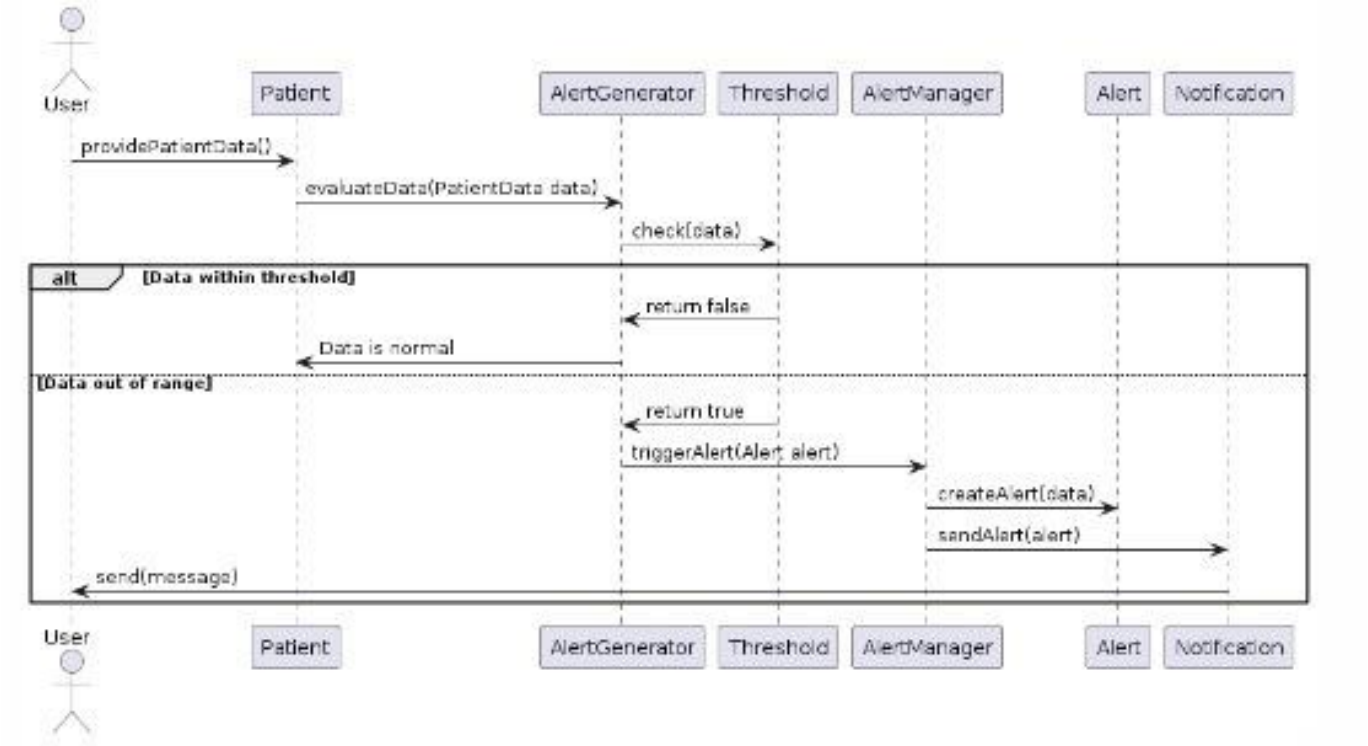
This state diagram shows the approach to monitoring patient data in real-time, efficiently identifying potential issues and ensuring timely alerts to maintain patient health and safety with medical intervention.

- Data received system begins with the receiving of incoming patient data. This data includes various health metrics that need to be monitored.
- The received data is then passed to the evaluateData function which uses the DataStorage class. This function then checks the data against the stored thresholds to determine the status.
    - Data Normal checks whether the check(data) returns false which shows that the patient data is within normal thresholds which means no further actions are required.
    - Data out of range checks whether the check(data) function return true which would mean the patients data is outside the acceptable threshold range and requires urgent care.

- Upon detecting data out of range the triggerAlert function triggers an alert which initiates the alerting process.
- The Create Alert system then continues to create an alert instance which involves using the createAlert function which compiles the necessary information to form an alert based on the out-of-range data.
- The Send Notification function alert is then sent to the user or medical worker using the send function. This ensures that the notification reaches the appropriate user properly and in a timely manner.

(213 words)

1. Sequence Diagram for Alert Generator System



This sequence diagram shows the sequence of operations within the alert generation system and all the interactions between the different components of the system and explains the approach taken to monitor patient data, evaluate it against certain thresholds and make sure timely alerts are created and sent out.

- The process initially begins with the user inputting/providing the patient data using the providePatientData() method in the Patient class.
- The Patient class then calls the evaluateData(PatientData data) method on the AlertGenerator class to start the data evaluation, the AlertGenerator class is responsible for analyzing the patient data.
- The AlertGenerator then contacts the Threshold class by invoking the check(data) method, which then the Threshold class compares the incoming patient data against predefined threshold to check whether or not the data is in within normal limits.
- If the check(data) method returns false this shows that the patients data is within the threshold this is then sent back to the AlertGenerator class and no further action is needed or taken showing that the patients condition is stable.

- Otherwise if the check(data) method returns true this indicates that the patient's data is out of the threshold range and the process to trigger an alert is started by the AlertGenerator which calls the triggerAlert(Alert alert) method on the AlertManager class.
- The AlertManager then goes on to create a new alert instance using the createAlert(data) method. This alert will contain the relevant informationabout the patient's condition and the specific data points that triggered the alert.
- Then to finish it off the AlertManager send out the alert to the Notification class using the send(Alert) method. The Notification class is the one responsible for actually delivering the alert message to the appropriate users like medical staff to allow intervention in time.

(291 words)