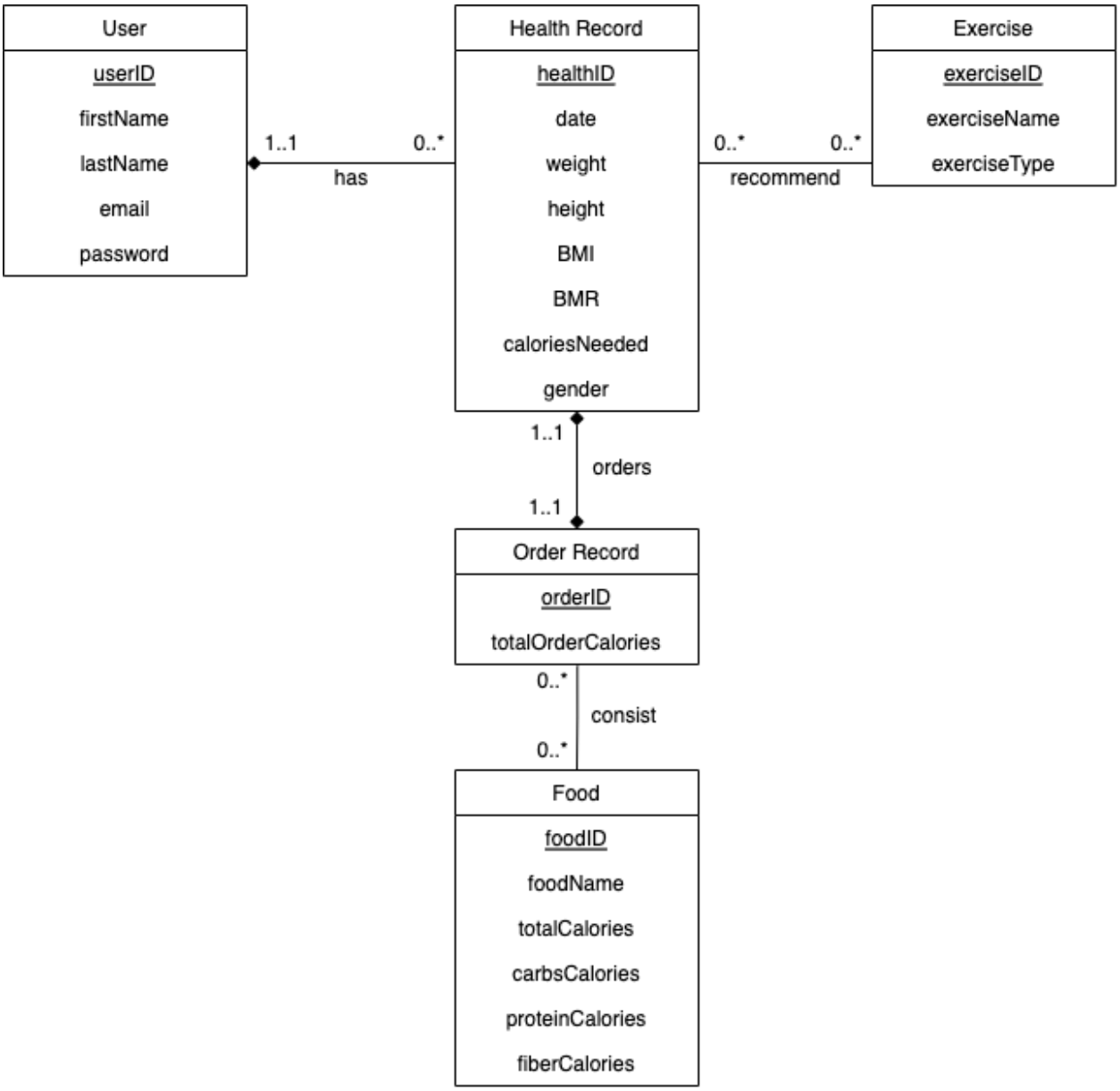
Stage 2: Conceptual and Logical Database Design (15%)

UML DIAGRAM



RELATIONAL SCHEMA

```
CREATE TABLE user(
  userID INT,
  firstName VARCHAR(255),
  lastName VARCHAR(255),
  email VARCHAR(255),
  pass VARCHAR(255),
  PRIMARY KEY(userID)
);

CREATE TABLE health_record(
  healthID INT,
  userID INT,
  gender VARCHAR(1),
  weight REAL,
  height REAL,
  BMI REAL,
  BMR REAL,
  CaloriesNeeded REAL,
  curr_date DATE,
  PRIMARY KEY(healthID),
  FOREIGN KEY(userID) REFERENCES user(userID) ON DELETE CASCADE
);

CREATE TABLE food(
  foodID INT,
  foodName VARCHAR(255),
  totalCalories REAL,
  carbsCalories REAL,
  proteinCalories REAL,
  fiberCalories REAL,
  PRIMARY KEY(foodID)
);

CREATE TABLE order_record(
  orderID INT,
  healthID INT,
  totalOrderCalories REAL,
  FOREIGN KEY(healthID) REFERENCES health_record(healthID) ON DELETE CASCADE,
  PRIMARY KEY(orderID)
);

CREATE TABLE consist(
  foodID INT,
  orderID INT,
  FOREIGN KEY(foodID) REFERENCES food(foodID) ON DELETE CASCADE,
  FOREIGN KEY(orderID) REFERENCES order_record(orderID) ON DELETE CASCADE
);

CREATE TABLE exercise(
exerciseID INT,
```

```
exerciseName VARCHAR(255),
exerciseType VARCHAR(255),
PRIMARY KEY(exerciseID)
);

CREATE TABLE recommend(
  healthID INT,
  exerciseID INT,
  FOREIGN KEY(healthID) REFERENCES health_record(healthID) ON DELETE CASCADE,
  FOREIGN KEY(exerciseID) REFERENCES exercise(exerciseID) ON DELETE CASCADE
);
```

UML DESCRIPTION:

ENTITIES:

1. USER
    a. This will be a database for storing the user information regarding to log-in and sign-up

       Reason why we make it as an entity:

        i. To store all of the user information in one place, so user can keep track their account.

    b. Content:

        i. UserID: will be generated by the developer
        ii. firstName
        iii. lastName
        iv. email
        v. password
2. HEALTH RECORD
    a. This will be a database for storing the user information regarding to its health condition at a certain time, characterized by healthID. Each user can have many health records. Reason why we make it as an entity:

        i. We want to use the information stored in health record to suggest exercises to the user based on their statistics.

ii. We want to save the user health record so we can display it's progress (which can't be done if we only use the local storage). This progress is tracked per healthID and the time is saved in the date attribute.

b. Content:

i. healthID: will be generated by the developer

ii. gender: inputed by user, 'M' → Male 'F' → Female (biologically to be used at calories determination)

iii. weight: inputed by user (in kg)

iv. height: inputed by user (in cm)

v. BMI: to indicate if a user is under/over/ideal weight. Formula: weight (kg) / [height (m)]2

vi. BMR: to figure out calories needed, will create a program to calculate BMR based on

vii. caloriesNeeded: from the BMR, we will calculate how many calories is healthy for the user to intake per day

viii. date: to keep track on previous user's health records.

3. ORDER

a. In this database, we will be storing the food orders made by the user at a certain time. An order is characterized uniquely by orderID. Reason we make it as entity:

i. We want to keep track of what the user orders at a certain time (which is why each healthID is linked to exactly one order).

b. Content:

i. orderID: will be generated by the developer

ii. totalOrderCalories: sums up the calories of the foods in that order.

4. FOOD

a. This will be the database for all the foods taken from (*link*) and its nutritional value

b. Content:
    i. foodID: will be generated by the developer
    ii. foodName: from data
    iii. totalCalories: from data (kcal)
    iv. carbs: from data (g)
    v. protein: from data (g)
    vi. fibre: from data (g)

5. EXERCISE
    a. This will be the database for exercise taken from (*link*). Each exercise is characterized by an exerciseID, and we have additional attributes exerciseName and exerciseType . Reason why we make it as an entity:
        i. So based on the health record, we can also suggest exercises to help them maintain/increase/decrease their BMI. For example, if the person's health record determines that he/she is underweight, we would suggest low intensity workouts from this database.
    b. Content:
        i. exerciseID: will be generated by developer
        ii. exerciseName: retrieved from database, name of the exercise (cycling, running etc)
        iii. exerciseType: retrieved from database, describes the type of the relationship (cardio, high intensity, low intensity, etc)

RELATIONSHIPS DESCRIPTION:

1. User - Health Record Relationship (One to many)
    - Each user can have multiple health records based on their stats submission. When a user logs in and enters a new stats (ie. weight has decreased), a new health record is generated. So, a user can have multiple health records but each health record belongs to exactly one user.
2. Health Record - Exercise Relationship (Many to many)

- Many exercises can be suggested to a health record, and a health record can have many exercises. For example, a user can be underweight and be suggested various low intensity workouts that can keep them fit. Each workout can also be suggested to many different health records, so it is a many to many relationship.

3. Health Record - Order (One to one)
    - For each health record, there can only one food order placed. This is because the user flow is log in > enter stats > health record generated > user gets to order foods. That order will be uniquely identified by orderID, and will be tied to exactly one healthID.
4. Order - Food (Many to many)
    - For each order, there can be many foods in the cart. Each food can also belong to multiple different orders. So it is a many-to-many relationship.

FORMULAS:

- Calorie needed:

1. If you are sedentary (little or no exercise) : Calorie-Calculation = BMR x 1.2.
2. If you are lightly active (light exercise/sports 1-3 days/week) : Calorie-Calculation = BMR x 1.375.

- Carb needed:

1. start by determining your daily calorie need and divide that number in half. That's how many calories should come from carbohydrates.
2. Each gram of carbohydrate has four calories. Divide the number you got from the first step by four.
3. The final number is equal to the number of carbohydrates in grams you need each day.

- Protein needed: **multiply your weight in pounds by 0.36**
  ,
- Fibre needed: 0.014 multiplied by calorie gives answer in grams

- BMI: weight (kg) / [height (m)]2
- BMR: (https://www.checkyourhealth.org/eat-healthy/cal_calculator.php#:~:text=To determine your total daily,Calorie-Calculation %3D BMR x 1.375)
- Total Calorie: $\sum_{i=1}^{n} CALORIE_i$ where n is the number of food item.