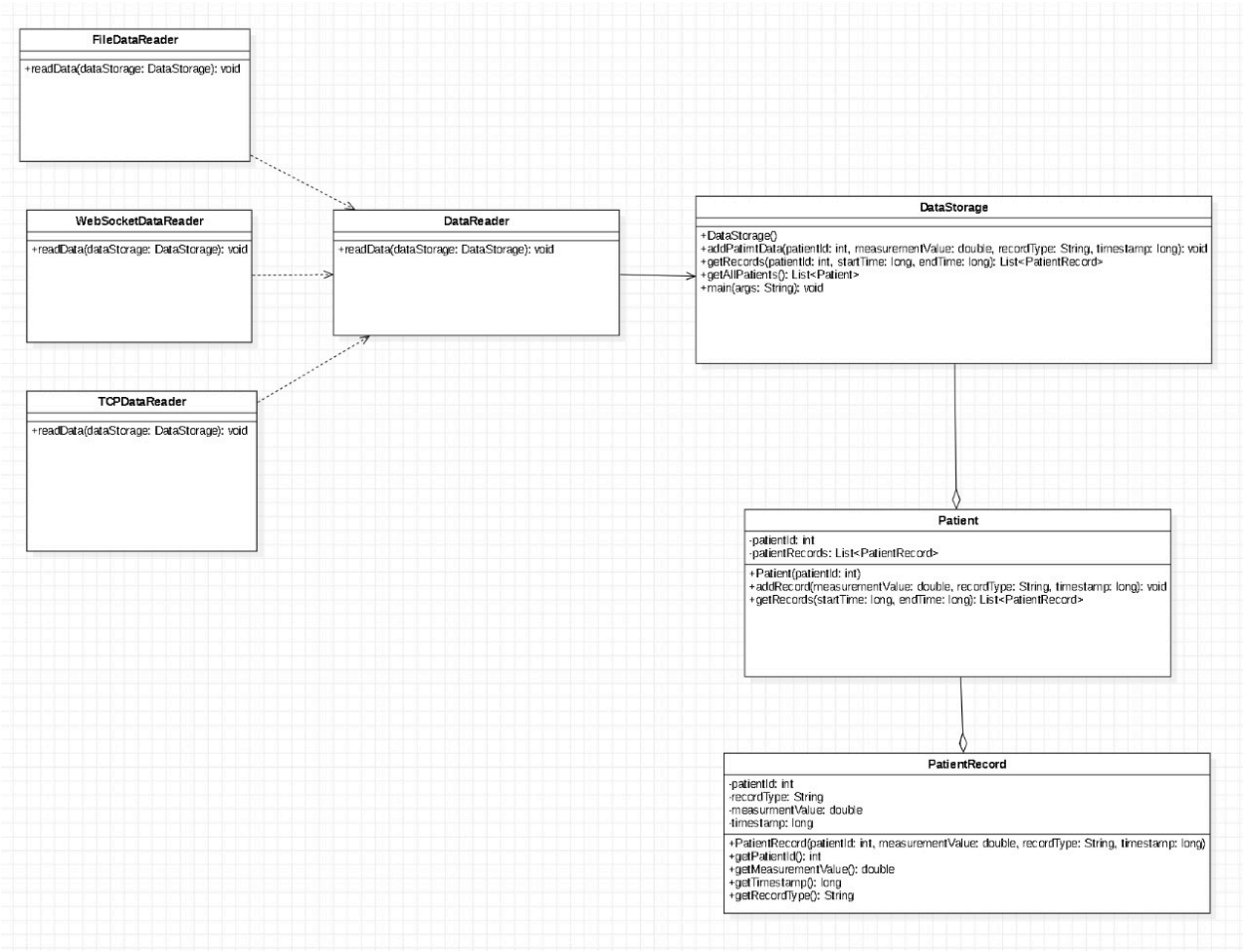

The UML diagram illustrates the structure and relationships within a system designed to read, store, and retrieve patient data records. The primary classes in the system include `DataReader`, `FileDataReader`, `WebSocketDataReader`, `TCPDataReader`, `DataStorage`, `Patient`, and `PatientRecord`.

The `DataReader` class serves as an abstract base interface, which is inherited by three subclasses: `FileDataReader`, `WebSocketDataReader`, and `TCPDataReader`. Each of these subclasses implements the `readData(dataStorage: DataStorage)` method, indicating that they can read data from different sources (file, WebSocket, and TCP, respectively) and store it in an instance of `DataStorage`.

The `DataStorage` class is responsible for managing patient data. It provides methods for adding patient data (`addPatientData(patientId: int, measurementValue: double, recordType: String, timestamp: long)`), retrieving records for a specific patient within a time range (`getRecords(patientId: int, startTime: long, endTime: long): List<PatientRecord>`), and retrieving all patients (`getAllPatients(): List<Patient>`). This class aggregates `Patient` objects, each of which holds multiple `PatientRecord` objects.

The `Patient` class represents an individual patient and includes an identifier (`patientId: int`) and a list of records (`patientRecords: List<PatientRecord>`). It provides methods to add records (`addRecord(measurementValue: double, recordType: String, timestamp: long)`) and retrieve records within a specified time range (`getRecords(startTime: long, endTime: long): List<PatientRecord>`).

The `PatientRecord` class encapsulates a single data record for a patient, including attributes such as `patientId`, `recordType`, `measurementValue`, and `timestamp`. It provides methods to retrieve each attribute.