Create a User

Creating a Branches

```
-- Adding Branches

INSERT INTO "Branches" ("BranchName", "BranchType", "Location", "PhoneNumber", "Email", "Facilities", "OpeningHours", "MembershipOptions", "Status")

VALUES

('Jett Fitness <u>Salaya</u>', 'Basic', '<u>Salaya</u>, Downtown', '02-789-6543', 'salaya@jettfitness.com', 'Gym', 'Mon-Fri 6am-10pm, Sat-Sun 8am-8pm', 'Monthly, Yearly',

('Jett Fitness Rama 2', 'Premium', 'Rama 2', '02-987-6544', 'rama2@jettfitness.com', 'Yoga Studio, Meditation Room, Pool, Spa', 'Mon-Fri 7am-9pm, Sat 9am-5p
```

	📭 BranchID	≎ ∭ BranchName ÷	☐ BranchType ‡	☐ Location ‡	☐ PhoneNumber ÷	□ Email ÷	☐ Facilities ÷	☐ OpeningHours
1		3 Downtown Fitness	Basic	123 Main St, Downtown		contact@downtownfitness.com	Gym, Pool, Spa	Mon-Fri 6am-10pm, Sat-Sun
2		4 Uptown fitness		456 High St, Uptown		info@uptownwellness.com	Yoga Studio, Meditation Room	Mon-Fri 7am-9pm, Sat 9am-9
3			Basic	Salaya, Downtown	02-789-6543	salaya@jettfitness.com		Mon-Fri 6am-10pm, Sat-Sun
4		6 Jett Fitness Rama 2	Premium	Rama 2		rama2@jettfitness.com	Yoga Studio, Meditation Room, Pool, Spa	Mon-Fri 7am-9pm, Sat 9am-5

Branches continue...

☐ MembershipOptions ÷	☐ ClassTypesOffered ÷	☐ EventsCalendarLink ÷	□ Status ÷
Monthly, Yearly	<null></null>	<null></null>	0pen
Monthly, Yearly, Pay per class	<null></null>	<null></null>	0pen
Monthly, Yearly	<null></null>	<null></null>	0pen
Monthly, Yearly	<null></null>	<null></null>	0pen

Creating an Instructor

```
-- Adding Instructors
INSERT INTO "Instructors" ("UserID", "BranchID", "InstructorName", "Specialization", "Rating", "HireDate", "ClassesTaught", "Availability", "Status", "YearsOfExperience")
VALUES
(14, 5, 'Cbum', 'Aerobics, Cardio', 4.8, '2023-05-15', 'Cardio Kickboxing', 'Weekdays 4pm-9pm', 'Active', 3);
```

Cbum is added

∏ InstructorID ÷	Ç UserID ≎	∏ BranchID ÷	☐ InstructorName ÷	□ Specialization ÷	□ Rating ÷	∭ HireDate ÷	☐ ClassesTaught ÷	∏ Availability ≎
	8		Jane Smith	Aerobics, Cardio	4.8	2023-05-15	Cardio Kickboxing	Weekdays 4pm-9pm
	9 11		Marry Smith	Cardio, Boxing		2024-03-20	Boxing	Weekdays 4pm-9pm
3 13	1 14		Cbum	Aerobics, Cardio	4.8	2023-05-15	Cardio Kickboxing	Weekdays 4pm-9pm

Creating an User

```
-- Adding Customers

INSERT INTO "Customers" ("UserID", "CustomerName", "Email",

"Address", "DateOfBirth", "Gender",

"EmergencyContact")

VALUES

(12, 'Mickey', 'mickey@gmail.com', 'Mahidol University', '2002-02-10', 'Male', '089-500-6296'),

(13, 'Bom', 'bom@gmail.com', 'Mahidol University', '2002-02-11', 'Male', '083-456-897');
```

Login into account

6 12 U	user1	password123	2024-03-28 09:00:00.000000	Active	Customer	false
7 13 U	user2	password123	2024-03-29 15:30:00.000000	Active	Customer	false

After login

```
-- After Creating account let's login first for Mickey and Bom SELECT user_login( username: 'user1', password: 'password123'); SELECT user_login( username: 'user2', password: 'password123');
```

The LoginStatus is True

7	13 user2	password123	2024-03-29 15:30:00.000000	Active	Customer	• true
6	12 user1	password123	2024-03-28 09:00:00.000000	Active	Customer	• true

Buying Membership

2024-03-28

11 Premium 2024-03-28

10 Basic

2025-03-26

2025-03-28

2025-03-28

100.00 Credit Card

100.00 Credit Card

200.00 Credit Card

☐ RenewalStatus ÷ ☐ NextPaymentDate ÷ ☐ AccessLevel ÷
□ RenewalStatus □ NextPaymentDate □ AccessLevel □
false 2024-04-25 basic
false 2024-04-26 Premium
• true 2024-04-28 Basic
• true 2024-04-28 Premium

We have also created a trigger to create transactions automatically after the users have purchased the membership.

	∏ TransactionID ÷	Ç CustomerID ÷	□ TransactionType		□ PaymentDate	□ Status ÷
1			Membership Purchase	100.00	2024-03-25	Completed
2			Product Purchase	100.00	2024-03-25	Completed
3			Product Purchase	50.00	2024-03-26	Completed
4			Membership Purchase	100.00	2024-03-26	Completed
5			Product Purchase	150.00	2024-03-26	Completed
6			Product Purchase	50.00	2024-03-26	Completed
7			Product Purchase	50.00	2024-03-26	Completed
8			Product Purchase	100.00	2024-03-27	Completed
9	10	8	Membership Purchase	100.00	2024-03-28	Completed
10	11		Membership Purchase	200.00	2024-03-28	Completed

If we don't login first then we cannot buy membership, purchased products or book classes. Let's use other customers as an example.

```
-- Buying membership without login

SELECT buy_membership( customer_id: 7, membership_type: 'Premium',

price: 200.00, payment_method: 'Credit Card',

access_level: 'Premium');

[P0001] ERROR: User must be logged in to perform this action.

Where: PL/pgSQL function buy_membership(integer,character varying,numeric,character varying,character varying) line 12 at RAISE
```

This is because we have created a condition for users to login to do anything within the app. **Let's create a class**

```
-- Creating a class

SELECT create_class( instructor_id: 11, class_name: 'Boxing', class_description: 'Learn Boxing Techniques', class_type: 'Boxing', duration: 60, difficulty_level: 'Hard', -

branch_id: 5, class_datetime: '2024-03-30 18:00:00', max_participants: 20);
```

	ClassID ÷	ClassName	☐ ClassDescription ÷	☐ ClassType ‡	☐ InstructorID ÷	☐ Duration ÷	☐ DifficultyLevel ÷
1		Yoga Class	Relax and unwind with yoga poses.	Yoga		60	Beginner
2	13	Body Combat	Body Combat	Body Combat	10	60	Intermediate
3	15	Boxing	Boxing	Boxing		90	Advanced
4	17	Yoga Class	Relax and unwind with yoga poses.	Yoga	10	60	Beginner
5	14	Zumba Mamba	Mamba going crazy	Zumba	10	45	Beginner
6	18	Boxing	Learn Boxing Techniques	Boxing	11	60	Hard

Now if we assign an instructor during creating class to the branches that the instructor is not in, it will trigger an error.

Let's book a class

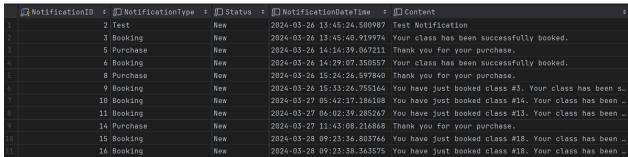
```
-- Booking class

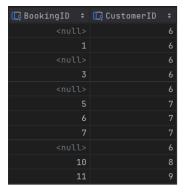
SELECT book_class( customer_id: 8, class_id: 18);

SELECT book_class( customer_id: 9, class_id: 18);
```

	№ BookingID ÷	☐ CustomerID ÷	∏ ClassID ÷	□ BookingDateTime	□ SessionType
1	1	6	3	2024-03-25 15:18:34.765730	Standard
2	2		3	2024-03-25 15:47:33.077211	Standard
3	3		3	2024-03-26 14:29:07.350557	Standard
4	5	7	3	2024-03-26 15:33:26.755164	Standard
5		7	14	2024-03-27 05:42:17.186108	Standard
6	7	7	13	2024-03-27 06:02:39.285267	Standard
7	10	8	18	2024-03-28 09:23:36.803766	Standard
8	11	9	18	2024-03-28 09:23:38.363575	Standard

After the customer booked the class it will notify the user in the notification table. As we have created a trigger for that.





It also triggers a change to LastActivityDay for customers also.

Now if Customer book class twice or have not login it will cause error as we have set the condition for it.

```
52 - Booking class

53 SELECT book_class ( customer_id: 8, class_id: 18);

54 SELECT book_class ( customer_id: 9, class_id: 18);

55

56

57

[P0001] ERROR: Customer has already booked this class.
Where: PL/pgSQL function book_class (integer, integer) line 40 at RAISE
```

And if we book class with inactive membership it will also cause an error

And if we book a class with an access level that is higher than our access level it will also cause errors.

```
-- Booking class in a Branch with different access level

SELECT book_class( customer_id: 8, class_id: 19);

[P0001] ERROR: Customer access level (Basic), does not match the class branch type (Premium).

Where: PL/pgSQL function book_class(integer,integer) line 45 at RAISE
```

If we change our mind we can also cancel our class.

```
-- Cancelling the class

SELECT cancel_class_booking( customer_id: 8, class_id: 18);
```

The booking disappeared...

<u>∏</u> BookingID	¢ į	Ç CustomerID ÷	☐ ClassID ÷	∏ BookingDateTime	∏ SessionType
1	1	6	3	2024-03-25 15:18:34.765730	Standard
2	2	6	3	2024-03-25 15:47:33.077211	Standard
3	3	6	3	2024-03-26 14:29:07.350557	Standard
4		7	3	2024-03-26 15:33:26.755164	Standard
5	6	7	14	2024-03-27 05:42:17.186108	Standard
6		7	13	2024-03-27 06:02:39.285267	Standard
7	11	9	18	2024-03-28 09:23:38.363575	Standard

Adding Product to SaleAndProduct Table

Purchasing a product

```
-- Purchasing a product

SELECT purchase_product( product_id: 3, customer_id: 8, purchase_quantity: 1,

purchase_price: 10, payment_method: 'Credit Card');
```

The condition applied almost the same to the booking class method. Which is:

- Customer must login
- Check if product exist and current stock of the product

And after purchasing the product the transaction is also loaded as we have also set triggers for this.



And it will trigger the notification function to notified us about our purchased.

№ NotificationID ÷		∏ Status ÷		☐ Content ÷
1 2	Test	New	2024-03-26 13:45:24.500987	Test Notification
2 3	Booking	New	2024-03-26 13:45:40.919974	Your class has been successfully booked.
3 5	Purchase	New	2024-03-26 14:14:39.067211	Thank you for your purchase.
4 6	Booking	New	2024-03-26 14:29:07.350557	Your class has been successfully booked.
5 8	Purchase	New	2024-03-26 15:24:26.597840	Thank you for your purchase.
6 9	Booking	New	2024-03-26 15:33:26.755164	You have just booked class #3. Your class has been s
7 10	Booking	New	2024-03-27 05:42:17.186108	You have just booked class #14. Your class has been
8 11	Booking	New	2024-03-27 06:02:39.285267	You have just booked class #13. Your class has been
9 14	Purchase	New	2024-03-27 11:43:08.216868	Thank you for your purchase.
10 16	Booking	New	2024-03-28 09:23:38.363575	You have just booked class #18. Your class has been
11 17	Purchase	New	2024-03-28 12:52:07.936768	Thank you for your purchase.
12 18	Purchase	New	2024-03-28 12:56:15.066821	Thank you for your purchase.

And if we purchase the product more than its stock quantity it will raise an error message.

```
82 -- Purchasing product more than its quantity
83 (9) SELECT purchase_product( product_id: 4, customer_id: 8, purchase_quantity: 3,
84 purchase_price: 10, payment_method: 'Credit Card');
85

[P0001] ERROR: Not enough stock to complete the purchase.
Where: PL/pgSQL function purchase_product(integer,integer,numeric,character varying) line 19 at RAISE
```

That's not it.

Our app can also send messages.

```
-- Sending direct message

SELECT send_direct_message( sender_id: 12, receiver_id: 14,

message_content: 'Hey Cbum I want to know more' ||

' about the boxing class you are teaching');
```

But we don't know how to check???

So we have created an inbox to see if someone has sent you a message or not?

```
-- Check user's inbox
SELECT user_inbox( user_id: 14);
```

It shows the **id** of the user that sends you the message and the ReadStatus that you have read or not. By the way, the user inbox just as normally showed only the latest messages.

```
User_inbox
1 (12,"Hey Cbum I want to know more about the boxing class you are teaching",Unread)
```

Let's take a look at the message.

```
-- See the direct message

SELECT get_direct_messages( user1_id: 12, user2_id: 14);
```

```
get_direct_messages

1 (11,12,14,"Hey Cbum I want to know more about the boxing class you are teaching","2024-03-28 13:12:53.171677")
```

Sending more messages to test

```
1 (12,"Hey thank you for reaching out to me I will let you know about information on-site. See you there.",Unread)
```

See the direct messages again

□user inhox

```
-- See the direct messages

SELECT get_direct_messages( user1_id: 14, user2_id: 12);
```

Now you can see the message in chat box pattern

```
☐ get_direct_messages

1 (11,12,14,"Hey Cbum I want to know more about the boxing class you are teaching","2024-03-28 13:12:53.171677")

2 (12,12,14,"I am a very big fan of your teaching","2024-03-28 13:32:06.971892")

3 (13,12,14,"Please contact me back asap","2024-03-28 13:32:13.351057")

4 (15,14,12,"Hey thank you for reaching out to me I will let you know about information on-site. See you there.","2024-03
```

And if you check user inbox again it is going to show read

```
--Check inbox again to see if the readStatus change or not?

SELECT user_inbox( user_id: 14);
```

And yes it does.

```
☐ user_inbox

1 (12,"Hey thank you for reaching out to me I will let you know about information on-site. See you there.",Read)
```

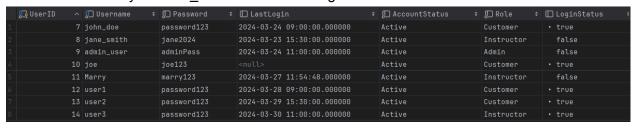
Since we have duplicate class we also created a procedure to fix it



If the due payment date is due, it will automatically renew the membership.

Lastly, don't forget to log out...

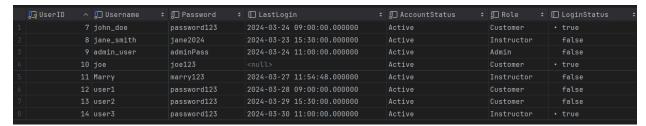
You can either use your user id or username to logout



```
-- Logging out

SELECT user_logout( user_id: 12);

SELECT user_logout( user_id: 'user2');
```



^{*}All the function used is attached in Mock UI Folder