COMP3005 Project V2

Jearina Tseung

101199241

Video Link: COMP3005 Project V2 Video

https://cmailcarletonca-my.sharepoint.com/:f:/g/personal/jearinatseung_cmail_carleton_ca/EvZ510Ay_MhArxr2KLh6X4cBzdum5HTmPnCGIRT4AMBMgA?e=7NJJ3A

Github Link: https://github.com/j-tseung/COMP3005_ProjectV2

Contents

Conceptual Design

The conceptual design for the Health and Fitness Club Management System involves a relational database structured to manage and facilitate operations related to members, trainers, and administrative processes. Below is an outline of the database schema, detailing entities, relationships, and key assumptions. **You can find the ER Diagram and Database Schema at the end of this document.**

Entities and Relationships:

- **Member Accounts**:
    - Contains: Email (unique, primary key), Name, Password, Gender, Age
    - Relationships: Central to Exercise Routines, Fitness Achievements, Health Statistics, Fitness Goals, Member Health Metrics
- **Exercise Routines**:
    - Linked to: Member Accounts
    - Contains: Routine details
    - Purpose: Tracks personalized exercise plans
- **Fitness Achievements**:
    - Linked to: Member Accounts
    - Contains: Achievements like fitness goals, ability metrics (e.g., can do pushups)
    - Purpose: Monitors and rewards member progress
- **Health Statistics**:
    - Linked to: Member Accounts
    - Contains: Detailed metrics like fitness level, strength, flexibility
    - Purpose: Supports personalized fitness programming
- **Fitness Goals**:
    - Linked to: Member Accounts
    - Contains: Personal fitness objectives
    - Purpose: Drives member motivation and tracking
- **Member Health Metrics**:
    - Linked to: Member Accounts

    - Contains: Physiological data (height, weight, body fat percentage)
    - Purpose: Provides a health baseline for fitness planning
- **Trainer Accounts**:
    - Contains: ID (primary key), Name, Password, Weekly Availability
    - Purpose: Manages trainer profiles and availability for scheduling
- **Rooms**:
    - Contains: Room ID (primary key), Name, Availability
    - Purpose: Manages physical resources for activities
- **Bookings**:
    - Links: Trainers, Rooms
    - Contains: Booking details (time, duration)
    - Purpose: Organizes personal training sessions and room reservations
- **Equipment**:
    - Linked to: Rooms
    - Contains: Equipment status and maintenance data
    - Purpose: Ensures equipment is fit for use
- **Class Schedule**:
    - Links: Trainers, Rooms
    - Contains: Class times, durations
    - Purpose: Organizes group fitness sessions
- **Payments**:
    - Linked to: Member Accounts
    - Contains: Transaction records (amounts, dates, types)
    - Purpose: Handles billing and financial transactions

Assumptions:

- **Cardinality**: Each member can have multiple associated records in tables like Exercise Routines and Health Statistics, but these records cannot exist without a corresponding member.

- **Mandatory Participation**: Essential data such as Member Accounts must exist for any related records in other tables (e.g., Fitness Goals, Health Statistics).
- **Data Integrity**: Constraints ensure accuracy (e.g., unique emails, range checks on numeric fields).
- **Dynamic Availability**: Trainers and rooms have modifiable availability settings to reflect real-time changes.
- **Maintenance Tracking**: Regular updates are assumed for equipment and room availability to maintain service quality.

For the ER Model, I used UML-Like Notation

Reduction to Relation Schemas

In the process of designing the database schema, each entity is represented as a corresponding table, with the entity's name serving as the table's identifier. For instance, the "Member Accounts" entity translates seamlessly into a table named "member_accounts," encompassing columns for attributes like email, name, password, gender, and age. This structural transformation ensures clarity and organization in storing information.

To maintain data integrity and facilitate meaningful relationships between tables, foreign keys are employed. For example, the relationship between "exercise_routines" and "member_accounts" is established through the email attribute, indicating that each member can have multiple exercise routines. Such one-to-many relationships are pivotal in structuring the database effectively.

Furthermore, the schema incorporates various constraints to uphold data consistency and enforce rules. Simple attributes are mapped directly to columns, with corresponding data types ensuring accurate representation. Constraints, such as ensuring fitness_level falls within a specified range in the "health_statistics" table, safeguard the integrity of the data.

DDL File

DDL File is called tables.sql which can be found on GitHub repository or can use the text below.

```
-- Member Accounts
CREATE TABLE member_accounts (
    email VARCHAR(255) UNIQUE PRIMARY KEY,
    name VARCHAR(255),
    password TEXT,
    gender TEXT,
    age INT
);

-- Exercise Routines
CREATE TABLE exercise_routines (
    email VARCHAR(255) PRIMARY KEY REFERENCES member_accounts(email),
    routine1 TEXT,
    routine2 TEXT,
    routine3 TEXT
);

-- Fitness Achievements
CREATE TABLE fitness_achievements (
    email VARCHAR(255) PRIMARY KEY REFERENCES member_accounts(email),
    first_fitness_goal_achieved BOOLEAN,
    never_skipped_leg_day BOOLEAN,
    can_do_pushup BOOLEAN,
    can_do_pullup BOOLEAN,
```

```
    can_touch_toes BOOLEAN,
    achieved_weight_loss_goal BOOLEAN,
    achieved_muscle_gain_goal BOOLEAN
);

-- Health Statistics
CREATE TABLE health_statistics (
    email VARCHAR(255) PRIMARY KEY REFERENCES member_accounts(email),
    fitness_level INT CHECK (fitness_level BETWEEN 1 AND 10),
    strength INT CHECK (strength BETWEEN 1 AND 10),
    flexibility INT CHECK (flexibility BETWEEN 1 AND 10),
    endurance INT CHECK (endurance BETWEEN 1 AND 10),
    stamina INT CHECK (stamina BETWEEN 0 AND 10),
    has_water BOOLEAN,
    has_protein BOOLEAN,
    is_injured BOOLEAN
);

-- Fitness Goals
CREATE TABLE fitness_goals (
    email VARCHAR(255) PRIMARY KEY REFERENCES member_accounts(email),
    goal1 TEXT,
    goal2 TEXT,
    goal3 TEXT
);

-- Member Health Metrics
```

```
CREATE TABLE member_health_metrics (
    email VARCHAR(255) PRIMARY KEY REFERENCES member_accounts(email),
    height DECIMAL,
    weight DECIMAL,
    body_fat_percentage DECIMAL,
    resting_heart_rate INT
);

-- Trainer Accounts
CREATE TABLE trainer_accounts (
    trainer_id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    password TEXT,
    monday_available BOOLEAN,
    tuesday_available BOOLEAN,
    wednesday_available BOOLEAN,
    thursday_available BOOLEAN,
    friday_available BOOLEAN,
    saturday_available BOOLEAN,
    sunday_available BOOLEAN
);

-- Rooms
CREATE TABLE rooms (
    room_id SERIAL PRIMARY KEY,
    room_name VARCHAR(255) NOT NULL,
    room_availability BOOLEAN
```

```
);

-- Bookings
CREATE TABLE bookings (
    booking_id SERIAL PRIMARY KEY,
    trainer_id INT NOT NULL REFERENCES trainer_accounts(trainer_id),
    room_id INT NOT NULL REFERENCES rooms(room_id),
    duration INT,
    day_of_week VARCHAR(3) CHECK (day_of_week IN ('Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun')),
    start_time TIME
);

-- Equipment
CREATE TABLE equipment (
    equipment_id SERIAL PRIMARY KEY,
    equipment_name VARCHAR(255) NOT NULL,
    room_id INT NOT NULL REFERENCES rooms(room_id),
    quality INT CHECK (quality BETWEEN 1 AND 10)
);


-- Class Schedule
CREATE TABLE class_schedule (
    class_id SERIAL PRIMARY KEY,
    class_name VARCHAR(255) NOT NULL,
    trainer_id INT NOT NULL REFERENCES trainer_accounts(trainer_id),
```

```
    room_id INT NOT NULL REFERENCES rooms(room_id),
    day_of_week VARCHAR(3) CHECK (day_of_week IN ('Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun')),
    start_time TIME,
    duration INT
);

-- Create Payment Table
CREATE TABLE payments (
    payment_id SERIAL PRIMARY KEY,
    email VARCHAR(255) REFERENCES member_accounts(email),
    amount DECIMAL(10, 2) NOT NULL,
    payment_date DATE NOT NULL,
    payment_type VARCHAR(255) NOT NULL,
    status VARCHAR(50) NOT NULL
);
```

DML File

DML File is called sample_data.sql. Can be found on GitHub repository or can use the text below.

```
-- guest account

INSERT INTO member_accounts (email, name) VALUES ('guest', 'guest');


-- admin account

INSERT INTO member_accounts (email, password) VALUES ('admin',
'$2b$12$kEILKLegjmsn.1MOnMlMk.JDLXTLOzlpoeRDmz2GmhNecmpW/Qnnm');

-- random accounts

INSERT INTO member_accounts (email, name, password, gender, age)

VALUES

  ('jane.doe@example.com', 'Jane Doe', 'password123', 'Female', 30),

  ('john.smith@example.com', 'John Smith', 'password456', 'Male', 35),

  ('alice.johnson@example.com', 'Alice Johnson', 'password789', 'Female', 28),

  ('dave.wilson@example.com', 'Dave Wilson', 'passwordABC', 'Male', 40),

  ('emily.white@example.com', 'Emily White', 'passwordDEF', 'Female', 32);


-- sample trainers_accounts (no password)

INSERT INTO trainer_accounts (trainer_id, name, password, monday_available,
tuesday_available, wednesday_available, thursday_available, friday_available,
saturday_available, sunday_available)

VALUES

  (1, 'John Doe', '', true, false, true, false, true, false, true),

  (2, 'Jane Smith', '', false, true, false, true, false, true, false),

  (3, 'Michael Johnson', '', true, false, true, false, true, false, true),

  (4, 'Emily Brown', '', false, true, false, true, false, true, false),

  (5, 'David Lee', '', true, true, true, true, true, true, true);
```

```
-- Sample Rooms
INSERT INTO rooms (room_id, room_name, room_availability) VALUES
(1, 'Yoga Room', true),
(2, 'Personal Training Room', true),
(3, 'Spin Room', true),
(4, 'Dance Room', true),
(5, 'Strength Training', true);

-- Sample Room Bookings
INSERT INTO bookings (booking_id, trainer_id, room_id, duration, day_of_week, start_time)
VALUES
(1, 1, 2, 60, 'Mon', '09:00:00'),
(2, 2, 4, 45, 'Tue', '14:30:00'),
(3, 3, 3, 120, 'Wed', '11:00:00'),
(4, 4, 1, 30, 'Thu', '16:00:00'),
(5, 5, 5, 90, 'Fri', '10:30:00');



-- sample equipment
INSERT INTO equipment (equipment_name, room_id, quality)
VALUES
('Treadmill', 1, 10),
('Yoga Mat', 1, 8),
('Yoga Blocks', 1, 3),
```

('Yoga Strap', 1, 7),

('Stepper Machine', 1, 8),

('Elliptical Trainer', 2, 1),

('Weight Bench', 2, 9),

('Spin Bike', 3, 8),

('Stationary Bike', 3, 7),

('Exercise Bike', 1, 7),

('Medicine Balls', 3, 7),

('Resistance Bands', 3, 8),

('Dance Pole', 4, 7),

('Dance Ribbons', 4, 1),

('Dance Barre', 4, 8),

('Jump Ropes', 4, 6),

('Barbell', 5, 9),

('Bench Press', 5, 8),

('Power Rack', 5, 10),

('Dumbbells', 5, 3);

INSERT INTO class_schedule (class_name, trainer_id, room_id, day_of_week, start_time, duration)

VALUES

('Yoga Class', 1, 1, 'Mon', '09:00:00', 60),

```
    ('Spin Class', 2, 3, 'Tue', '14:00:00', 45),
    ('Zumba Class', 3, 4, 'Wed', '18:30:00', 60),
    ('Pilates Class', 4, 1, 'Thu', '10:00:00', 60),
    ('HIIT Class', 5, 5, 'Fri', '16:00:00', 45);


-- Insert Sample Data into Payment Table
INSERT INTO payments (email, amount, payment_date, payment_type, status)
VALUES
    ('jane.doe@example.com', 50.00, '2024-04-01', 'Annual Membership', 'Refunded'),
    ('john.smith@example.com', 75.00, '2024-04-02', 'Personal Training', 'Pending'),
    ('alice.johnson@example.com', 20.00, '2024-04-03', 'Yoga Class', 'Completed'),
    ('dave.wilson@example.com', 100.00, '2024-04-04', 'Annual Membership', 'Refunded'),
    ('emily.white@example.com', 65.00, '2024-04-05', 'HIIT Class', 'Refunded');
```

Implementation

Application Architecture The Health and Fitness Club Management System is designed as a Command-Line Interface (CLI) application, which interacts directly with a relational database.

Programming Language and Database

- Programming Language: Python
- Database System: PostgreSQL
- Database Access: psycopg2 library

System Features:

1. Command-Line Interface: The application is run entirely from the command line, providing a straightforward and efficient user interaction model.
2. Modular Design: The application is structured into multiple Python modules, each handling different aspects of the system:
    - Member.py: Manages member-related activities such as registration, profile updates, and fitness goal tracking.
    - Trainer.py: Handles trainer schedules and provides access to member profiles.
    - Admin.py: Oversees room bookings, equipment maintenance, class scheduling, and billing processes.
    - Fitness.py: Includes methods for members to log activities, track achievements, and view their health statistics.
    - DatabaseManager.py: Manages database connections and queries.
    - ClearScreen.py: Utility for clearing the command-line screen to keep the interface clean.
3. Database Utilization:
    - The system uses PostgreSQL, a powerful relational database, to store and manage all data efficiently.
    - Tables are designed to handle various data types and relationships, ensuring data integrity and reflecting real-world interactions within the club.
    - Use of SQL transactions to maintain data consistency across operations, especially in areas like booking management and payment processing.
4. Data Integrity and Security:

- Database constraints and foreign keys are used extensively to ensure data integrity.
- Passwords are hashed using bcrypt to ensure security during member and trainer logins.

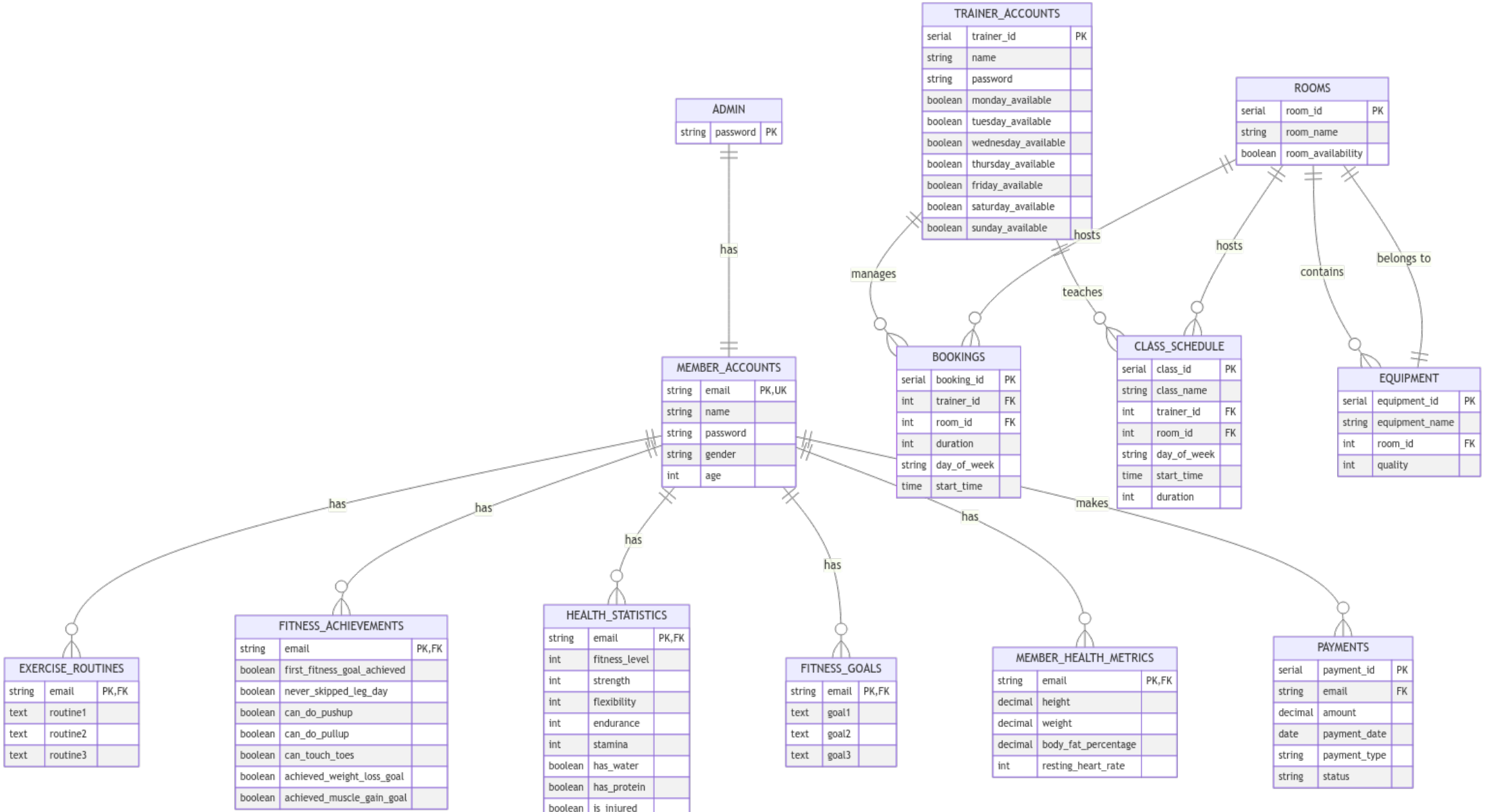
5. Interaction with Database:
    - All data retrieval and manipulation are done through SQL queries from within the application, using psycopg2 as the database adapter.
    - The application provides robust error handling to manage database connection issues and SQL errors, ensuring the system remains reliable and user-friendly.

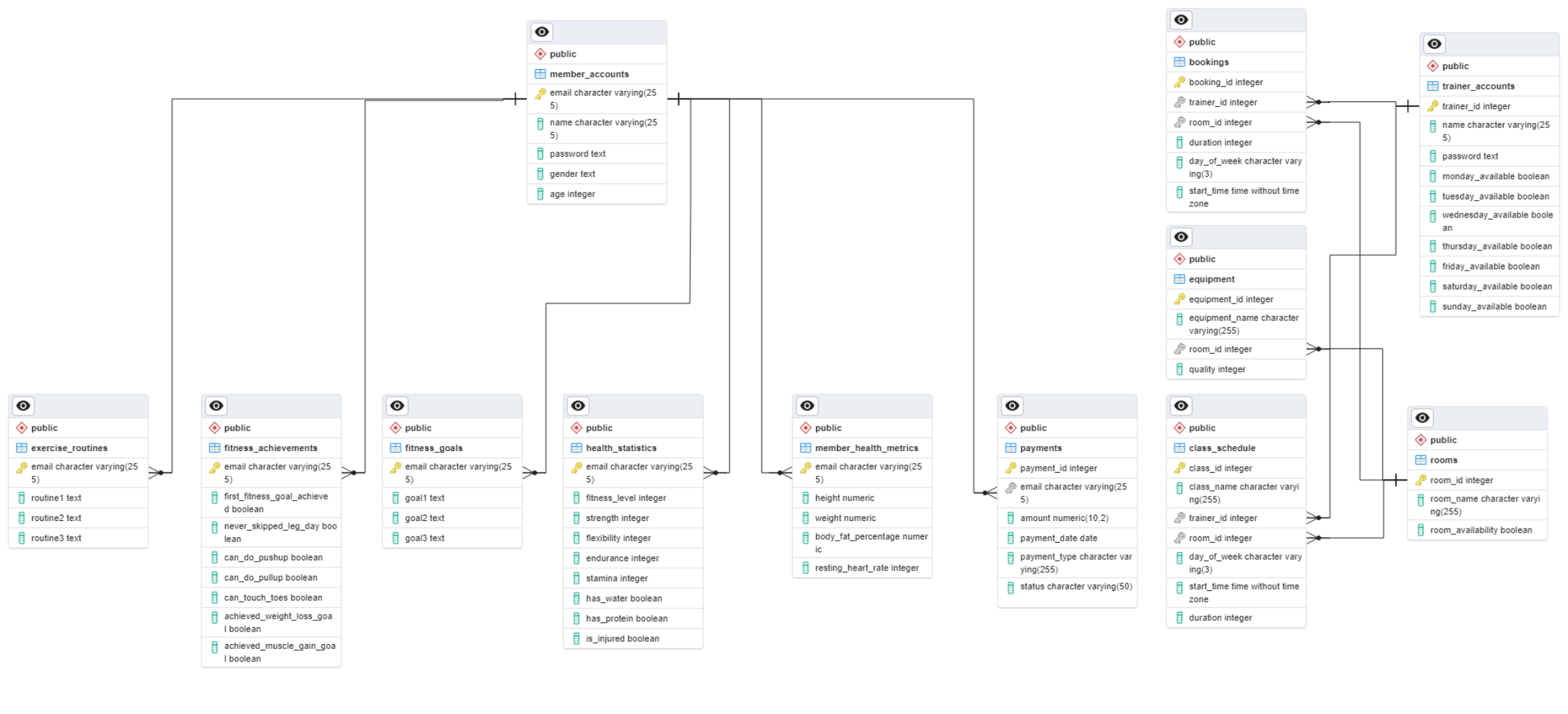
Bonus Features

1. Going to the gym:
    a. System offers an “interactive rpg” gym simulator, giving guests and members health stats like stamina, strength, endurance, etc that have the possibility of levelling up upon finishing a workout. Going to the gym affects equipment maintenance, as “quality of equipment” goes down as users exercise wih it.
2. Guest Access and Handling:
    a. The system offers a unique feature where "guest" users can temporarily input their fitness data without creating a permanent account. This allows potential members to explore the facilities and services offered by the club before committing to a membership.
3. Dynamic Health Statistics Initialization:
    a. For new or guest members, the system dynamically initializes health statistics based on user inputs regarding their fitness level. This personalized setup helps tailor the club's offerings to individual needs right from the start.
4. Password Hashing for Security:
    a. Security is enhanced through bcrypt for hashing passwords. This ensures that even if data access is compromised, the passwords remain secure.
5. Customizable Fitness Goals and Achievements:
    a. Members can customize their fitness goals, and the system tracks their achievements dynamically. This feature motivates members by providing tangible goals and tracking their progress towards these goals.
6. Schedule Management for Trainers:
    a. Trainers can manage their schedules directly through the system, setting their availability on days of the week. This helps optimize the trainers' time and ensures that members have up-to-date information.
7. Health and Fitness Data Visualization:
    a. While the system primarily operates via a command-line interface, it incorporates basic visualizations (like tables and structured outputs) to make the data more comprehensible and easier to interact with, enhancing the user experience.
8. Robust Error Handling and User Prompts:
    a. Comprehensive error handling mechanisms are implemented to ensure the system remains operational and user-friendly, even when encountering unexpected database or input errors.
9. Interactive CLI Menus:
    a. The CLI features interactive menus that guide users through various functionalities, making the system accessible even to those unfamiliar with command-line applications.

ER Model and Database Schema

Scroll pls.


TRAINER_ACCOUNTS
serial trainer_id PK
string name
string password
boolean monday_available
boolean tuesday_available
boolean wednesday_available
boolean thursday_available
boolean friday_available
boolean saturday_available
boolean sunday_available
ROOMS
serial room_id PK
string room_name
boolean room_availability
ADMIN
string password PK
has
manages
hosts
teaches
hosts
contains
belongs to
MEMBER_ACCOUNTS
string email PK,UK
string name
string password
string gender
int age
BOOKINGS
serial booking_id PK
int trainer_id FK
int room_id FK
int duration
string day_of_week
time start_time
CLASS_SCHEDULE
serial class_id PK
string class_name
int trainer_id FK
int room_id FK
string day_of_week
time start_time
int duration
EQUIPMENT
serial equipment_id PK
string equipment_name
int room_id FK
int quality
has
has
has
has
has
makes
EXERCISE_ROUTINES
string email PK,FK
text routine1
text routine2
text routine3
FITNESS_ACHIEVEMENTS
string email PK,FK
boolean first_fitness_goal_achieved
boolean never_skipped_leg_day
boolean can_do_pushup
boolean can_do_pullup
boolean can_touch_toes
boolean achieved_weight_loss_goal
boolean achieved_muscle_gain_goal
HEALTH_STATISTICS
string email PK,FK
int fitness_level
int strength
int flexibility
int endurance
int stamina
boolean has_water
boolean has_protein
boolean is_injured
FITNESS_GOALS
string email PK,FK
text goal1
text goal2
text goal3
MEMBER_HEALTH_METRICS
string email PK,FK
decimal height
decimal weight
decimal body_fat_percentage
int resting_heart_rate
PAYMENTS
serial payment_id PK
string email FK
decimal amount
date payment_date
string payment_type
string status


public
member_accounts
email character varying(255)
name character varying(255)
password text
gender text
age integer
public
exercise_routines
email character varying(255)
routine1 text
routine2 text
routine3 text
public
fitness_achievements
email character varying(255)
first_fitness_goal_achieved boolean
never_skipped_leg_day boolean
can_do_pushup boolean
can_do_pullup boolean
can_touch_toes boolean
achieved_weight_loss_goal boolean
achieved_muscle_gain_goal boolean
public
fitness_goals
email character varying(255)
goal1 text
goal2 text
goal3 text
public
health_statistics
email character varying(255)
fitness_level integer
strength integer
flexibility integer
endurance integer
stamina integer
has_water boolean
has_protein boolean
is_injured boolean
public
member_health_metrics
email character varying(255)
height numeric
weight numeric
body_fat_percentage numeric
resting_heart_rate integer
public
payments
payment_id integer
email character varying(255)
amount numeric(10,2)
payment_date date
payment_type character varying(255)
status character varying(50)
public
bookings
booking_id integer
trainer_id integer
room_id integer
duration integer
day_of_week character varying(3)
start_time time without time zone
public
equipment
equipment_id integer
equipment_name character varying(255)
room_id integer
quality integer
public
class_schedule
class_id integer
class_name character varying(255)
trainer_id integer
room_id integer
day_of_week character varying(3)
start_time time without time zone
duration integer
public
trainer_accounts
trainer_id integer
name character varying(255)
password text
monday_available boolean
tuesday_available boolean
wednesday_available boolean
thursday_available boolean
friday_available boolean
saturday_available boolean
sunday_available boolean
public
rooms
room_id integer
room_name character varying(255)
room_availability boolean