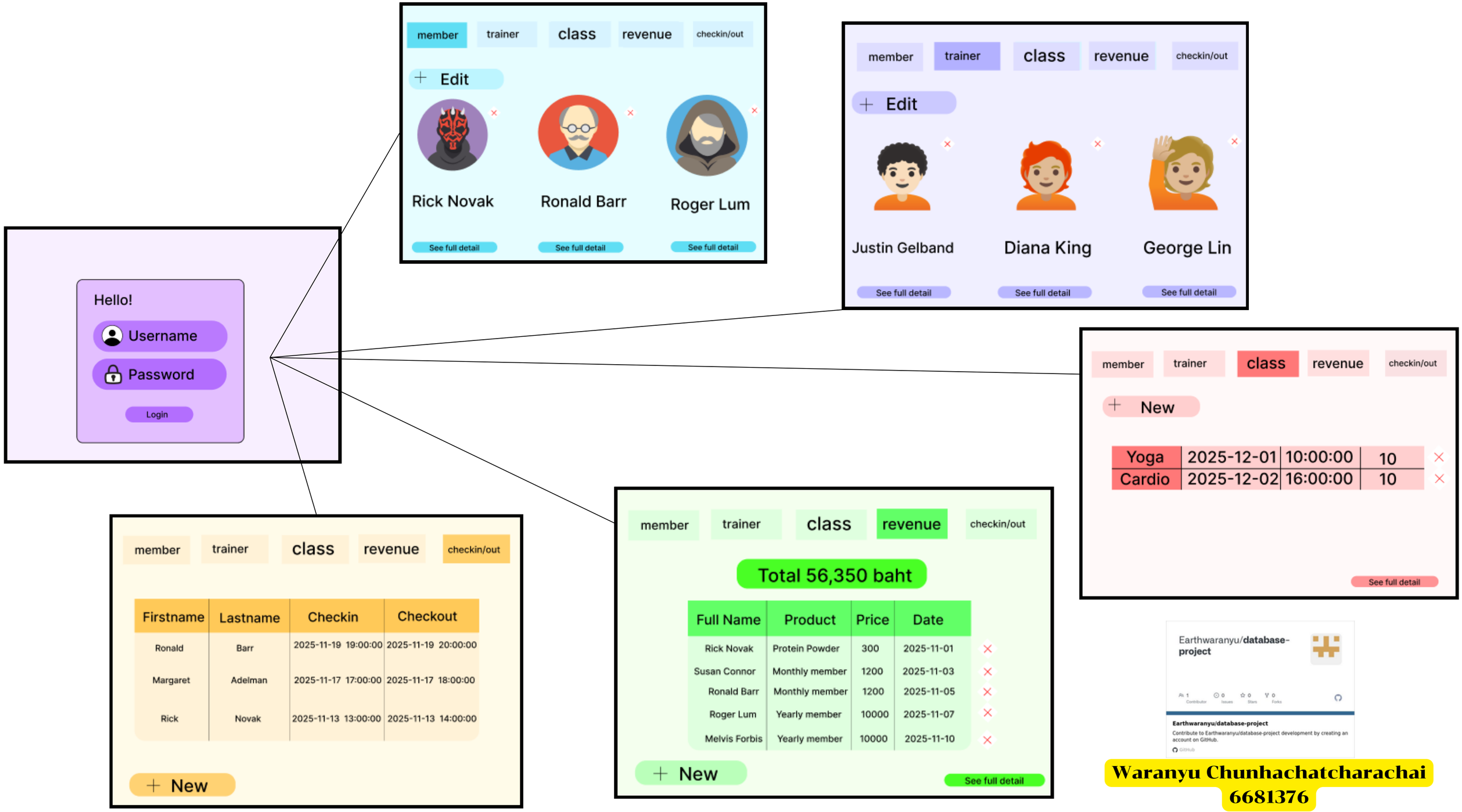
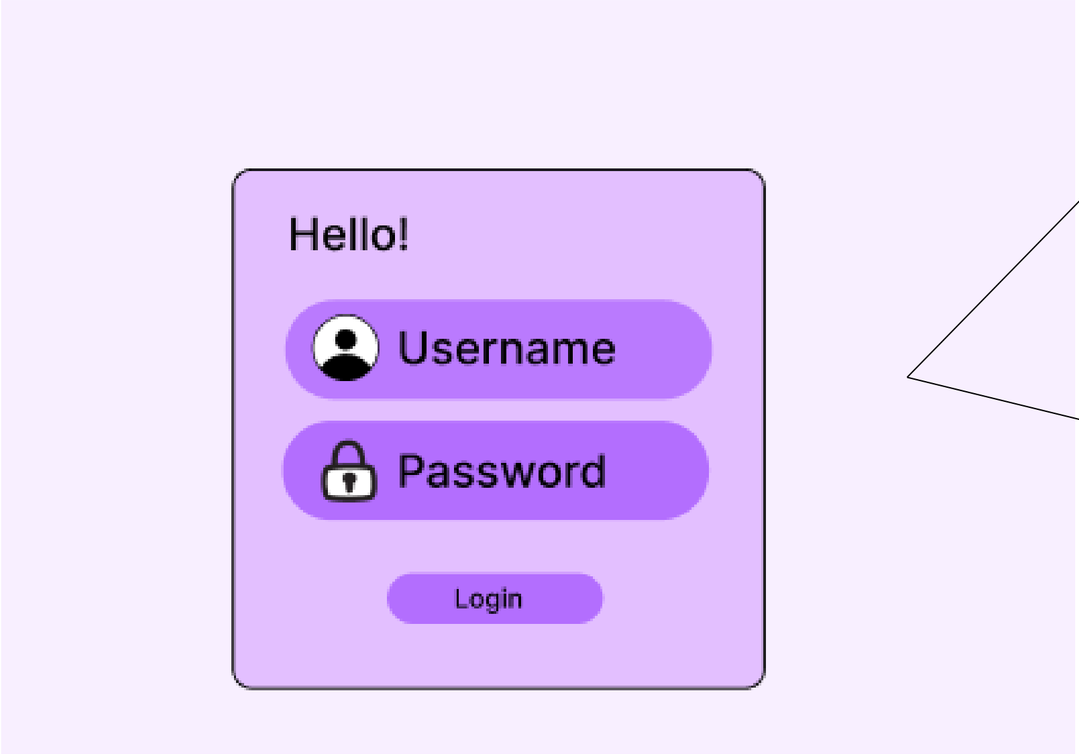

member
trainer
class
revenue
checkin/out
Edit
Rick Novak
Ronald Barr
Roger Lum
See full detail
See full detail
See full detail
member
trainer
class
revenue
checkin/out
Edit
Justin Gelband
Diana King
George Lin
See full detail
See full detail
See full detail
Hello!
Username
Password
Login
member
trainer
class
revenue
checkin/out
New
Yoga 2025-12-01 10:00:00 10
Cardio 2025-12-02 16:00:00 10
See full detail
member
trainer
class
revenue
checkin/out
Firstname Lastname Checkin Checkout
Ronald Barr 2025-11-19 19:00:00 2025-11-19 20:00:00
Margaret Adelman 2025-11-17 17:00:00 2025-11-17 18:00:00
Rick Novak 2025-11-13 13:00:00 2025-11-13 14:00:00
New
member
trainer
class
revenue
checkin/out
Total 56,350 baht
Full Name Product Price Date
Rick Novak Protein Powder 300 2025-11-01
Susan Connor Monthly member 1200 2025-11-03
Ronald Barr Monthly member 1200 2025-11-05
Roger Lum Yearly member 10000 2025-11-07
Melvis Forbis Yearly member 10000 2025-11-10
New
See full detail
Earthwaranyu/database-project
Earthwaranyu/database-project
Contribute to Earthwaranyu/database-project development by creating an account on GitHub.
Waranyu Chunhachatcharachai
6681376




```sql
DROP TABLE IF EXISTS owner_user CASCADE;

CREATE TABLE owner_user(
    username VARCHAR(50) PRIMARY KEY,
    password VARCHAR(50) NOT NULL
);

INSERT INTO owner_user (username, password)
VALUES ('OWNER', '5555');
```

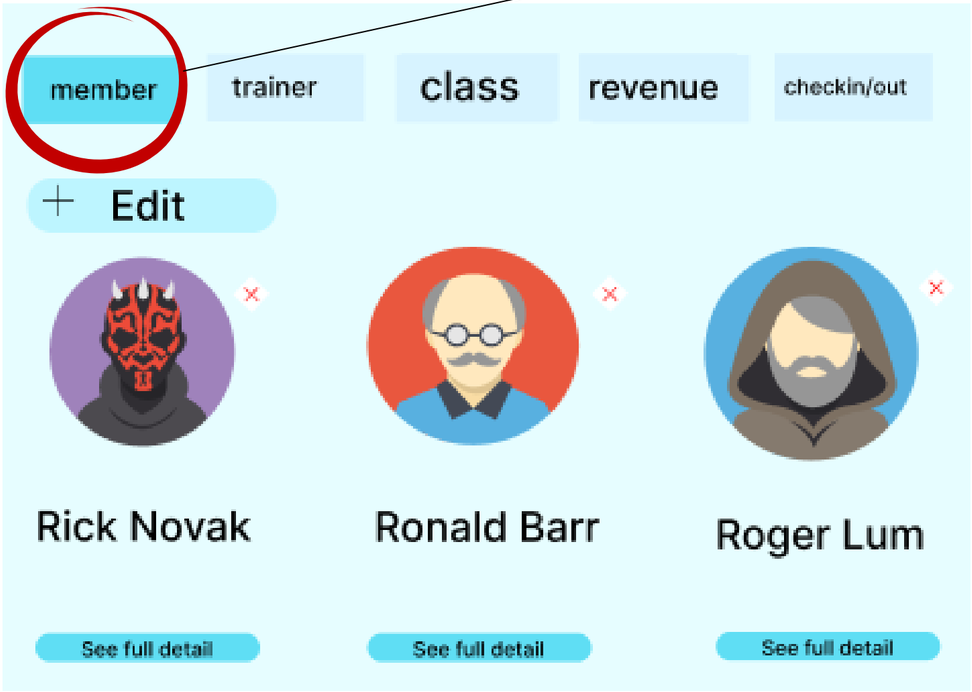
```sql
CREATE OR REPLACE FUNCTION login(input_user VARCHAR, input_password VARCHAR)
RETURNS BOOLEAN AS $$
BEGIN
    RETURN EXISTS(
        SELECT *
        FROM owner_user
        WHERE username = input_user AND password = input_password
    );
end;

$$ LANGUAGE plpgsql;
```

```sql
SELECT login( input_user 'OWNER', input_password '5555')
```

	login
1	true

```sql
SELECT login( input_user 'OWNER', input_password '6666')
```

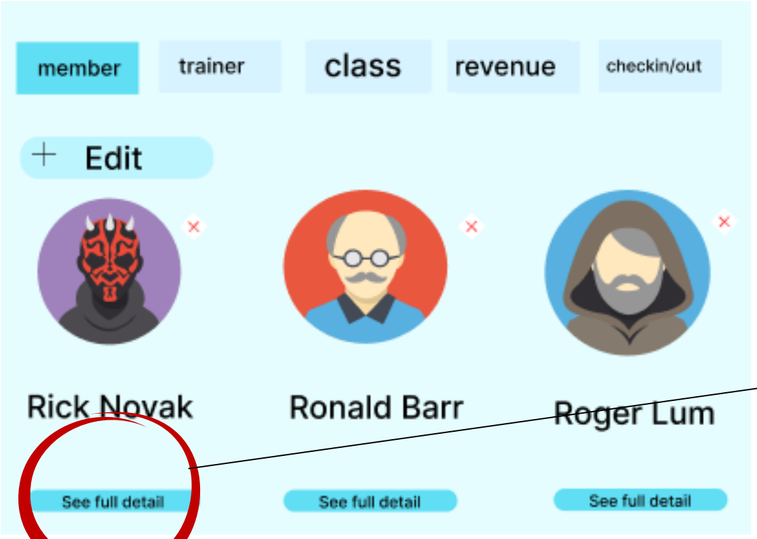
	login
1	false



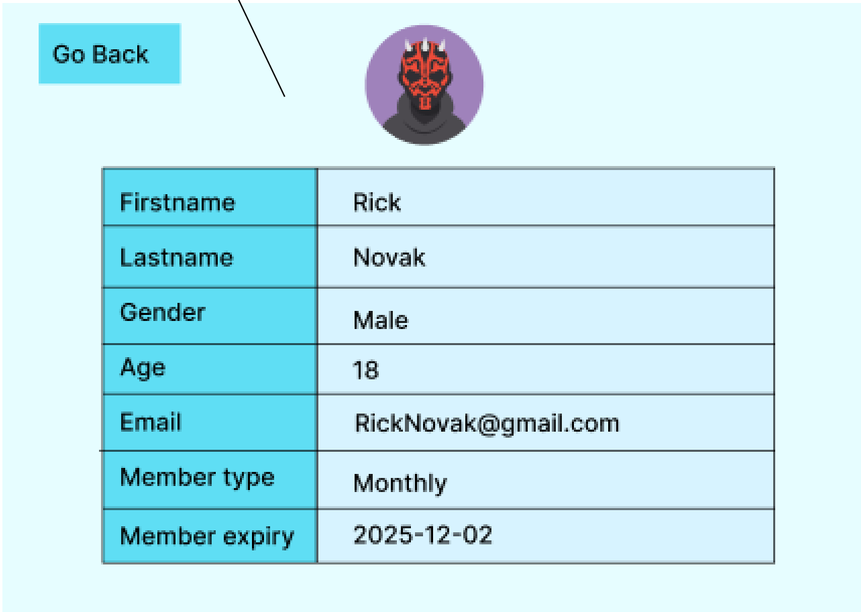

```
DROP FUNCTION IF EXISTS member_page();

CREATE OR REPLACE FUNCTION member_page()
RETURNS TABLE (
    member_id INT,
    first_name VARCHAR,
    last_name VARCHAR
)  AS $$
BEGIN
    RETURN QUERY
    SELECT
        m.member_id,
        m.first_name,
        m.last_name
    FROM member m
    ORDER BY m.member_id ASC;
END;
$$  LANGUAGE plpgsql;


SELECT * FROM member_page();
```

```
Select member_page();
```

	member_page
1	(1,Emily,Johnson)
2	(2,Rick,Novak)
3	(3,Susan,Connor)
4	(4,Margaret,Adelman)
5	(5,Ronald,Barr)
6	(6,Marie,Broadbet)
7	(7,Roger,Lum)
8	(8,Marlene,Donahue)
9	(9,Jeff,Johnson)
10	(10,Melvin,Forbis)



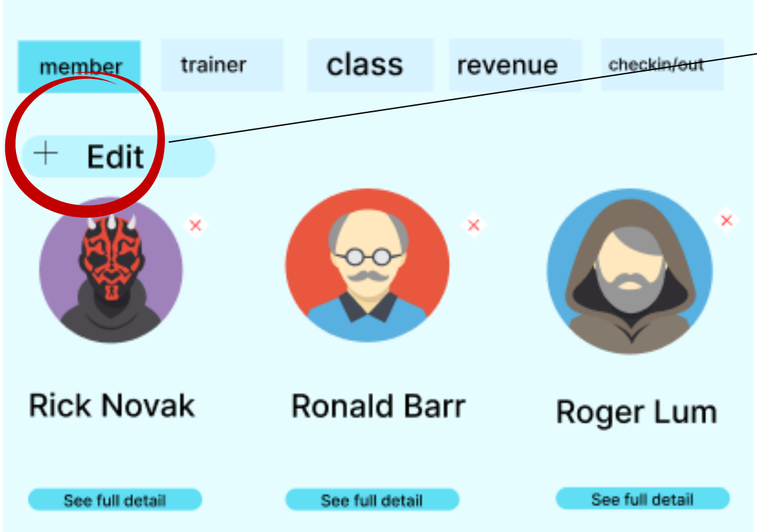

```
DROP FUNCTION IF EXISTS see_full_detail_member(INT);

CREATE OR REPLACE FUNCTION see_full_detail_member(input_member_id INT)
RETURNS TABLE (
    first_name VARCHAR,
    last_name VARCHAR,
    gender VARCHAR,
    age INT,
    email VARCHAR,
    membership_type VARCHAR,
    membership_expiry DATE

) AS $$
BEGIN
    RETURN QUERY
    SELECT
        m.first_name,
        m.last_name,
        m.gender,
        m.age,
        m.email,
        m.membership_type,
        m.membership_expiry
    FROM member m
    Where member_id = input_member_id;
END;
$$ LANGUAGE plpgsql;
```




```
Select see_full_detail_member( input_member_id 2);
```

see_full_detail_member

```
(Rick,Novak,Male,18,RickNovak@gmail.com,Monthly,2025-12-01)
```




```sql
DROP PROCEDURE IF EXISTS edit_member_detail(INT,VARCHAR, VARCHAR, VARCHAR, INT, VARCHAR, VARCHAR, DATE);

CREATE OR REPLACE PROCEDURE edit_member_detail(
    input_member_id INT,
    input_first_name VARCHAR,
    input_last_name VARCHAR,
    input_gender VARCHAR,
    input_age INT,
    input_email VARCHAR,
    input_type VARCHAR,
    input_expiry DATE
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE member
    SET
        first_name = input_first_name,
        last_name = input_last_name,
        gender = input_gender,
        age = input_age,
        email = input_email,
        membership_type = input_type,
        membership_expiry = input_expiry
    WHERE member_id = input_member_id;
end;
$$;

CALL edit_member_detail(2, 'Rick', 'Novak', 'Male', 18, 'RickNovak@gmail.com', 'Monthly', '2025-12-02')
```

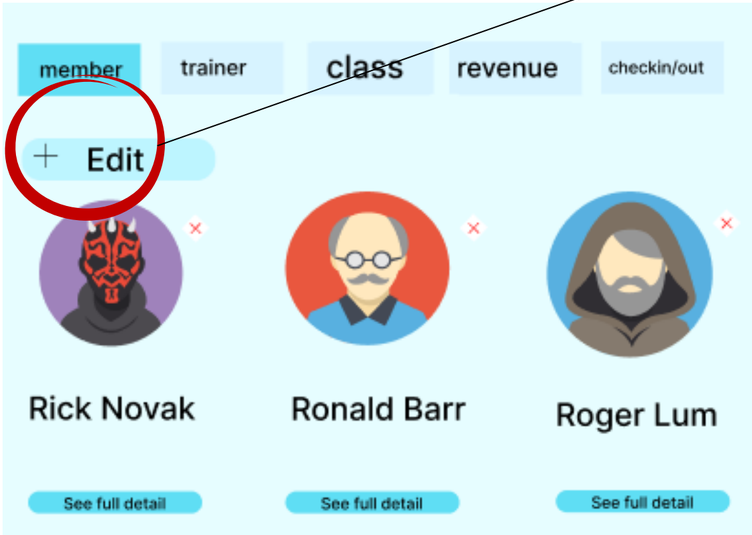
Before

```
see_full_detail_member
1 (Rick,Novak,Male,18,RickNovak@gmail.com,Monthly,2025-12-01)
```

```sql
call edit_member_detail( input_member_id 2, input_first_name 'Rick', input_last_name 'Novak',
    input_gender 'Male', input_age 20, input_email 'RickNovak@gmail.com', input_type 'Yearly',
    input_expiry '2026-01-01');
```

After

```
see_full_detail_member
1 (Rick,Novak,Male,20,RickNovak@gmail.com,Yearly,2026-01-01)
```




```sql
DROP PROCEDURE IF EXISTS add_member(VARCHAR, VARCHAR, VARCHAR, INT, VARCHAR, VARCHAR, DATE);

CREATE OR REPLACE PROCEDURE add_member (
    input_first_name VARCHAR,
    input_last_name VARCHAR,
    input_gender VARCHAR,
    input_age INT,
    input_email VARCHAR,
    input_type VARCHAR,
    input_expiry DATE
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO member(
        first_name, last_name, gender, age, email, membership_type, membership_expiry
    )
    VALUES(
           input_first_name, input_last_name, input_gender,
           input_age, input_email, input_type, input_expiry
          );
end;
$$;
```

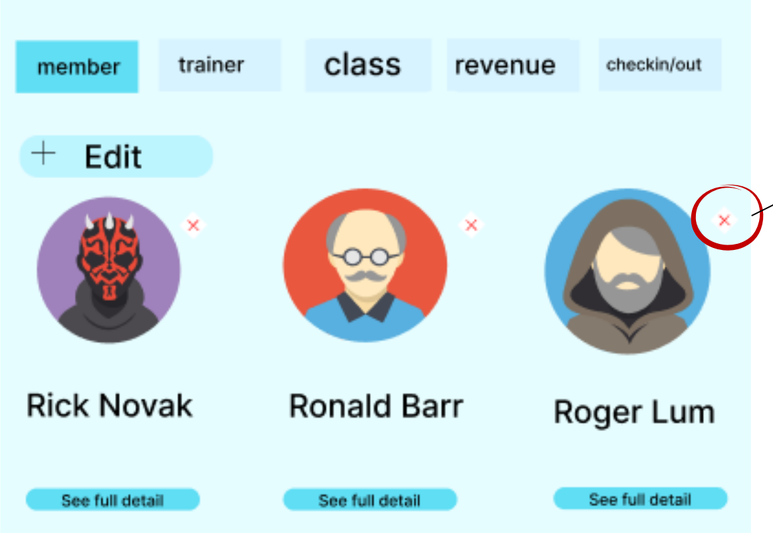
Before

	member_page
1	(1,Emily,Johnson)
2	(2,Rick,Novak)
3	(3,Susan,Connor)
4	(4,Margaret,Adelman)
5	(5,Ronald,Barr)
6	(6,Marie,Broadbet)
7	(7,Roger,Lum)
8	(8,Marlene,Donahue)
9	(9,Jeff,Johnson)
10	(10,Melvin,Forbis)

After

```sql
call add_member( input_first_name 'James', input_last_name 'Smith', input_gender 'Male', input_age 40,
 input_email 'JamesSmith@gmail.com', input_type 'Monthly', input_expiry '2025-12-30');
```

	member_page
1	(1,Emily,Johnson)
2	(2,Rick,Novak)
3	(3,Susan,Connor)
4	(4,Margaret,Adelman)
5	(5,Ronald,Barr)
6	(6,Marie,Broadbet)
7	(7,Roger,Lum)
8	(8,Marlene,Donahue)
9	(9,Jeff,Johnson)
10	(10,Melvin,Forbis)
11	(13,James,Smith)




```
DROP PROCEDURE IF EXISTS delete_member(INT);

CREATE OR REPLACE PROCEDURE delete_member(input_member_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM checkin_log
    WHERE member_id = input_member_id;

    DELETE FROM purchase_history
    WHERE member_id = input_member_id;

    DELETE FROM member
    WHERE member_id = input_member_id;
end;
$$;

CALL delete_member(11);
```

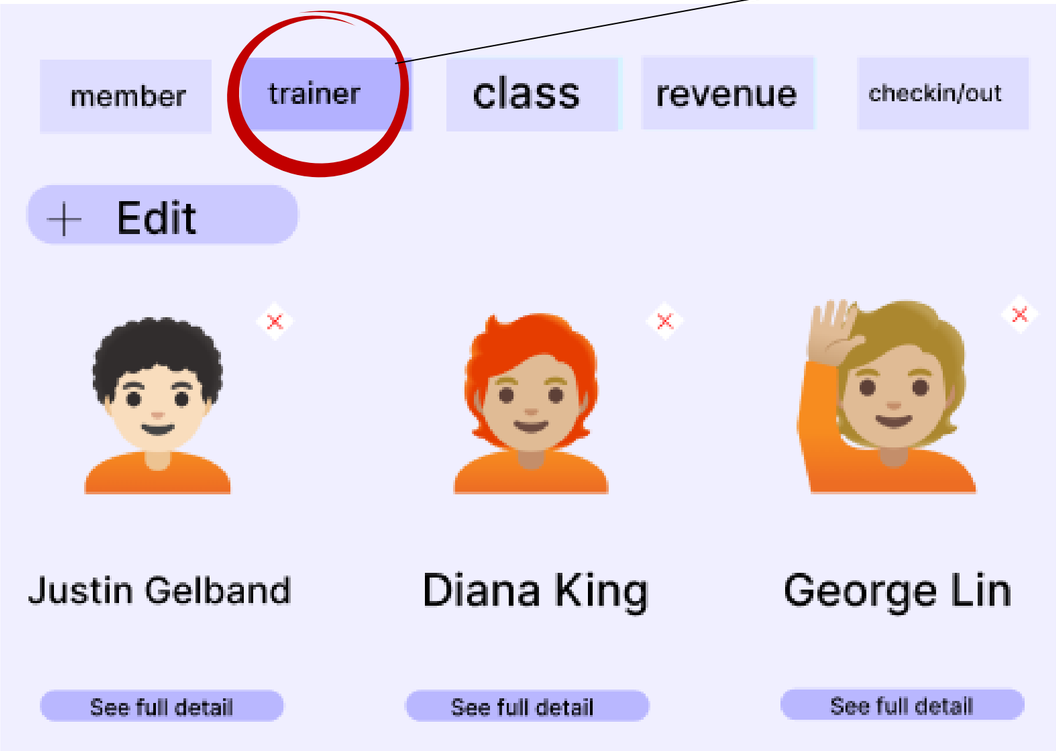
Before

	member_page
1	(1,Emily,Johnson)
2	(2,Rick,Novak)
3	(3,Susan,Connor)
4	(4,Margaret,Adelman)
5	(5,Ronald,Barr)
6	(6,Marie,Broadbet)
7	(7,Roger,Lum)
8	(8,Marlene,Donahue)
9	(9,Jeff,Johnson)
10	(10,Melvin,Forbis)
11	(13,James,Smith)

After

```
call delete_member( input_member_id 2);
```

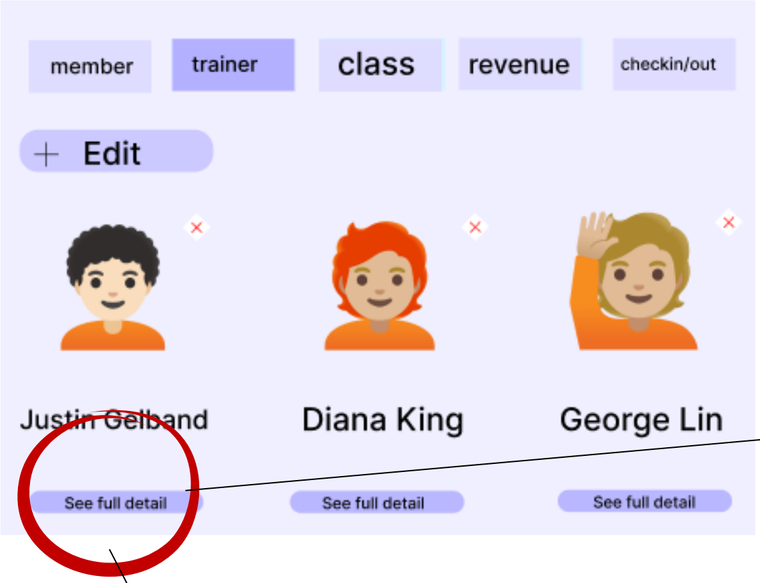
	member_page
1	(1,Emily,Johnson)
2	(3,Susan,Connor)
3	(4,Margaret,Adelman)
4	(5,Ronald,Barr)
5	(6,Marie,Broadbet)
6	(7,Roger,Lum)
7	(8,Marlene,Donahue)
8	(9,Jeff,Johnson)
9	(10,Melvin,Forbis)
10	(13,James,Smith)



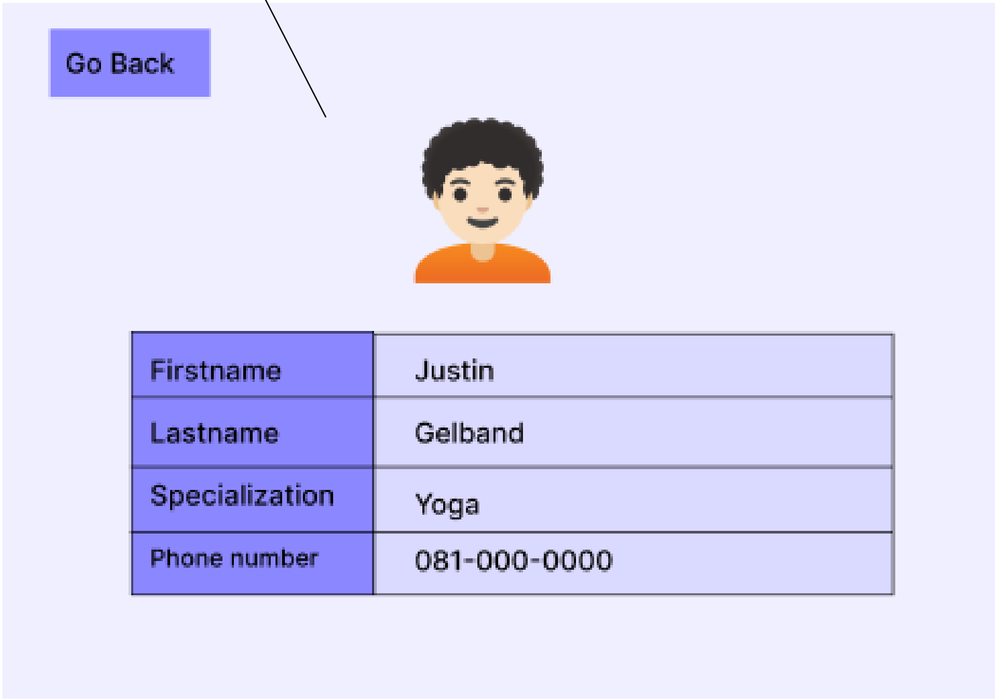

```sql
DROP FUNCTION IF EXISTS trainer_page();

CREATE OR REPLACE FUNCTION trainer_page()
RETURNS TABLE(
    input_trainer_id INT,
    first_name VARCHAR,
    last_name VARCHAR
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        t.trainer_id,
        t.first_name,
        t.last_name
    FROM trainer t
    ORDER BY t.trainer_id ASC;
end;
$$ LANGUAGE plpgsql;


SELECT trainer_page()
```

```sql
select trainer_page();
```

	trainer_page
1	(1,Justin,Gelband)
2	(2,Aaron,Williamson)
3	(3,Alex,Tan)
4	(4,Bella,Fox)
5	(5,Chris,Wong)
6	(6,Diana,King)
7	(7,Ethan,Park)
8	(8,Fiona,Chu)
9	(9,George,Lin)
10	(10,Hannah,Cho)

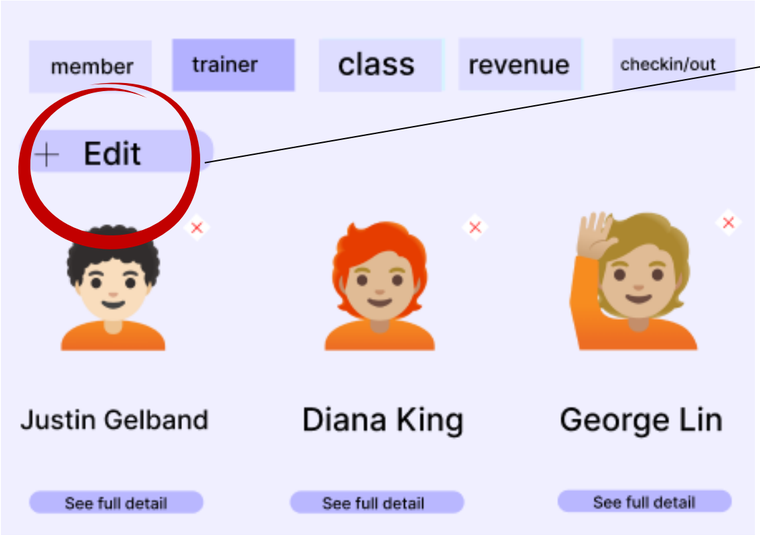



```
DROP FUNCTION IF EXISTS see_full_detail_trainer(INT);

CREATE OR REPLACE FUNCTION see_full_detail_trainer(input_trainer_id INT)
RETURNS TABLE(
    first_name VARCHAR,
    last_name VARCHAR,
    specialization VARCHAR,
    phone VARCHAR
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        t.first_name,
        t.last_name,
        t.specialization,
        t.phone
    FROM trainer t
    WHERE t.trainer_id = input_trainer_id;
end;

$$ LANGUAGE plpgsql;


SELECT see_full_detail_trainer(2)
```




```
select see_full_detail_trainer( input_trainer_id 1);
```

	see_full_detail_trainer
1	(Justin,Gelband,"Yoga instructor",081-000-0000)




```
DROP PROCEDURE IF EXISTS edit_trainer_detail(INT, VARCHAR, VARCHAR, VARCHAR, VARCHAR);

CREATE OR REPLACE PROCEDURE edit_trainer_detail(
    input_trainer_id INT,
    input_first_name VARCHAR,
    input_last_name VARCHAR,
    input_specialization VARCHAR,
    input_phone VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE trainer
    SET
        first_name = input_first_name,
        last_name = input_last_name,
        specialization = input_specialization,
        phone = input_phone
    WHERE trainer_id = input_trainer_id;
end;

$$;
```

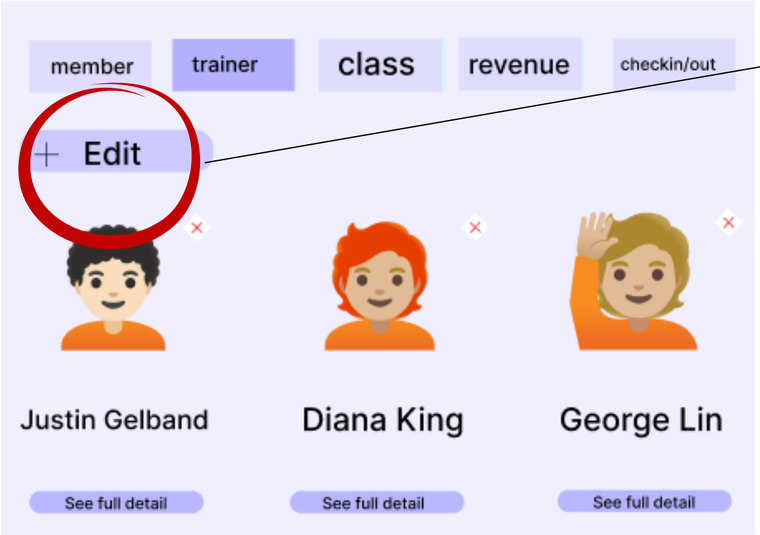
Before

```
see_full_detail_trainer
1 (Justin,Gelband,"Yoga instructor",081-000-0000)
```

After

```
call edit_trainer_detail( input_trainer_id 1, input_first_name 'Justin', input_last_name 'Gelband',
    input_specialization 'Strength training instructor', input_phone '081-123-1234');
```

```
see_full_detail_trainer
1 (Justin,Gelband,"Strength training instructor",081-123-1234)
```




```
DROP PROCEDURE IF EXISTS add_trainer(VARCHAR, VARCHAR, VARCHAR, VARCHAR);

CREATE OR REPLACE PROCEDURE add_trainer(
    input_first_name VARCHAR,
    input_last_name VARCHAR,
    input_specialization VARCHAR,
    input_phone VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO trainer(first_name, last_name, specialization, phone)
    VALUES(input_first_name, input_last_name,
           input_specialization, input_phone);
end;

$$;
```

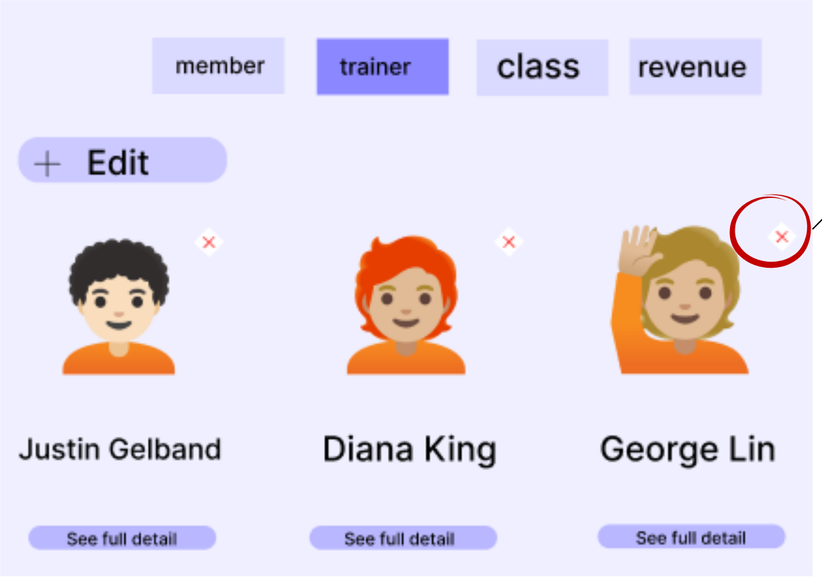
Before

	trainer_page
1	(1,Justin,Gelband)
2	(2,Aaron,Williamson)
3	(3,Alex,Tan)
4	(4,Bella,Fox)
5	(5,Chris,Wong)
6	(6,Diana,King)
7	(7,Ethan,Park)
8	(8,Fiona,Chu)
9	(9,George,Lin)
10	(10,Hannah,Cho)

After

```
call add_trainer( input_first_name 'Elizabeth', input_last_name 'Olivia', input_specialization 'Sprinting',
      input_phone '081-987-6543');
```

	trainer_page
1	(1,Justin,Gelband)
2	(2,Aaron,Williamson)
3	(3,Alex,Tan)
4	(4,Bella,Fox)
5	(5,Chris,Wong)
6	(6,Diana,King)
7	(7,Ethan,Park)
8	(8,Fiona,Chu)
9	(9,George,Lin)
10	(10,Hannah,Cho)
11	(11,Elizabeth,Olivia)




```
DROP PROCEDURE IF EXISTS delete_trainer(INT);

CREATE OR REPLACE PROCEDURE delete_trainer(input_trainer_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM class
    WHERE trainer_id = input_trainer_id;

    DELETE FROM trainer
    WHERE trainer_id = input_trainer_id;
end;

$$;
```

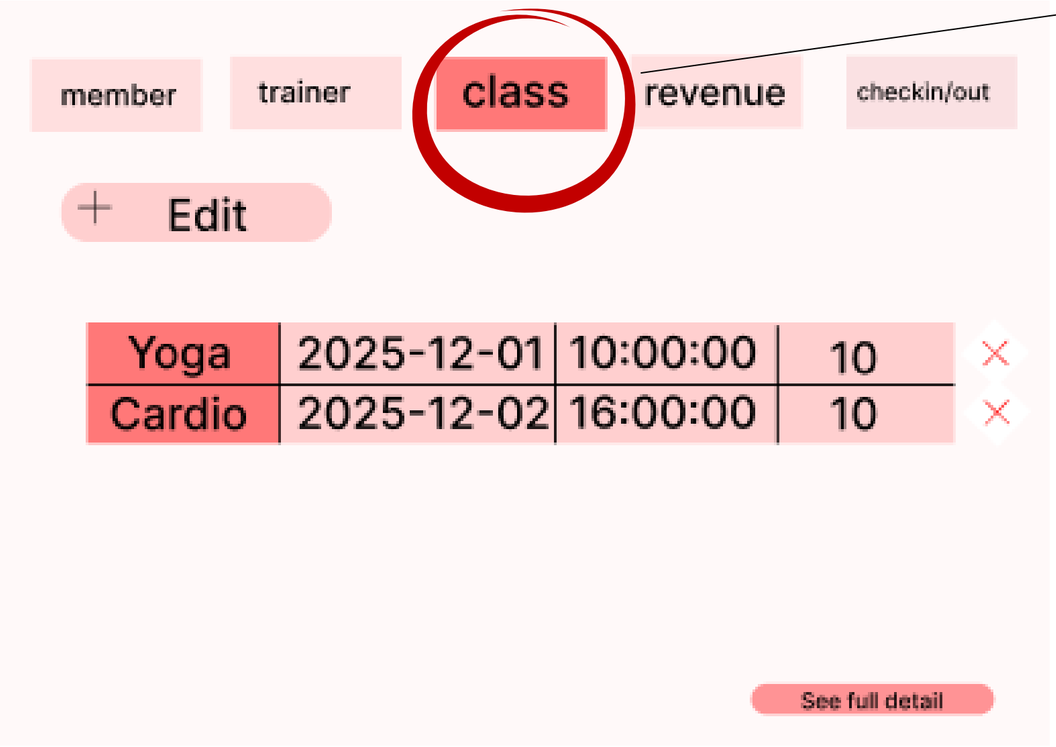
Before

	trainer_page
1	(1,Justin,Gelband)
2	(2,Aaron,Williamson)
3	(3,Alex,Tan)
4	(4,Bella,Fox)
5	(5,Chris,Wong)
6	(6,Diana,King)
7	(7,Ethan,Park)
8	(8,Fiona,Chu)
9	(9,George,Lin)
10	(10,Hannah,Cho)
11	(11,Elizabeth,Olivia)

After

```
call delete_trainer( input_trainer_id 2);
```

	trainer_page
1	(1,Justin,Gelband)
2	(3,Alex,Tan)
3	(4,Bella,Fox)
4	(5,Chris,Wong)
5	(6,Diana,King)
6	(7,Ethan,Park)
7	(8,Fiona,Chu)
8	(9,George,Lin)
9	(10,Hannah,Cho)
10	(11,Elizabeth,Olivia)




```
DROP FUNCTION IF EXISTS class_page;

CREATE OR REPLACE FUNCTION class_page()
RETURNS TABLE(
    class_id INT,
    class_name VARCHAR,
    class_date DATE,
    class_time TIME,
    capacity INT
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        c.class_id,
        c.class_name,
        c.schedule::DATE,
        c.schedule::TIME,
        c.capacity
    FROM class c
    ORDER BY c.schedule ASC;
end;
$$ LANGUAGE plpgsql;

SELECT * FROM class_page();
```

```
select class_page();
```

	class_page
1	(1,"Evening Yoga",2025-12-01,18:00:00,10)
2	(9,"Crossfit Intro",2025-12-02,07:00:00,10)
3	(11,Sprinting,2025-12-02,08:00:00,10)
4	(5,"Pilates Core",2025-12-02,09:00:00,10)
5	(7,"Cardio Blast",2025-12-02,11:00:00,10)
6	(4,"Zumba Dance",2025-12-02,14:00:00,10)
7	(2,"HIIT Burn",2025-12-02,16:00:00,10)
8	(10,"Dance Fit",2025-12-02,17:00:00,10)
9	(6,"Boxing Basics",2025-12-02,18:00:00,10)
10	(8,"Stretch & Relax",2025-12-02,19:00:00,10)

member | trainer | class | revenue | checkin/out

+ Edit

Yoga	2025-12-01	10:00:00	10
Cardio	2025-12-02	16:00:00	10

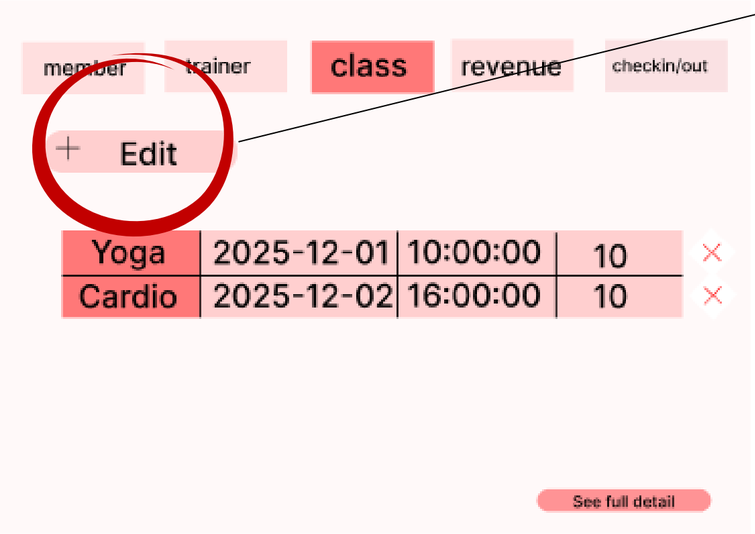
See full detail

```
DROP FUNCTION IF EXISTS see_full_detail_class(INT);

CREATE OR REPLACE FUNCTION see_full_detail_class(input_class_id INT)
RETURNS TABLE(
    class_name VARCHAR,
    trainer_first_name VARCHAR,
    trainer_last_name VARCHAR,
    schedule TIMESTAMP,
    capacity INT,
    trainer_phone VARCHAR
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        c.class_name,
        t.first_name,
        t.last_name,
        c.schedule,
        c.capacity,
        t.phone
    FROM Class c
    JOIN Trainer t ON c.trainer_id = t.trainer_id
    WHERE c.class_id = input_class_id;
end;
$$ LANGUAGE plpgsql;


SELECT * FROM see_full_detail_class(2)
```

```
select see_full_detail_class( input_class_id 1);
```

see_full_detail_class

```
1 ("Evening Yoga",Justin,Gelband,"2025-12-01 18:00:00",10,081-123-1234)
```




```sql
DROP PROCEDURE IF EXISTS add_class(VARCHAR, INT, TIMESTAMP, INT);

CREATE OR REPLACE PROCEDURE add_class(
    input_class_name VARCHAR,
    input_trainer_id INT,
    input_schedule TIMESTAMP,
    input_capacity INT
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO Class(class_name, trainer_id, schedule, capacity)
    VALUES(input_class_name, input_trainer_id, input_schedule,
           input_capacity);
end;
$$;


CALL add_class('Sprinting', '10', '2025-12-02 08:00:00',
     10)
```

Before

class_page

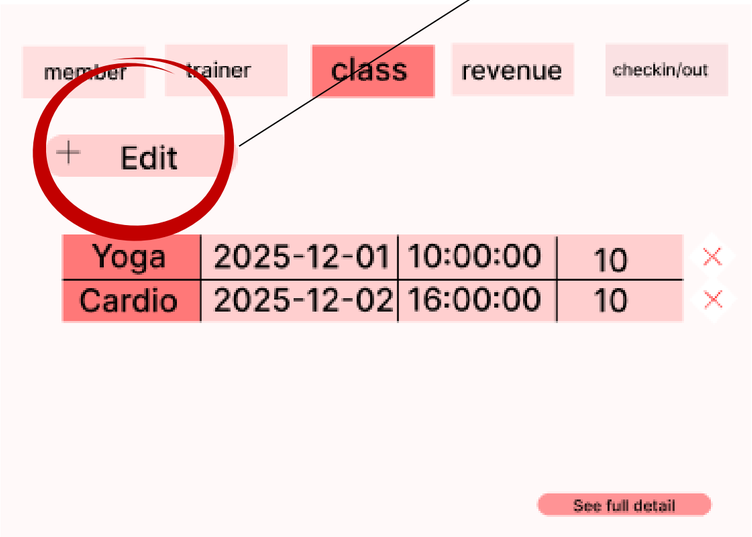
```
(1,"Evening Yoga",2025-12-01,18:00:00,10)
(9,"Crossfit Intro",2025-12-02,07:00:00,10)
(11,Sprinting,2025-12-02,08:00:00,10)
(5,"Pilates Core",2025-12-02,09:00:00,10)
(7,"Cardio Blast",2025-12-02,11:00:00,10)
(4,"Zumba Dance",2025-12-02,14:00:00,10)
(2,"HIIT Burn",2025-12-02,16:00:00,10)
(10,"Dance Fit",2025-12-02,17:00:00,10)
(6,"Boxing Basics",2025-12-02,18:00:00,10)
(8,"Stretch & Relax",2025-12-02,19:00:00,10)
```

After

```sql
call add_class( input_class_name 'Jumping rope', input_trainer_id 3, input_schedule '2025-12-04 15:00:00',
                input_capacity 10);
```

class_page

```
(1,"Evening Yoga",2025-12-01,18:00:00,10)
(9,"Crossfit Intro",2025-12-02,07:00:00,10)
(11,Sprinting,2025-12-02,08:00:00,10)
(5,"Pilates Core",2025-12-02,09:00:00,10)
(7,"Cardio Blast",2025-12-02,11:00:00,10)
(4,"Zumba Dance",2025-12-02,14:00:00,10)
(2,"HIIT Burn",2025-12-02,16:00:00,10)
(10,"Dance Fit",2025-12-02,17:00:00,10)
(6,"Boxing Basics",2025-12-02,18:00:00,10)
(8,"Stretch & Relax",2025-12-02,19:00:00,10)
(13,"Jumping rope",2025-12-04,15:00:00,10)
```




```
DROP PROCEDURE IF EXISTS edit_class_detail(INT, VARCHAR, INT, TIMESTAMP, INT);

CREATE OR REPLACE PROCEDURE edit_class_detail(
    input_class_id INT,
    input_class_name VARCHAR,
    input_trainer_id INT,
    input_schedule TIMESTAMP,
    input_capacity INT
)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE class
    SET
        class_name = input_class_name,
        trainer_id = input_trainer_id,
        schedule = input_schedule,
        capacity = input_capacity
    WHERE class_id = input_class_id;
end;
$$;


CALL edit_class_detail(1, 'Evening Yoga', 1,
     '2025-12-01 18:00:00', 10)
```

Before

class_page
(1,"Evening Yoga",2025-12-01,18:00:00,10)
(9,"Crossfit Intro",2025-12-02,07:00:00,10)
(11,Sprinting,2025-12-02,08:00:00,10)
(5,"Pilates Core",2025-12-02,09:00:00,10)
(7,"Cardio Blast",2025-12-02,11:00:00,10)
(4,"Zumba Dance",2025-12-02,14:00:00,10)
(2,"HIIT Burn",2025-12-02,16:00:00,10)
(10,"Dance Fit",2025-12-02,17:00:00,10)
(6,"Boxing Basics",2025-12-02,18:00:00,10)
(8,"Stretch & Relax",2025-12-02,19:00:00,10)
(13,"Jumping rope",2025-12-04,15:00:00,10)

After

```
call edit_class_detail( input_class_id 13, input_class_name 'Jumping rope', input_trainer_id 3,
     input_schedule '2025-12-05 15:00:00', input_capacity 15);
```

class_page
(9,"Crossfit Intro",2025-12-02,07:00:00,10)
(11,Sprinting,2025-12-02,08:00:00,10)
(5,"Pilates Core",2025-12-02,09:00:00,10)
(7,"Cardio Blast",2025-12-02,11:00:00,10)
(4,"Zumba Dance",2025-12-02,14:00:00,10)
(2,"HIIT Burn",2025-12-02,16:00:00,10)
(10,"Dance Fit",2025-12-02,17:00:00,10)
(6,"Boxing Basics",2025-12-02,18:00:00,10)
(8,"Stretch & Relax",2025-12-02,19:00:00,10)
(13,"Jumping rope",2025-12-05,15:00:00,15)

```
DROP PROCEDURE IF EXISTS delete_class(INT);

CREATE OR REPLACE PROCEDURE delete_class(input_class_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM class
    WHERE class_id = input_class_id;
end;

$$
```

Before

	class_page
2	(9,"Crossfit Intro",2025-12-02,07:00:00,10)
3	(11,Sprinting,2025-12-02,08:00:00,10)
4	(5,"Pilates Core",2025-12-02,09:00:00,10)
5	(7,"Cardio Blast",2025-12-02,11:00:00,10)
6	(4,"Zumba Dance",2025-12-02,14:00:00,10)
7	(2,"HIIT Burn",2025-12-02,16:00:00,10)
8	(10,"Dance Fit",2025-12-02,17:00:00,10)
9	(6,"Boxing Basics",2025-12-02,18:00:00,10)
10	(8,"Stretch & Relax",2025-12-02,19:00:00,10)
11	(13,"Jumping rope",2025-12-05,15:00:00,15)

After

```
call delete_class( input_class_id 11);
```

	class_page
2	(9,"Crossfit Intro",2025-12-02,07:00:00,10)
3	(5,"Pilates Core",2025-12-02,09:00:00,10)
4	(7,"Cardio Blast",2025-12-02,11:00:00,10)
5	(4,"Zumba Dance",2025-12-02,14:00:00,10)
6	(2,"HIIT Burn",2025-12-02,16:00:00,10)
7	(10,"Dance Fit",2025-12-02,17:00:00,10)
8	(6,"Boxing Basics",2025-12-02,18:00:00,10)
9	(8,"Stretch & Relax",2025-12-02,19:00:00,10)
10	(13,"Jumping rope",2025-12-05,15:00:00,15)

member | trainer | class | revenue | ~~checkin/out~~

Total 55,150 baht

Full Name	Product	Price	Date
Rick Novak	Protein Powder	300	2025-11-01
Susan Connor	Monthly member	1200	2025-11-03
Ronald Barr	Monthly member	1200	2025-11-05
Roger Lum	Yearly member	10000	2025-11-07
Melvis Forbis	Yearly member	10000	2025-11-10

+ New

See full detail

```sql
DROP FUNCTION IF EXISTS revenue_page();

CREATE OR REPLACE FUNCTION revenue_page()
RETURNS TABLE(
    input_purchase_id INT,
    first_name VARCHAR,
    last_name VARCHAR,
    item_type VARCHAR,
    price DECIMAL(10,2),
    purchase_date DATE
)  AS $$
BEGIN
    RETURN QUERY
    SELECT
        p.purchase_id,
        m.first_name,
        m.last_name,
        p.item_type,
        p.price,
        p.purchase_date
    FROM purchase_history p
    JOIN member m ON p.member_id = m.member_id
    ORDER BY p.purchase_date DESC;
end;

$$ LANGUAGE plpgsql;


SELECT * FROM revenue_page()
```

```sql
select revenue_page();
```

	revenue_page
1	(12,Roger,Lum,"Energy Drink",50.00,2025-11-07)
2	(11,Marie,Broadbet,"Protein Powder",300.00,2025-11-06)
3	(5,Ronald,Barr,"Monthly Membership",1200.00,2025-11-05)
4	(4,Margaret,Adelman,"Monthly Membership",1200.00,2025-11-04)
5	(3,Susan,Connor,"Monthly Membership",1200.00,2025-11-03)
6	(1,Emily,Johnson,"Monthly Membership",1200.00,2025-11-01)
7	(10,Melvin,Forbis,"Yearly Membership",10000.00,2024-12-10)
8	(9,Jeff,Johnson,"Yearly Membership",10000.00,2024-12-09)
9	(8,Marlene,Donahue,"Yearly Membership",10000.00,2024-12-08)
10	(7,Roger,Lum,"Yearly Membership",10000.00,2024-12-07)
11	(6,Marie,Broadbet,"Yearly Membership",10000.00,2024-12-06)

member | trainer | class | revenue | checkin/out

Total 55,150 baht

Full Name	Product	Price	Date
Rick Novak	Protein Powder	300	2025-11-01
Susan Connor	Monthly member	1200	2025-11-03
Ronald Barr	Monthly member	1200	2025-11-05
Roger Lum	Yearly member	10000	2025-11-07
Melvis Forbis	Yearly member	10000	2025-11-10

+ New

See full detail

```
DROP FUNCTION IF EXISTS total_revenue();

CREATE OR REPLACE FUNCTION total_revenue()
RETURNS DECIMAL(10,2) AS $$
DECLARE
    total_sales DECIMAL(10,2);
BEGIN
    SELECT SUM(price) INTO total_sales
    FROM purchase_history;

    IF total_sales IS NULL THEN
        RETURN 0.00;
    ELSE
        RETURN total_sales;
    end if;
end;

$$ LANGUAGE plpgsql;

SELECT total_revenue()
```

```
select total_revenue()
```

	total_revenue
1	55150

member | trainer | class | revenue | checkin/out

Total 55,150 baht

Full Name	Product	Price	Date
Rick Novak	Protein Powder	300	2025-11-01
Susan Connor	Monthly member	1200	2025-11-03
Ronald Barr	Monthly member	1200	2025-11-05
Roger Lum	Yearly member	10000	2025-11-07
Melvis Forbis	Yearly member	10000	2025-11-10

+ New

See full detail

```sql
DROP PROCEDURE IF EXISTS add_revenue(INT, VARCHAR, DECIMAL, DATE, DATE);

CREATE OR REPLACE PROCEDURE add_revenue(
    input_member_id INT,
    input_item_type VARCHAR,
    input_price DECIMAL,
    input_purchase_date DATE,
    input_expiry_date DATE
)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO purchase_history(member_id, item_type, price, purchase_date, expiry_date)
    VALUES (input_member_id, input_item_type, input_price,
            input_purchase_date, input_expiry_date);
end;

$$;
```

After

```sql
call add_revenue( input_member_id 13, input_item_type 'Protein bar', input_price 50,
    input_purchase_date '2025-11-08', input_expiry_date Null);
```

Before

```
revenue_page
(12,Roger,Lum,"Energy Drink",50.00,2025-11-07)
(11,Marie,Broadbet,"Protein Powder",300.00,2025-11-06)
(5,Ronald,Barr,"Monthly Membership",1200.00,2025-11-05)
(4,Margaret,Adelman,"Monthly Membership",1200.00,2025-11-04)
(3,Susan,Connor,"Monthly Membership",1200.00,2025-11-03)
(1,Emily,Johnson,"Monthly Membership",1200.00,2025-11-01)
(10,Melvin,Forbis,"Yearly Membership",10000.00,2024-12-10)
(9,Jeff,Johnson,"Yearly Membership",10000.00,2024-12-09)
(8,Marlene,Donahue,"Yearly Membership",10000.00,2024-12-08)
(7,Roger,Lum,"Yearly Membership",10000.00,2024-12-07)
(6,Marie,Broadbet,"Yearly Membership",10000.00,2024-12-06)
```

```
revenue_page
(16,James,Smith,"Protein bar",50.00,2025-11-08)
(12,Roger,Lum,"Energy Drink",50.00,2025-11-07)
(11,Marie,Broadbet,"Protein Powder",300.00,2025-11-06)
(5,Ronald,Barr,"Monthly Membership",1200.00,2025-11-05)
(4,Margaret,Adelman,"Monthly Membership",1200.00,2025-11-04)
(3,Susan,Connor,"Monthly Membership",1200.00,2025-11-03)
(1,Emily,Johnson,"Monthly Membership",1200.00,2025-11-01)
(10,Melvin,Forbis,"Yearly Membership",10000.00,2024-12-10)
(9,Jeff,Johnson,"Yearly Membership",10000.00,2024-12-09)
(8,Marlene,Donahue,"Yearly Membership",10000.00,2024-12-08)
(7,Roger,Lum,"Yearly Membership",10000.00,2024-12-07)
```

member | trainer | class | revenue | checkin/out

Total 55,150 baht

Full Name	Product	Price	Date
Rick Novak	Protein Powder	300	2025-11-01
Susan Connor	Monthly member	1200	2025-11-03
Ronald Barr	Monthly member	1200	2025-11-05
Roger Lum	Yearly member	10000	2025-11-07
Melvis Forbis	Yearly member	10000	2025-11-10

+ New

See full detail

```sql
DROP FUNCTION IF EXISTS  see_full_detail_revenue(INT);

CREATE OR REPLACE FUNCTION see_full_detail_revenue(input_purchase_id INT)
RETURNS TABLE(
    purchase_id INT,
    first_name VARCHAR,
    last_name VARCHAR,
    item_type VARCHAR,
    price DECIMAL,
    purchase_date DATE,
    expiry_date DATE,
    email VARCHAR
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        p.purchase_id,
        m.first_name,
        m.last_name,
        p.item_type,
        p.price,
        p.purchase_date,
        p.expiry_date,
        m.email
    FROM purchase_history p
    JOIN member m ON p.member_id = m.member_id
    WHERE p.purchase_id = input_purchase_id;
end;

$$ LANGUAGE plpgsql;


SELECT see_full_detail_revenue(1)
```

```sql
select see_full_detail_revenue( input_purchase_id 16);
```

	see_full_detail_revenue
1	(16,James,Smith,"Protein bar",50.00,2025-11-08,,JamesSmith@gmail.com)

member | trainer | class | revenue | checkin/out

Total 55,150 baht

Full Name	Product	Price	Date
Rick Novak	Protein Powder	300	2025-11-01
Susan Connor	Monthly member	1200	2025-11-03
Ronald Barr	Monthly member	1200	2025-11-05
Roger Lum	Yearly member	10000	2025-11-07
Melvis Forbis	Yearly member	10000	2025-11-10

+ New

See full detail

```
DROP PROCEDURE IF EXISTS delete_revenue(INT);

CREATE OR REPLACE PROCEDURE delete_revenue(input_purchase_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM purchase_history
    WHERE purchase_id = input_purchase_id;
end;
$$;
```

Before

```
revenue_page
(16,James,Smith,"Protein bar",50.00,2025-11-08)
(12,Roger,Lum,"Energy Drink",50.00,2025-11-07)
(11,Marie,Broadbet,"Protein Powder",300.00,2025-11-06)
(5,Ronald,Barr,"Monthly Membership",1200.00,2025-11-05)
(4,Margaret,Adelman,"Monthly Membership",1200.00,2025-11-04)
(3,Susan,Connor,"Monthly Membership",1200.00,2025-11-03)
(1,Emily,Johnson,"Monthly Membership",1200.00,2025-11-01)
(10,Melvin,Forbis,"Yearly Membership",10000.00,2024-12-10)
(9,Jeff,Johnson,"Yearly Membership",10000.00,2024-12-09)
(8,Marlene,Donahue,"Yearly Membership",10000.00,2024-12-08)
(7,Roger,Lum,"Yearly Membership",10000.00,2024-12-07)
```

After

```
call delete_revenue( input_purchase_id 12);
```

```
revenue_page
(16,James,Smith,"Protein bar",50.00,2025-11-08)
(11,Marie,Broadbet,"Protein Powder",300.00,2025-11-06)
(5,Ronald,Barr,"Monthly Membership",1200.00,2025-11-05)
(4,Margaret,Adelman,"Monthly Membership",1200.00,2025-11-04)
(3,Susan,Connor,"Monthly Membership",1200.00,2025-11-03)
(1,Emily,Johnson,"Monthly Membership",1200.00,2025-11-01)
(10,Melvin,Forbis,"Yearly Membership",10000.00,2024-12-10)
(9,Jeff,Johnson,"Yearly Membership",10000.00,2024-12-09)
(8,Marlene,Donahue,"Yearly Membership",10000.00,2024-12-08)
(7,Roger,Lum,"Yearly Membership",10000.00,2024-12-07)
(6,Marie,Broadbet,"Yearly Membership",10000.00,2024-12-06)
```

member | trainer | class | revenue | checkin/out

Firstname	Lastname	Checkin	Checkout
Ronald	Barr	2025-11-19 19:00:00	2025-11-19 20:00:00
Margaret	Adelman	2025-11-17 17:00:00	2025-11-17 18:00:00
Rick	Novak	2025-11-13 13:00:00	2025-11-13 14:00:00

+ New

```
DROP FUNCTION IF EXISTS see_full_detail_checkin_checkout();

CREATE OR REPLACE FUNCTION see_full_detail_checkin_checkout()
RETURNS TABLE(
    check_id INT,
    first_name VARCHAR,
    last_name VARCHAR,
    checkin_time TIMESTAMP,
    checkout_time TIMESTAMP
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        c.check_id,
        m.first_name,
        m.last_name,
        c.checkin_time,
        c.checkout_time
    FROM checkin_log c
    JOIN member m on c.member_id = m.member_id
    ORDER BY c.checkin_time DESC;

end;
$$ LANGUAGE plpgsql;

SELECT see_full_detail_checkin_checkout()
```

```
select see_full_detail_checkin_checkout();
```

	see_full_detail_checkin_checkout
1	(10,Jeff,Johnson,"2025-11-29 18:00:00","2025-11-29 19:00:00")
2	(9,Melvin,Forbis,"2025-11-27 14:00:00","2025-11-27 15:00:00")
3	(8,Marlene,Donahue,"2025-11-25 07:00:00","2025-11-25 08:00:00")
4	(6,Marie,Broadbet,"2025-11-21 09:00:00","2025-11-21 10:00:00")
5	(5,Margaret,Adelman,"2025-11-19 19:00:00","2025-11-19 20:00:00")
6	(4,Roger,Lum,"2025-11-17 17:00:00","2025-11-17 18:00:00")
7	(3,Ronald,Barr,"2025-11-15 15:00:00","2025-11-15 16:00:00")
8	(2,Susan,Connor,"2025-11-13 13:00:00","2025-11-13 14:00:00")
9	(1,Emily,Johnson,"2025-11-11 11:00:00","2025-11-11 12:00:00")

member | trainer | class | revenue | checkin/out

Firstname	Lastname	Checkin	Checkout
Ronald	Barr	2025-11-19 19:00:00	2025-11-19 20:00:00
Margaret	Adelman	2025-11-17 17:00:00	2025-11-17 18:00:00
Rick	Novak	2025-11-13 13:00:00	2025-11-13 14:00:00

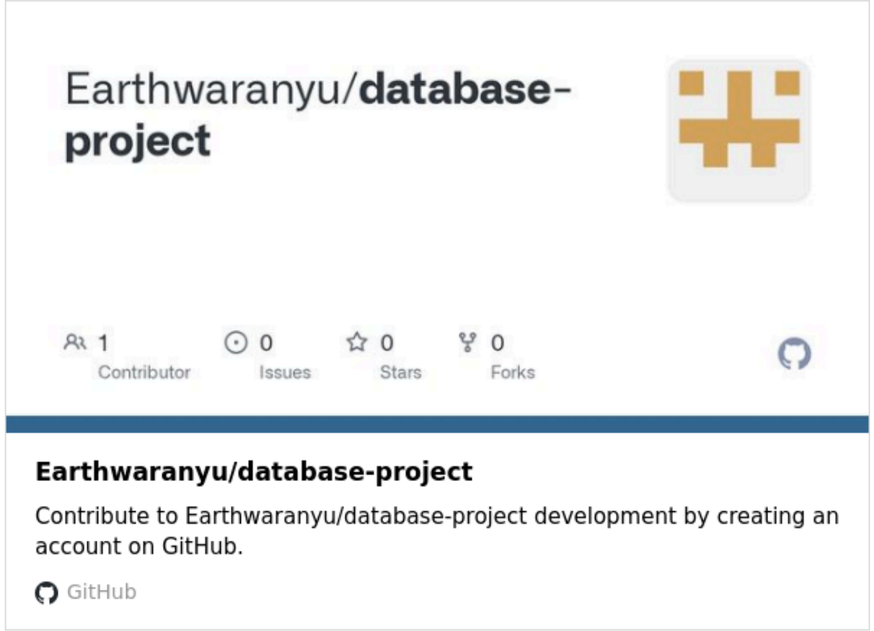
+ New

```
DROP PROCEDURE IF EXISTS check_in_member(INT);

CREATE OR REPLACE PROCEDURE check_in_member(input_member_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    INSERT INTO checkin_log (member_id, checkin_time, checkout_time)
    VALUES(input_member_id, NOW(), NULL);
end;

$$;


CALL check_in_member(2)
```

```
32  call check_in_member( input_member_id 2)
33
34
35
36
```

[P0001] ERROR: Access Denied: Rick Novak membership expired on 2025-12-01
Where: PL/pgSQL function check_expiry_before_entry() line 13 at RAISE
SQL statement "INSERT INTO checkin_log (member_id, checkin_time, checkout_time)
VALUES(input_member_id, NOW(), NULL)"
PL/pgSQL function check_in_member(integer) line 3 at SQL statement

```
DROP PROCEDURE IF EXISTS check_out_member(INT);

CREATE OR REPLACE PROCEDURE check_out_member(input_member_id INT)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE checkin_log
    SET checkout_time = NOW()
    WHERE member_id = input_member_id AND checkout_time IS NULL;
end;

$$;

CALL check_out_member(2)
```



Waranyu Chunhachatcharachai 6681376

Git repo URL https://github.com/Earthwaranyu/database-project

External Database URL postgresql://gym_kp3q_user:q1PdtofuezTlPU8EnjiAOdIhhHmdY8Ko@dpg-d4jtk0buibrs73f3np40-a.singapore-postgres.render.com/gym_kp3q