Stage 2: Database Design
Team075 - SQL_Squad

1) UML Diagram

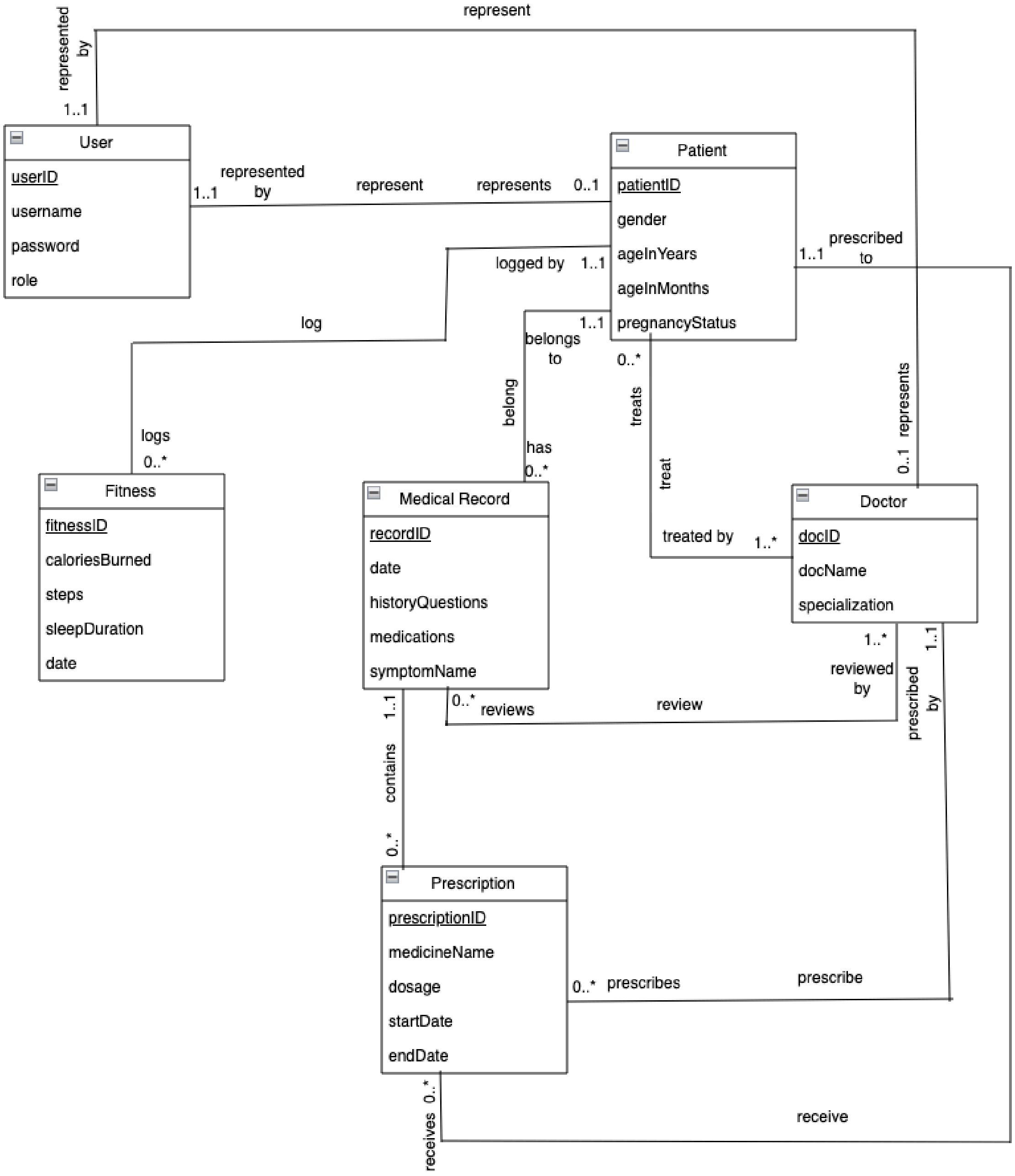

Figure 1: UML Diagram

2) Assumptions for each entity and relationship in the model

1. Patient Entity:

- **Assumption:** A patient must have personal information, including gender, age, and pregnancy status. Each patient is related to all the doctors they visit.
- **Reasoning:** The Patient is modeled as a separate entity because it encapsulates various attributes related to a patient's medical background and association with a doctor.
- **Cardinality:**
  - A patient has a 1-to-1 relationship with a User, meaning each patient corresponds to a unique user account in the system.
  - A patient has a many-to-many relationship with Doctor, meaning multiple patients can be assigned to the same doctor, and each patient can have one or many assigned doctors.
  - A patient has 0 or many medical history records(one-to-many)
  - A patient logs 0 or many fitness records(one-to-many)
  - A patient receives 0 or many symptoms(one-to-many)

2. Doctor Entity:

- **Assumption:** Doctors have identifying information like a name and a specialization. Specialization is relevant because patients often seek out doctors based on their specific medical needs.
- **Reasoning:** The Doctor entity captures the medical professionals associated with each patient and will help us extract the details of patients under the care of a doctor.
- **Cardinality:**
  - Each doctor is related to many Patient entities (one doctor may be assigned to multiple patients).
  - Each doctor reviews 0 or many patients and each medical history is reviewed by 1 or many doctors.
  - Each doctor reviews 0 or many medical records.
  - Each user entity pertains to 1 doctor (one-to-one).

3. User Entity:

- **Assumption:** A User entity represents the login information of any user in the system, whether a doctor or a fitness-tracking user (patient).
- **Reasoning:** We separate User from other entities because User information involves login credentials, which are distinct from a patient's medical information or fitness records. Users can be both doctors and patients (defined by role).
- **Cardinality:**
    - The system supports a 1-to-1 relationship between User and Patient (a patient corresponds to a unique user account).
    - The system supports a 1-to-1 relationship between User and Doctor(a doctor corresponds to a unique user account).

4. Medical Record Entity:

- **Assumption:** This entity stores detailed medical records, including past symptoms and medications.
- **Reasoning:** Medical Record is separated as an entity because it involves multiple fields related to the patient's medical problems that are not directly part of the Patient table. Medical Record is stored by date and a new record is created for every result of medical examination to keep the history available for doctor's future reference.
- **Cardinality:**
    - Each patient has many 'Medical Record'(the entity defined in our UML - expressed over here like this for clarity) records, while a 'Medical Record'(the entity defined in our UML) record belongs to only one patient (hence a 1-to-many relationship between User and Medical Record).
    - Each medical record contains 0 or many prescriptions, while each prescription can belong to only one medical record.
    - Each medical record is reviewed by 1 or many doctors (one-to-many).

5. Prescription Entity:

- **Assumption:** Prescriptions are tracked separately for each date they are issued for a patient.
- **Reasoning:** Prescriptions are modeled as a separate entity to track details, dosage, and validity for each user as they change over time. New record is created for every new prescription and is stored by date.

- **Cardinality:**
  - Each prescription is contained in a single medical record, and a medical record can have 0 or more prescriptions.
  - Each prescription record is associated with exactly one patient, and a patient can have 0 or more prescriptions.

6. Fitness Entity:

- **Assumption:** The fitness entity tracks physical activities like calories burned, steps taken, and sleep duration for each "patient" user on a daily basis.
- **Reasoning:** Fitness is separated as a distinct entity because its attributes focus on enabling patients to track their own fitness metrics.
- **Cardinality:**
  - There is a 1-to-many relationship between Patient and Fitness, where each patient can have multiple fitness records over time.
  - Each fitness record belongs to exactly one patient.

3) Normalization of the Database

Functional Dependencies for the UML Diagram:

- **Users:** userID → userName, password, role
- **Patients:** patientID → gender, ageInYears, ageInMonths, pregnancyStatus
- **Doctors:** docID → docName, specialization
- **Medical Record:** recordID → date, historyQuestions, medications, symptomName
- **Prescriptions:** prescriptionID → medicineName, dosage, startDate, endDate
- **Fitness:** fitnessID → caloriesBurned, steps, sleepDuration, date

Left-Middle-Right Table for Normalization (BCNF)

Left Side	Middle	Right Side	None
userID	None	userName, password, role	
patientID	None	gender, ageInYears,	

		ageInMonths, pregnancyStatus	
docID	None	docName, specialization	
recordID	None	date, historyQuestions, medications, symptomName	
prescriptionID	None	medicineName, dosage, startDate, endDate	
fitnessID	None	caloriesBurned, steps, sleepDuration, date	

Users: Since the only functional dependency has userID on the left-hand side and userID is the primary key, this relation is in BCNF.

Patients: Since the only functional dependency has patientID on the left-hand side and patientID is the primary key, this relation is in BCNF.

Doctors: Since the only functional dependency has docID on the left-hand side and docID is the primary key, this relation is in BCNF.

Medical Record: Since the only functional dependency has recordID on the left-hand side and recordID is the primary key, this relation is in BCNF.

Prescriptions: Since the only functional dependency has prescriptionID on the left-hand side and prescriptionID is the primary key, this relation is in BCNF.

Fitness: Since the only functional dependency has fitnessID on the left-hand side and fitnessID is the primary key, this relation is in BCNF.

4) Relational Schema

Entities

User(userID: INT [PK], username: VARCHAR(20), password: VARCHAR(20), role: INT)

Patient(patientID: INT [PK], gender: INT, ageInYears: INT, ageInMonths: INT, pregnancyStatus: INT)

Doctor(docID: INT [PK], docName: VARCHAR(30), specialization: VARCHAR(30))

MedicalRecord(recordID: INT [PK], date: DATE, historyQuestions: VARCHAR(30), medications: VARCHAR(30), symptomName: VARCHAR(30), patientID: INT [FK to Patient.patientID])

Prescription(prescriptionID: INT [PK], medicineName: VARCHAR(30), dosage: VARCHAR(30), startDate: DATE, endDate: DATE, patientID: INT [FK to Patient.patientID], docID: INT [FK to Doctor.docID], recordID: INT [FK to MedicalRecord.recordID])

Fitness(fitnessID: INT [PK], caloriesBurned: REAL, steps: INT, sleepDuration: REAL, date: DATE, patientID: INT [FK to Patient.patientID])

Relations with schemas/tables

Treat(docID: INT [PK, FK to Doctor.docID], patientID: INT [PK, FK to Patient.patientID])

Review(docID: INT [PK, FK to Doctor.docID], recordID: INT [PK, FK to MedicalRecord.recordID])

Note: All relevant foreign keys have been added into the schemas for relationships where the corresponding entity represents the "many" side and the other entity represents the "one" side. Additionally, separate tables have been created for all many-to-many relationships, following the rules and format as described in the track.