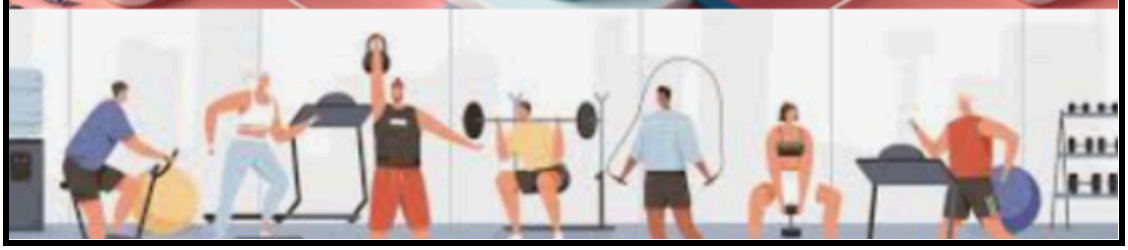
COMPREHENSIVE DATABASE SYSTEM FOR MANAGING FITNESS CENTER OPERATIONS



CONCLUSION

THE FITNESS CENTER DATABASE SYSTEM PROVIDES A SOLID FOUNDATION FOR MANAGING MEMBERSHIPS, CLASS ENROLLMENTS, TRAINER SCHEDULES, AND SUBSCRIPTION PLANS. THROUGH CLEAR RELATIONSHIPS BETWEEN THE ENTITIES, THE SYSTEM ENSURES ACCURATE TRACKING OF MEMBER ACTIVITIES, CLASS PARTICIPATION, AND SUBSCRIPTION DETAILS. THE IMPLEMENTATION OF VIEWS AND QUERIES FURTHER ENHANCES DATA ACCESSIBILITY AND SIMPLIFIES DAILY OPERATIONS FOR BOTH ADMINISTRATORS AND MEMBERS. THIS STRUCTURED AND NORMALIZED APPROACH GUARANTEES EFFICIENT DATA STORAGE AND RETRIEVAL, ENSURING LONG-TERM SCALABILITY.

1. IN THE FUTURE, THE DATABASE COULD BE ENHANCED WITH FEATURES SUCH AS:

PAYMENT TRACKING: ADDING TABLES TO RECORD MEMBER PAYMENTS AND TRACK SUBSCRIPTION RENEWAL HISTORIES.

CLASS FEEDBACK SYSTEM: IMPLEMENTING A FEEDBACK SYSTEM WHERE MEMBERS CAN RATE AND REVIEW CLASSES AND TRAINERS.

SCHEDULING SYSTEM: INTRODUCING A SCHEDULING FEATURE TO ALLOW MEMBERS TO RESERVE CLASSES BASED ON TRAINER AVAILABILITY AND CAPACITY.

NOTIFICATIONS: ENABLING AUTOMATIC NOTIFICATIONS FOR CLASS REMINDERS, SUBSCRIPTION RENEWALS, OR PROMOTIONS.

MOBILE APPLICATION INTEGRATION: CREATING AN API LAYER FOR INTEGRATION WITH A MOBILE APP TO ENHANCE MEMBER INTERACTION AND ALLOW FOR EASY CLASS BOOKINGS, CANCELLATIONS, AND REAL-TIME UPDATES.

Key Entities:

Table - 1

1. Member (member_id, first_name, last_name, email, phone, dob, gender, membership_start, membership_end, *subscription_id (Foreign Key)*)

Table - 2

2. Subscription (subscription_id, plan_name, cost, duration_in_months, features)

Table - 3

3. Trainer (trainer_id, first_name, last_name, specialty, phone, email, experience, salary, availability)

Table - 4

4. Class (class_id, class_name, start_time, end_time, *trainer_id(Foreign Key)*, max_participants)

Table - 5

5. MemberClassEnrollment (enrollment_id, *member_id(Foreign Key),*

(Foreign Key)class_id, enrollment_date)

Relationships:

1. **Member ↔ Subscription (One-to-Many)**

Relationship: Each member subscribes to a specific subscription plan, but a subscription plan can be chosen by multiple members.

Foreign Key: The subscription_id in the Member entity references the subscription_id in the Subscription entity.

Explanation:
- One Subscription can have many Members.
- A Member can only have one Subscription at a time.

2. **Trainer ↔ Class (One-to-Many)**

Relationship: A trainer can teach multiple classes, but each class is assigned to one trainer.

Foreign Key: The trainer_id in the Class entity references the trainer_id in the Trainer entity.

Explanation:
- One Trainer can be assigned to multiple Classes.
- Each Class is conducted by a single Trainer.

3. **Member ↔ Class (Many-to-Many)**

Relationship: A member can enroll in multiple classes, and each class can have multiple members enrolled in it.

Join Table: The MemberClassEnrollment entity serves as a bridge between Member and Class.

- **Foreign Keys**: member_id (from Member) and class_id (from Class) in the MemberClassEnrollment entity.

Explanation:
- Many Members can enroll in many Classes.
- Many Classes can have many Members enrolled in them.

Views

View Name: MemberSubscriptionInfo

Description: The MemberSubscriptionInfo view provides a comprehensive overview of members and their associated subscription details. It displays the following information for each member:

- **Member ID**:
- **Full Name**:
- **Email**:
- **Phone**:
- **Plan Name**:
- **Cost**:
- **Duration**:

This view simplifies access to key member and subscription data, making it easy to analyze membership details at a glance.

Stored Procedures

1. Procedure to Get Members Who Joined After a Specific Date

This procedure retrieves all members who joined after a given date.

We can use the following query to use this procedure.

CALL UpdateMemberPhone(1, '9876543211');

```
DELIMITER ;
CALL UpdateMemberSubscription(1, 3);
DELIMITER //

CREATE PROCEDURE UpdateMemberPhone(
    IN memberId INT,
    IN newPhone VARCHAR(50)
)
BEGIN
    UPDATE Member
    SET phone = newPhone
    WHERE member_id = memberId;
END //

DELIMITER ;

CALL UpdateMemberPhone(1, '9876543211');
```

Software Requirements

1) **Database Management System (DBMS)**:
   - **DBMS**: MySQL
   - **Version**: MySQL 8.0

2) Development Tools:

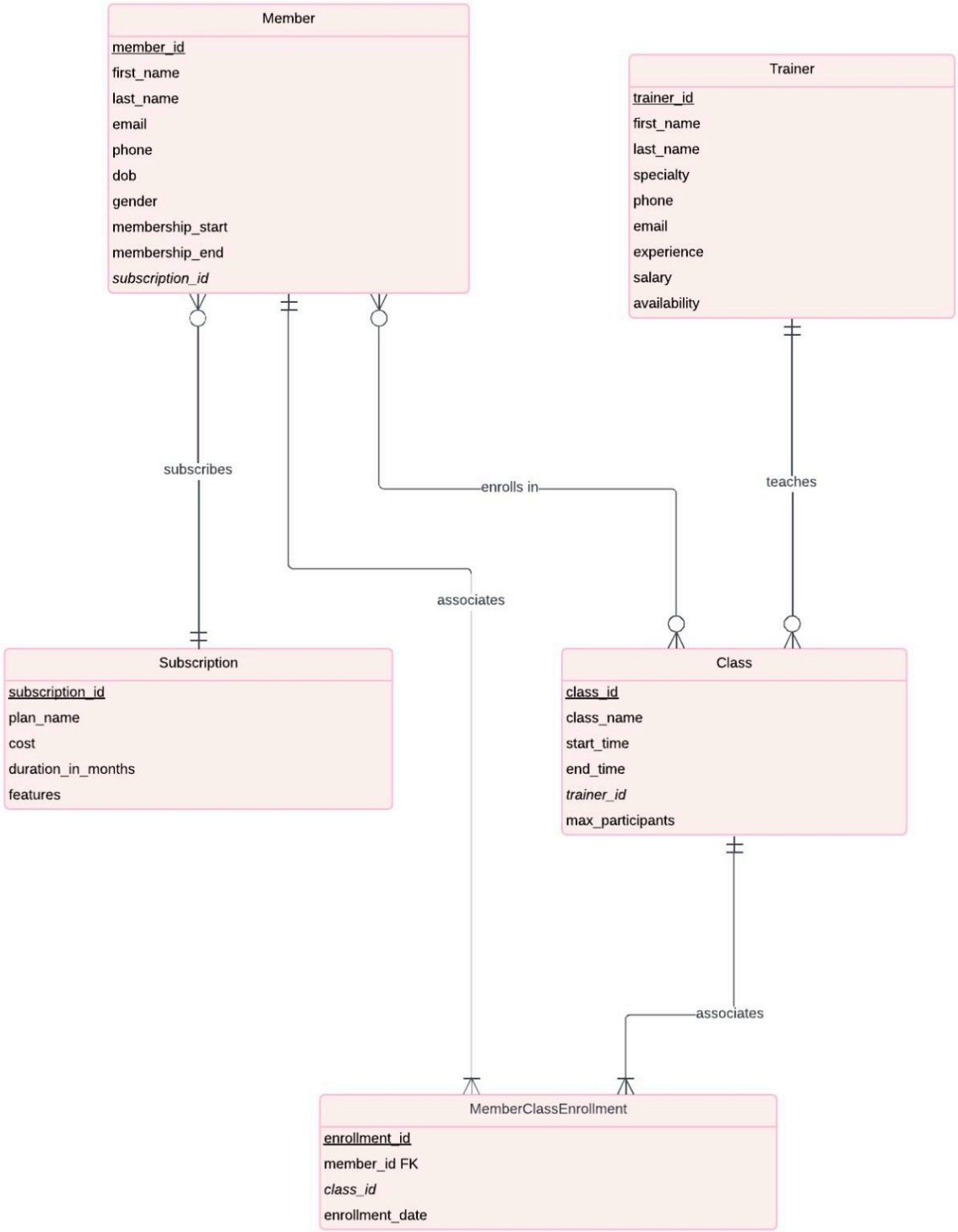
- **Database Management Tool**: MySQL Workbench

3) Operating Systems:

- **Development Environment**:
  - Windows 11

Design Phase

- **ER Diagram:**



4 Implementation

Subscription Table (Schema and Records)

Schema:

```
-- Subscription Table
CREATE TABLE Subscription (
    subscription_id INT PRIMARY KEY AUTO_INCREMENT,
    plan_name VARCHAR(255) NOT NULL,
    cost DECIMAL(10, 2) NOT NULL,
    duration_in_months INT NOT NULL,
    features TEXT
);
```

Records:

```
163 • select * from Subscription;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

subscription_id	plan_name	cost	duration_in_months	features
1	Monthly	50.00	1	Access to gym and yoga classes
2	Quarterly	140.00	3	Access to all facilities and classes
3	Annual	500.00	12	Full access and free personal training sessions
4	Weekly	20.00	0	Access to gym only
5	Student Plan	30.00	1	Discounted rate for students
6	Senior Plan	40.00	1	Access for senior citizens
7	Premium Annual	600.00	12	Full access and unlimited classes
8	Couples Plan	90.00	1	Access for couples
9	Corporate Plan	300.00	6	Corporate discount rate
10	Trial	10.00	0	1-week access to gym
NULL	NULL	NULL	NULL	NULL

Member Table (Schema and Records)

Schema

```sql
-- Member Table
CREATE TABLE Member (
    member_id INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    phone VARCHAR(50) NOT NULL,
    dob DATE NOT NULL,
    gender VARCHAR(10) NOT NULL,
    membership_start DATE NOT NULL,
    membership_end DATE NOT NULL,
    subscription_id INT,
    FOREIGN KEY (subscription_id) REFERENCES Subscription(subscription_id)
);
```

Records

```sql
select * from Member;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

member_id	first_name	last_name	email	phone	dob	gender	membership_start	membership_end	subscription_id
1	Rahul	Sharma	rahul.sharma@example.com	9876543210	1992-01-15	Male	2023-01-01	2023-12-31	1
2	Priya	Verma	priya.verma@example.com	9876543221	1989-02-10	Female	2023-03-01	2023-06-30	2
3	Amit	Patel	amit.patel@example.com	9876543232	1990-05-20	Male	2023-06-01	2023-06-30	4
4	Sneha	Kapoor	sneha.kapoor@example.com	9876543243	1995-11-12	Female	2023-05-01	2023-10-31	3
5	Vikram	Singh	vikram.singh@example.com	9876543254	1988-07-25	Male	2023-07-01	2024-06-30	1
6	Megha	Iyer	megha.iyer@example.com	9876543265	1993-08-14	Female	2023-08-01	2024-07-31	5
7	Arjun	Desai	arjun.desai@example.com	9876543276	1991-03-30	Male	2023-02-01	2023-12-31	3
8	Radhika	Menon	radhika.menon@example.com	9876543287	1994-09-09	Female	2023-09-01	2023-09-30	4
9	Kiran	Rao	kiran.rao@example.com	9876543298	1996-12-01	Male	2023-06-01	2023-06-30	2
10	Sonal	Jain	sonal.jain@example.com	9876543309	1997-04-18	Female	2023-04-01	2023-07-31	1
11	Rahul	Sharma	rahul.sharma@example.com	9876543210	1992-01-15	Male	2023-01-01	2023-12-31	1
12	Priya	Verma	priya.verma@example.com	9876543221	1989-02-10	Female	2023-03-01	2023-06-30	2
13	Amit	Patel	amit.patel@example.com	9876543232	1990-05-20	Male	2023-06-01	2023-06-30	4
14	Sneha	Kapoor	sneha.kapoor@example.com	9876543243	1995-11-12	Female	2023-05-01	2023-10-31	3
15	Vikram	Singh	vikram.singh@example.com	9876543254	1988-07-25	Male	2023-07-01	2024-06-30	1

MemberClassEnrollment Table (Schema and Records)

Schema

```sql
-- MemberClassEnrollment Table (Junction table for many-to-many relation between Member and Class)
CREATE TABLE MemberClassEnrollment (
    enrollment_id INT PRIMARY KEY AUTO_INCREMENT,
    member_id INT,
    class_id INT,
    enrollment_date DATE NOT NULL,
    FOREIGN KEY (member_id) REFERENCES Member(member_id),
    FOREIGN KEY (class_id) REFERENCES Class(class_id)
);
```

Records

```sql
select * from MemberClassEnrollment ;
```

Result Grid | Filter Rows: | Edit:

enrollment_id	member_id	class_id	enrollment_date
1	1	1	2023-08-01
2	2	2	2023-09-15
3	3	3	2023-07-20
4	4	4	2023-09-10
5	5	1	2023-08-05
6	6	5	2023-07-25
7	7	6	2023-08-10
8	8	7	2023-09-01
9	9	8	2023-09-12
10	10	9	2023-08-20
NULL	NULL	NULL	NULL

Trainer Table (schema and Records)

Schema

```sql
-- Trainer Table
CREATE TABLE Trainer (
    trainer_id INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    specialty VARCHAR(255) NOT NULL,
    phone VARCHAR(50) NOT NULL,
    email VARCHAR(255) NOT NULL,
    experience INT NOT NULL,
    salary DECIMAL(10, 2) NOT NULL,
    availability VARCHAR(50) NOT NULL
);
```

Records

```sql
select * from Trainer ;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

trainer_id	first_name	last_name	specialty	phone	email	experience	salary	availability
1	Ravi	Kumar	Yoga	9876543310	ravi.kumar@example.com	5	50000.00	Morning
2	Sanya	Shah	Weight Training	9876543321	sanya.shah@example.com	3	45000.00	Evening
3	Anil	Mehta	Cardio	9876543332	anil.mehta@example.com	7	52000.00	Afternoon
4	Pooja	Chaudhary	Pilates	9876543343	pooja.chaudhary@example.com	4	47000.00	Morning
5	Manish	Joshi	CrossFit	9876543354	manish.joshi@example.com	6	51000.00	Evening
6	Tina	Agarwal	Aerobics	9876543365	tina.agarwal@example.com	2	43000.00	Afternoon
7	Siddharth	Reddy	Swimming	9876543376	siddharth.reddy@example.com	8	55000.00	Morning
8	Neha	Ghosh	Dance Fitness	9876543387	neha.ghosh@example.com	3	44000.00	Evening
9	Abhishek	Bhatt	Strength Training	9876543398	abhishek.bhatt@example.com	10	60000.00	Morning
10	Ritika	Mukherjee	Zumba	9876543409	ritika.mukherjee@example.com	1	40000.00	Afternoon
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Class Table (schema and Records)

Schema

```
42      -- Class Table
43  •   CREATE TABLE Class (
44          class_id INT PRIMARY KEY AUTO_INCREMENT,
45          class_name VARCHAR(255) NOT NULL,
46          start_time TIME NOT NULL,
47          end_time TIME NOT NULL,
48          trainer_id INT,
49          max_participants INT NOT NULL,
50          FOREIGN KEY (trainer_id) REFERENCES Trainer(trainer_id)
51      );
```

Records

```
163 •   select * from class ;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

class_id	class_name	start_time	end_time	trainer_id	max_participants
1	Yoga	08:00:00	09:00:00	1	20
2	Weight Training	18:00:00	19:00:00	2	15
3	Cardio	12:00:00	13:00:00	3	25
4	Pilates	09:00:00	10:00:00	4	10
5	CrossFit	19:00:00	20:00:00	5	15
6	Aerobics	14:00:00	15:00:00	6	20
7	Swimming	07:00:00	08:00:00	7	10
8	Dance Fitness	17:00:00	18:00:00	8	12
9	Strength Training	10:00:00	11:00:00	9	15
10	Zumba	15:00:00	16:00:00	10	18
NULL	NULL	NULL	NULL	NULL	NULL

Database Operations

1) List of All Members and the Total Number of Classes They Have Enrolled In

```
164 •   SELECT
165         m.first_name AS Member_First_Name,
166         m.last_name AS Member_Last_Name,
167         COUNT(e.class_id) AS Total_Classes_Enrolled
168     FROM
169         Member m
170     LEFT JOIN
171         MemberClassEnrollment e ON m.member_id = e.member_id
172     GROUP BY
173         m.member_id, m.first_name, m.last_name;
174
```

2)Classes by Trainer Specialty

This query shows classes categorized by trainer specialties, useful for users looking for specific expertise.

```
SELECT
    t.specialty,
    CONCAT(t.first_name, ' ', t.last_name) AS trainer_name
FROM
    Trainer t
JOIN
    Class c ON t.trainer_id = c.trainer_id
ORDER BY
    t.specialty, c.class_name;
```

3)Classes by Membership Type

This query lists all classes along with the subscription plans that allow access to them, helping users choose classes based on their subscription type.

```sql
SELECT
    c.class_name,
    s.plan_name,
    s.cost,
    CONCAT(t.first_name, ' ', t.last_name) AS trainer_name
FROM
    Class c
JOIN
    Trainer t ON c.trainer_id = t.trainer_id
JOIN
    Member m ON c.class_id = m.subscription_id  -- Assuming a connection exists
JOIN
    Subscription s ON m.subscription_id = s.subscription_id
ORDER BY
    c.class_name;
```

4) List Classes Along with Their Trainers and Number of Participants

This query joins the Class, Trainer, and MemberClassEnrollment tables to find all classes, their trainers, and the number of members enrolled in each class.

```sql
SELECT
    c.class_name AS ClassName,
    t.first_name AS TrainerFirstName,
    t.last_name AS TrainerLastName,
    COUNT(mce.member_id) AS Participants
FROM
    Class c
JOIN
    Trainer t ON c.trainer_id = t.trainer_id
LEFT JOIN
    MemberClassEnrollment mce ON c.class_id = mce.class_id
GROUP BY
    c.class_name, t.first_name, t.last_name;
```

5) Find Members with Annual Subscriptions

This query lists members who have premium or annual subscription plans, helping admins focus on high-value customers for special services or offers.

```sql
SELECT
    m.first_name AS MemberFirstName,
    m.last_name AS MemberLastName,
    s.plan_name AS SubscriptionPlan,
    s.cost
FROM
    Member m
JOIN
    Subscription s ON m.subscription_id = s.subscription_id
WHERE
    s.plan_name IN ('Annual', 'Premium Annual');
```

Results:

1)

Member_First_Name	Member_Last_Name	Total_Classes_Enrolled
Rahul	Sharma	1
Priya	Verma	1
Amit	Patel	1
Sneha	Kapoor	1
Vikram	Singh	1
Megha	Iyer	1
Arjun	Desai	1
Radhika	Menon	1
Kiran	Rao	1
Sonal	Jain	1
Rahul	Sharma	0
Priya	Verma	0
Amit	Patel	0
Sneha	Kapoor	0
Vikram	Singh	0

2)

specialty	trainer_name
Aerobics	Tina Agarwal
Cardio	Anil Mehta
CrossFit	Manish Joshi
Dance Fitness	Neha Ghosh
Pilates	Pooja Chaudhary
Strength Training	Abhishek Bhatt
Swimming	Siddharth Reddy
Weight Training	Sanya Shah
Yoga	Ravi Kumar
Zumba	Ritika Mukherjee

3)

class_name	plan_name	cost	trainer_name
Cardio	Annual	500.00	Anil Mehta
Cardio	Annual	500.00	Anil Mehta
Cardio	Annual	500.00	Anil Mehta
Cardio	Annual	500.00	Anil Mehta
CrossFit	Student Plan	30.00	Manish Joshi
CrossFit	Student Plan	30.00	Manish Joshi
Pilates	Weekly	20.00	Pooja Chaudhary
Pilates	Weekly	20.00	Pooja Chaudhary
Pilates	Weekly	20.00	Pooja Chaudhary
Pilates	Weekly	20.00	Pooja Chaudhary
Weight Tr...	Quarterly	140.00	Sanya Shah
Weight Tr...	Quarterly	140.00	Sanya Shah
Weight Tr...	Quarterly	140.00	Sanya Shah
Weight Tr...	Quarterly	140.00	Sanya Shah
Yoga	Monthly	50.00	Ravi Kumar

4)

ClassName	TrainerFirstName	TrainerLastName	Participants
Yoga	Ravi	Kumar	2
Weight Training	Sanya	Shah	1
Cardio	Anil	Mehta	1
Pilates	Pooja	Chaudhary	1
CrossFit	Manish	Joshi	1
Aerobics	Tina	Agarwal	1
Swimming	Siddharth	Reddy	1
Dance Fitness	Neha	Ghosh	1
Strength Training	Abhishek	Bhatt	1
Zumba	Ritika	Mukherjee	0

5)

MemberFirstName	MemberLastName	SubscriptionPlan	cost
Sneha	Kapoor	Annual	500.00
Arjun	Desai	Annual	500.00
Sneha	Kapoor	Annual	500.00
Arjun	Desai	Annual	500.00

CONCLUSION

THE FITNESS CENTER DATABASE SYSTEM PROVIDES A SOLID FOUNDATION FOR MANAGING MEMBERSHIPS, CLASS ENROLLMENTS, TRAINER SCHEDULES, AND SUBSCRIPTION PLANS. THROUGH CLEAR RELATIONSHIPS BETWEEN THE ENTITIES, THE SYSTEM ENSURES ACCURATE TRACKING OF MEMBER ACTIVITIES, CLASS PARTICIPATION, AND SUBSCRIPTION DETAILS. THE IMPLEMENTATION OF VIEWS AND QUERIES FURTHER ENHANCES DATA ACCESSIBILITY AND SIMPLIFIES DAILY OPERATIONS FOR BOTH ADMINISTRATORS AND MEMBERS. THIS STRUCTURED AND NORMALIZED APPROACH GUARANTEES EFFICIENT DATA STORAGE AND RETRIEVAL, ENSURING LONG-TERM SCALABILITY.

1. IN THE FUTURE, THE DATABASE COULD BE ENHANCED WITH FEATURES SUCH AS:

PAYMENT TRACKING: ADDING TABLES TO RECORD MEMBER PAYMENTS AND TRACK SUBSCRIPTION RENEWAL HISTORIES.

CLASS FEEDBACK SYSTEM: IMPLEMENTING A FEEDBACK SYSTEM WHERE MEMBERS CAN RATE AND REVIEW CLASSES AND TRAINERS.

SCHEDULING SYSTEM: INTRODUCING A SCHEDULING FEATURE TO ALLOW MEMBERS TO RESERVE CLASSES BASED ON TRAINER AVAILABILITY AND CAPACITY.

NOTIFICATIONS: ENABLING AUTOMATIC NOTIFICATIONS FOR CLASS REMINDERS, SUBSCRIPTION RENEWALS, OR PROMOTIONS.

MOBILE APPLICATION INTEGRATION: CREATING AN API LAYER FOR INTEGRATION WITH A MOBILE APP TO ENHANCE MEMBER INTERACTION AND ALLOW FOR EASY CLASS BOOKINGS, CANCELLATIONS, AND REAL-TIME UPDATES.