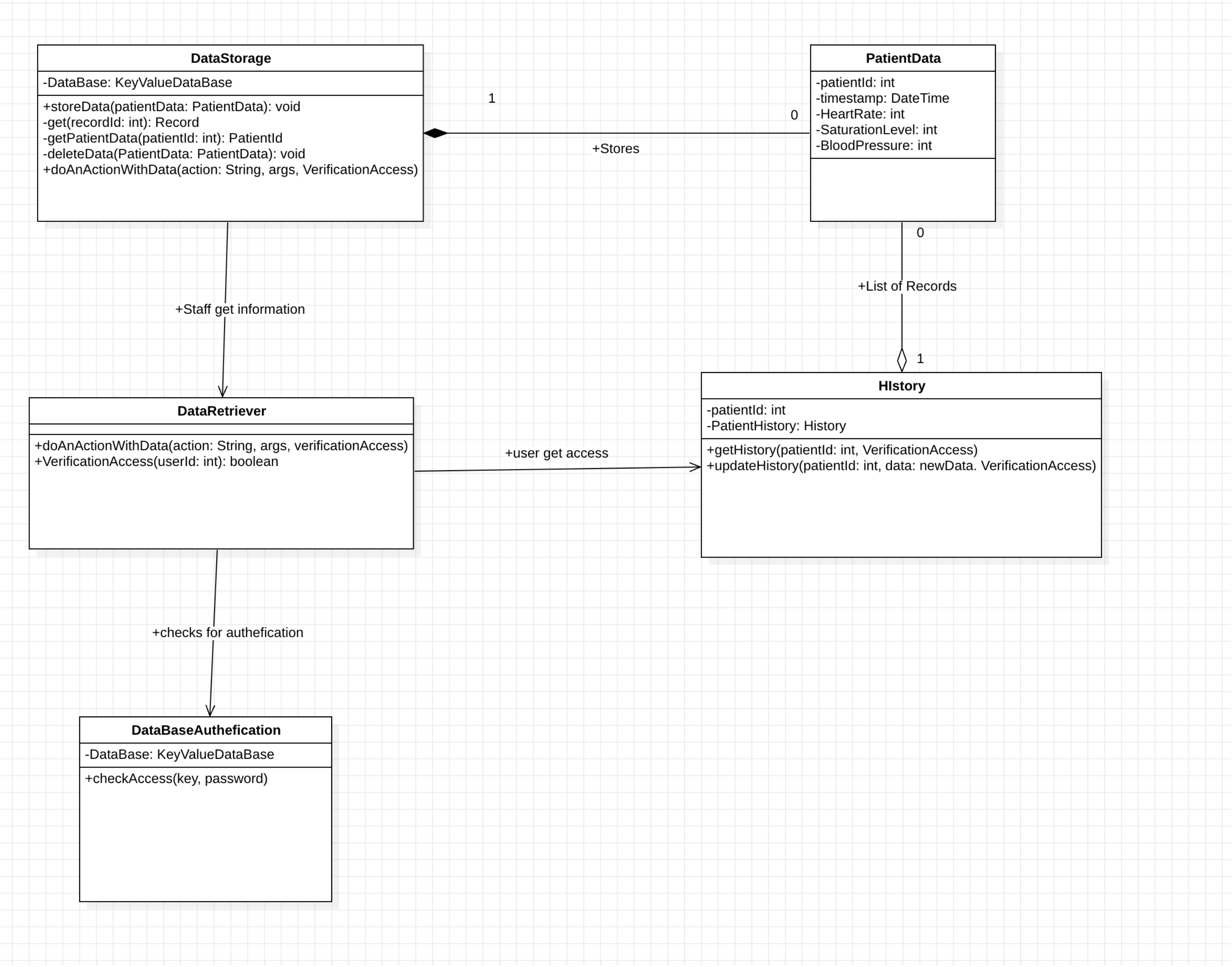
DataStorage
-DataBase: KeyValueDataBase
+storeData(patientData: PatientData): void
-get(recordId: int): Record
-getPatientData(patientId: int): PatientId
-deleteData(PatientData: PatientData): void
+doAnActionWithData(action: String, args, VerificationAccess)
1
0
+Stores
PatientData
-patientId: int
-timestamp: DateTime
-HeartRate: int
-SaturationLevel: int
-BloodPressure: int
0
+List of Records
1
+Staff get information
HIstory
-patientId: int
-PatientHistory: History
+getHistory(patientId: int, VerificationAccess)
+updateHistory(patientId: int, data: newData. VerificationAccess)
DataRetriever
+doAnActionWithData(action: String, args, verificationAccess)
+VerificationAccess(userId: int): boolean
+user get access
+checks for authefication
DataBaseAuthefication
-DataBase: KeyValueDataBase
+checkAccess(key, password)

This diagram models storage and access system:

1. DataStorage
   - Backed by a key-value database, it is responsible for persisting PatientData records and exposing CRUD methods (storeData(), get(), getPatientData(), deleteData()).
   - It aggregates zero or more PatientData instances (1 – 0..*), each of which carries a timestamp and vital signs.
2. PatientData & History
   - Each PatientData object (heart rate, saturation, blood pressure at a given time) is stored in DataStorage.
   - A separate History class maintains the full timeline for one patient (1 – 0..*), providing getHistory(patientId, verificationAccess) and updateHistory(patientId, data, verificationAccess).
3. DataRetriever
   - All operations (doAnActionWithData(...)) require a VerificationAccess token, which it first checks via DataBaseAuthentication.
   - Once authenticated, DataRetriever routes calls to either DataStorage or History as appropriate.
4. DataBaseAuthentication
   - Encapsulates credential checks against the same key-value database.

Flow:
A staff client calls DataRetriever.doAnActionWithData(...). Before executing, DataRetriever invokes DataBaseAuthentication.checkAccess(...). If verification succeeds, it forwards the request—either storing or fetching raw PatientData via DataStorage, or retrieving/updating the patient's timeline through History.