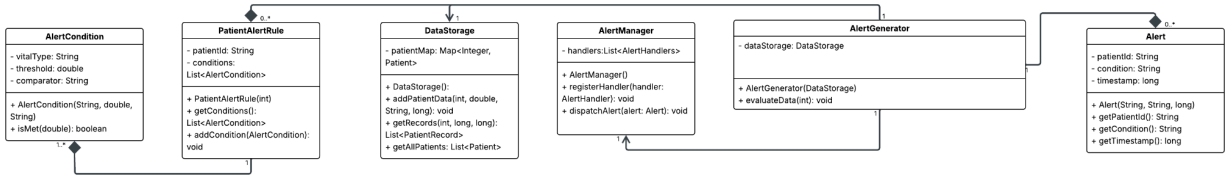
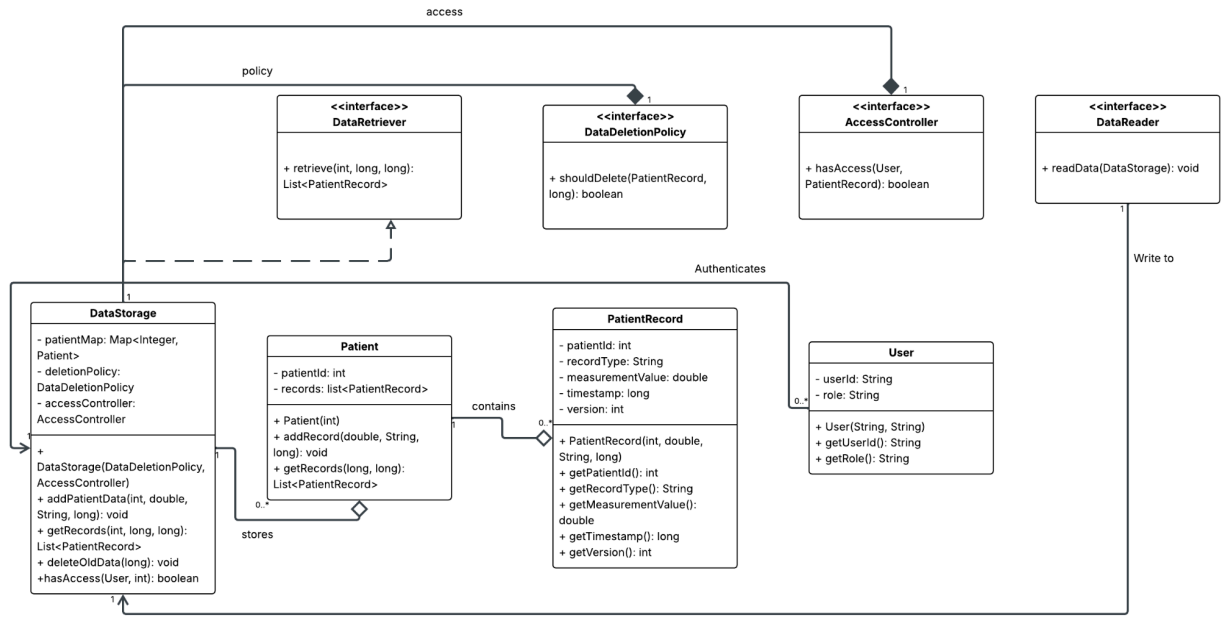


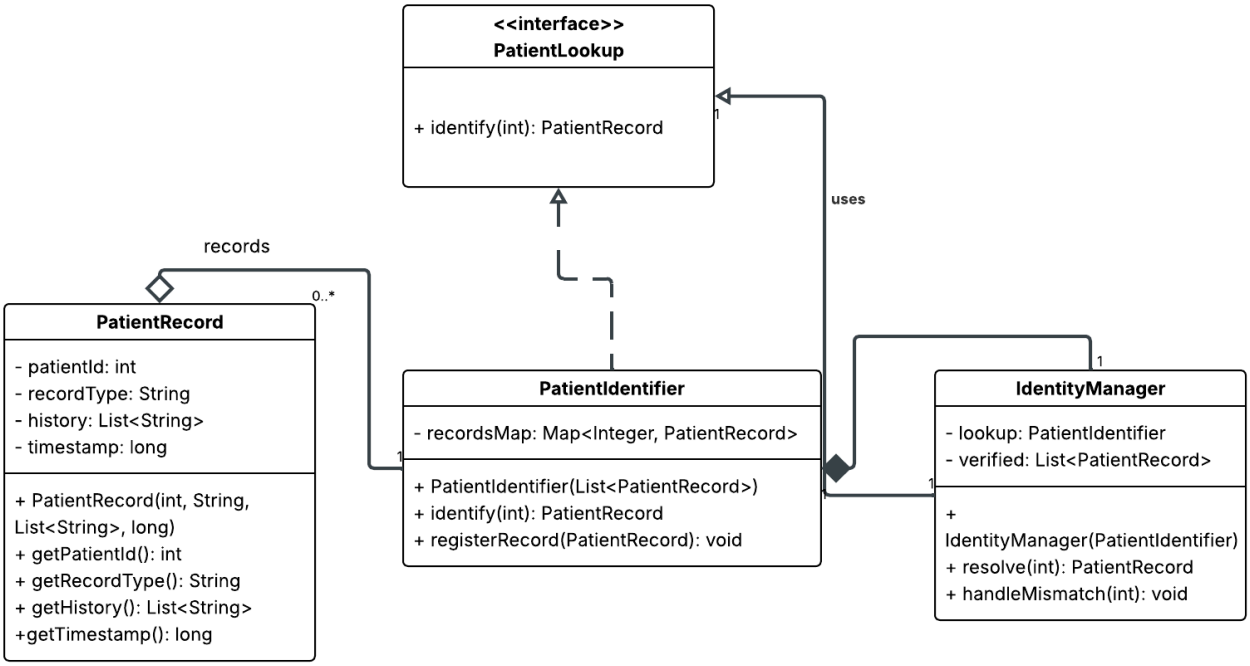
1. Alert Generation System

The core of this diagram is the AlertGenerator, which grabs medical data from DataStorage and looks up that patient's alert rule. When a condition is met, alertGenerator makes a new Alert object. There is also AlertManager which when it receives an alert sends an alert to all handlers. We have clear separation of storage, evaluation, alert creation and routing here.



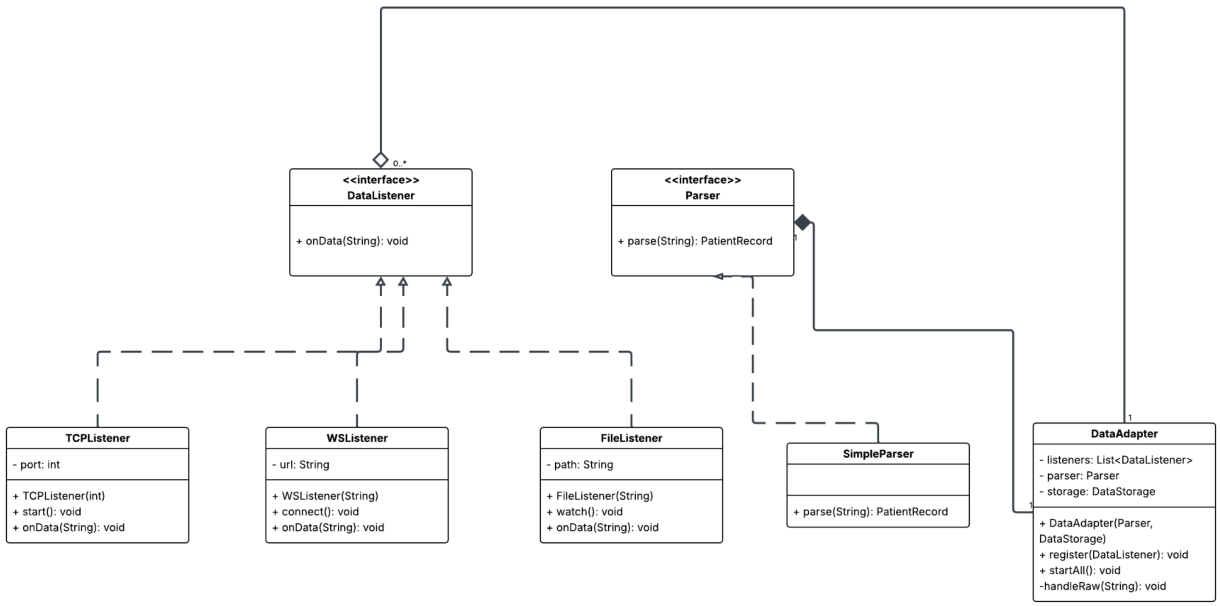
2. Data Storage System

Here everything revolves around DataStorage, this class holds patient objects, it's also linked to a deletion policy that gets rid of old records and Access Controller which controls who is allowed to access the records. The controller and deletion policy are tightly bound with DataStorage.
The user class is used to check if the user's role grants them access to the records.
DataReader has incoming data going through it and can be used to retrieve specific data. There are also aggregation arrows between DataStorage and Patient and Patient and PatientRecords, they are not tightly bound because not all patients are necessarily in DataStorage and the same goes for the records.



3. Patient Identification System

Here the patientLookup is an interface that can take an id and turn it into a record. We then have PatientRecord which holds all information about the patient. There is also the PatientIdentifier which holds a map with as key the id and value the record that corresponds to that key. The identify method will check if the id has a record. IdentityManager is used to manage the PatientIdentifier and runs handleMismatch if a error is thrown when running identify.



4. Data Access Layer

Here the most important part is the DataAdapter which takes a parser and a DataStorage instance, then listeners get connected to the adapter, this includes tcp websocket and file listeners. These listeners all implement the datalistener so that means they will all read data if there is data. When onData gets called, data gets sent to the parser through DataAdapter's handleRaw method and the parser turns it into a PatientRecord object.