1. Alert Generation System
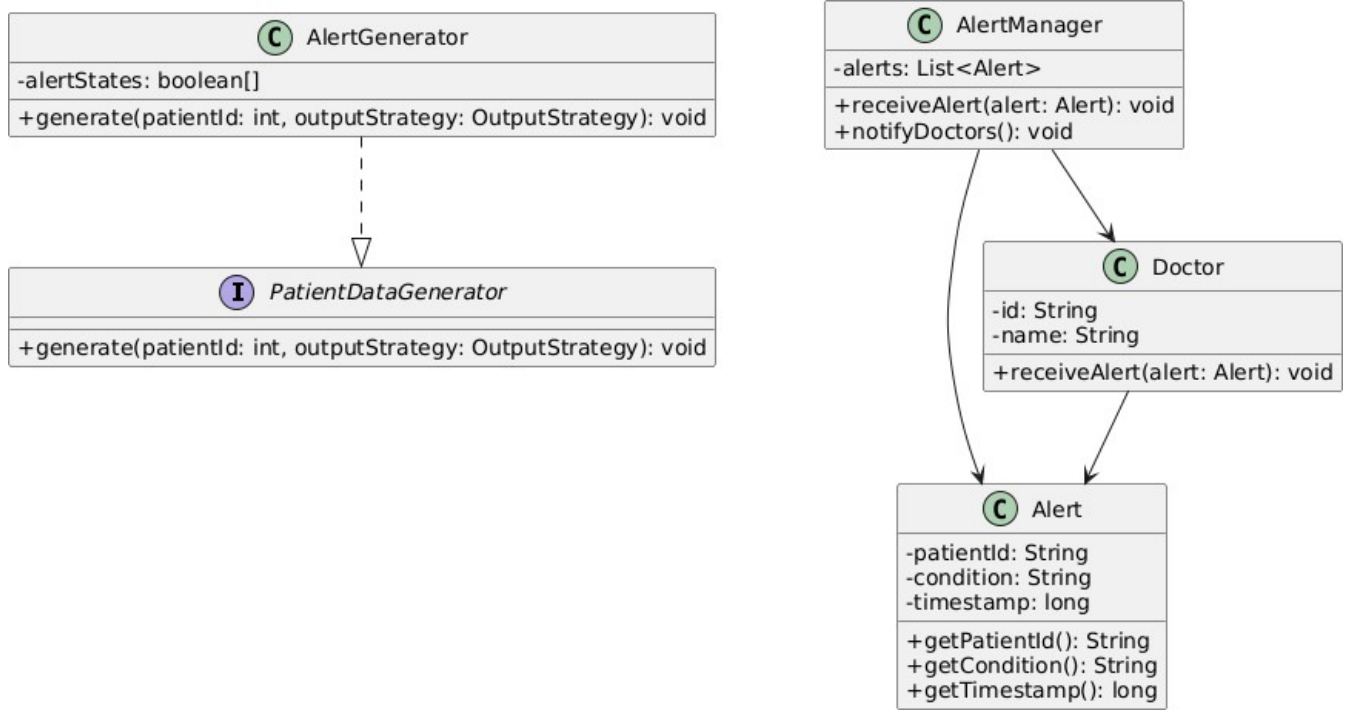

The **Alert Generation System** monitors patient data and triggers alerts based on specific health conditions. It consists of four key components: the Alert Generator, the Alert Manager, the Doctor, and the Alert. **AlertGenerator**: This class generates alerts for patients based on their health data. If a patient already has an active alert, there's a 90% chance it will be resolved; if not, a small chance exists for a new alert to be triggered. **Alert**: The Alert object contains the patient's ID, the condition causing the alert (e.g., "Critical Blood Pressure"), and the timestamp when the alert was triggered. **AlertManager:** Responsible for managing and dispatching alerts, the AlertManager forwards each alert to the Doctor. It ensures all alerts are handled efficiently. **Doctor**: The Doctor receives alerts and takes necessary actions based on the patient's condition. They are responsible for responding to alerts and ensuring patient care. This system simulates real-time patient monitoring, where the AlertGenerator triggers alerts, the AlertManager oversees them, and the doctor responds, ensuring timely medical intervention for patients in need.

2. Data Storage System
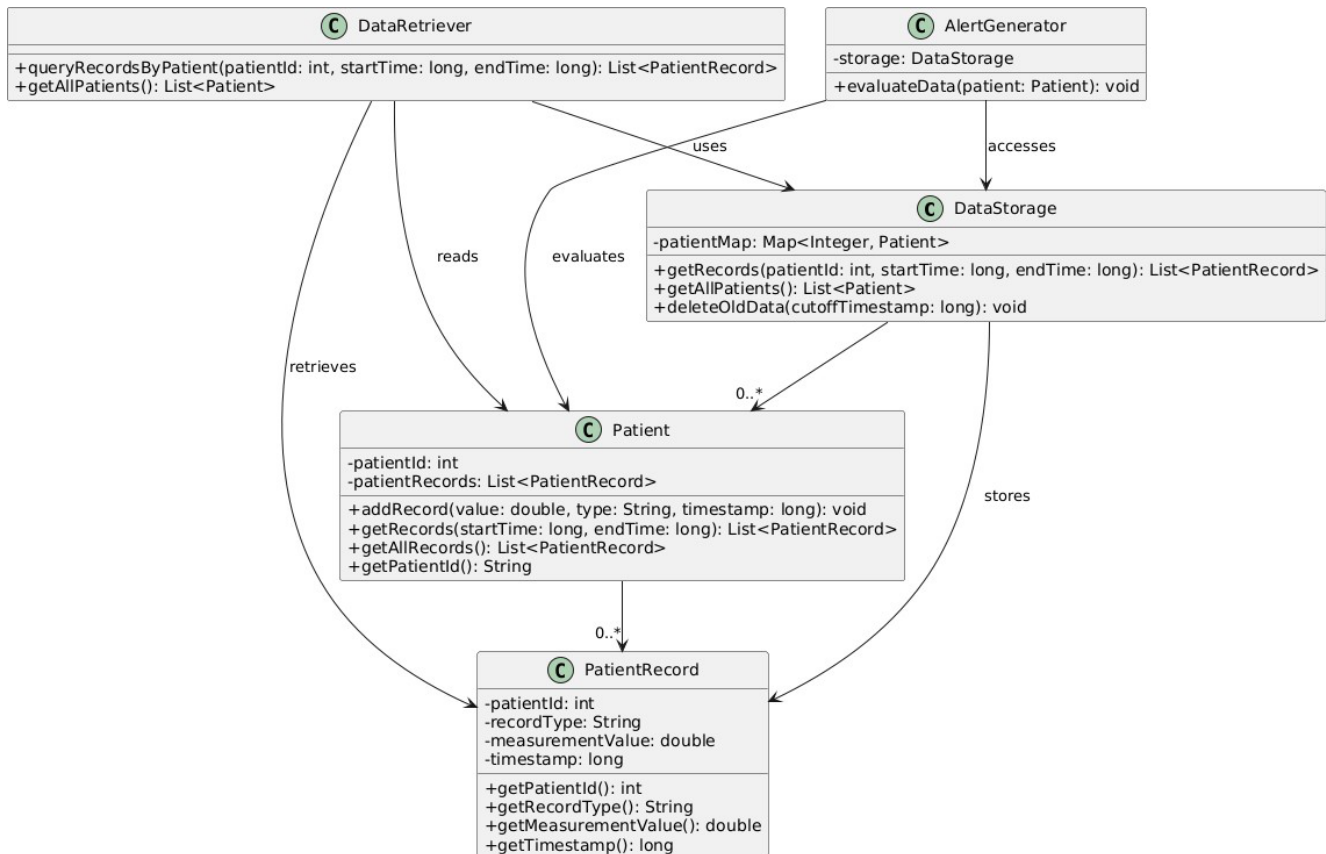
The **Data Storage System** is responsible for managing all patient data, including important health information like heart rate and blood pressure, along with timestamps. This makes it easy for doctors to track patients in real-time and look at their past records to spot any changes or trends. The **Alert Generator** works with this data to check for any problems, like low oxygen levels or high blood pressure. If something is wrong, it sends out alerts so that medical staff can take action quickly. The system also keeps things organized by deleting old data after a certain period. Only authorized people can access or change the patient records, which ensures the data stays safe and private. Each **Patient Record** contains the patient's ID, the type of data, and the time it was taken, making it easy to track and manage all the information efficiently. This system helps doctors make informed decisions while keeping everything secure and up-to-date.

3. Patient Identification System
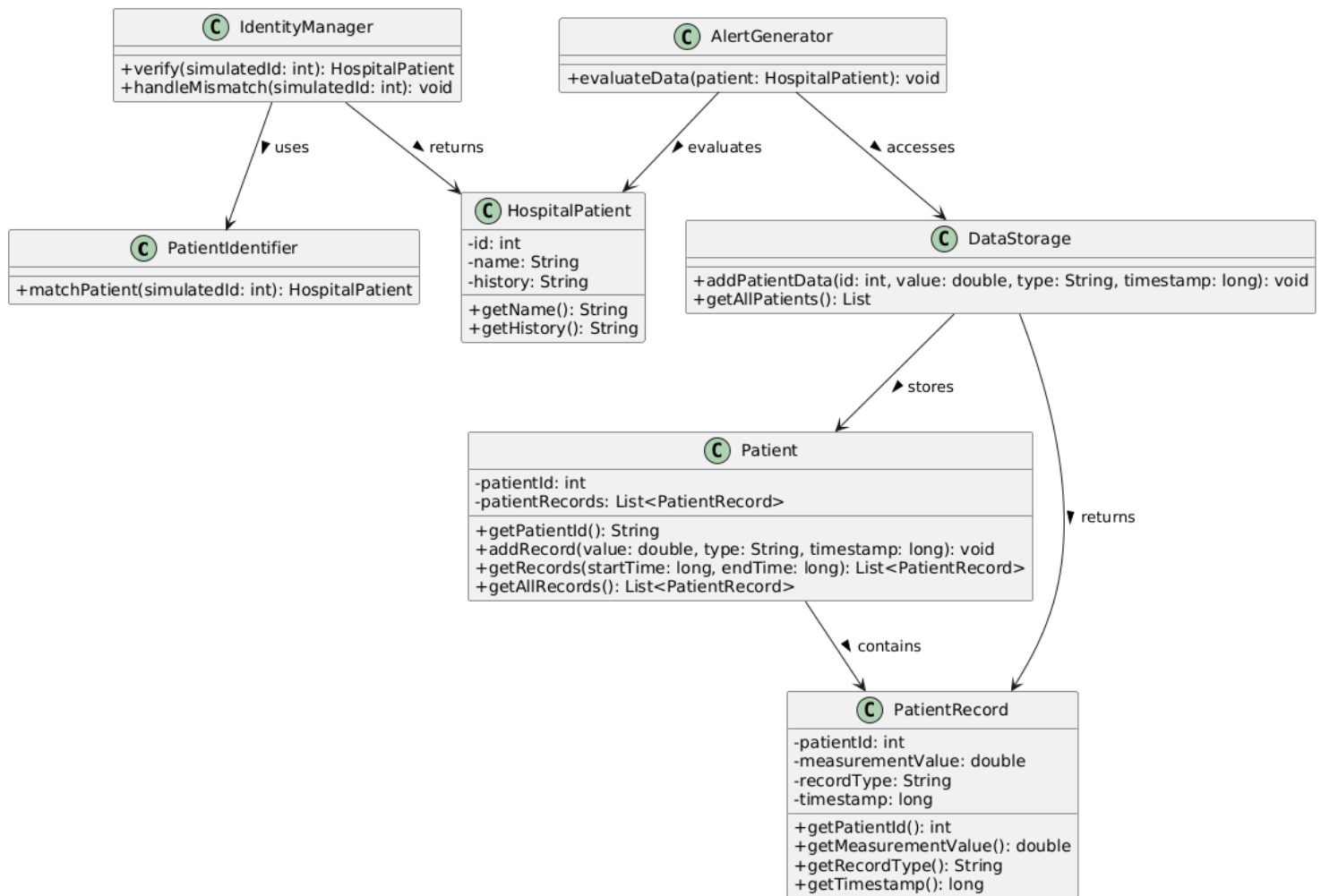
The Patient Identification System is a key part of the hospital software, making sure incoming data gets linked to the right person. When a patient arrives **(HospitalPatient),** their identity is verified through the **IdentityManager, which uses the PatientIdentifier** to try to match them with existing records. This is really important because every patient has their own medical history—things like blood type, allergies, and past conditions—which are stored in the **Patient class as a list of PatientRecord objects**.

If the **IdentityManager** doesn't find a match, the AlertGenerator has to handle the case, possibly triggering the creation of a new patient entry so no data is lost. Once a patient is verified, the system can safely store and access their health data through DataStorage, and the AlertGenerator can evaluate their condition based on all the collected information.

4. Data Access Layer

The Data Access Layer connects the outside world to the internal healthcare system. It allows patient data to flow in from different sources—TCP, WebSocket, and File-based inputs. Each of these has its own listener class (**like TcpDataListener, WebSocketDataListener, and FileDataListener**), but they all follow a shared interface so the system stays flexible and organized. This layer listens for incoming data from a signal generator, processes it, and passes it along in a format the rest of the system can use. For example, a TCP server checks for client connections and sends patient data through sockets, while the WebSocket server can broadcast data live to multiple users. File-based input stores the data as logs for later analysis. Once the data is received, a DataParser converts it from raw formats (like CSV or JSON) into usable objects. The **DataSourceAdapter** then delivers this parsed data to the DataStorage system. **The Data Access** Layer keeps everything cleanly separated so that no matter how the data is sent, it can always be processed, stored, and accessed securely by doctors, patients, or other parts of the CHMS system.