User:

Endpoint: /user/register/
Methods: POST
Fields/payload: email, password, first_name, last_name, avatar, phone_number
Additional notes: Create a user account. Email, password, first_name, last_name, phone_number are required. Email must be unique.

Endpoint: /user/login/
Methods: POST
Fields/payload: email, password
Additional notes: If credentials are valid, log in user. Give auth token.

Endpoint: /user/profile/
Methods: GET
Additional notes: A user must be logged in. Show user's first_name, last_name, email, phone_number, avatar.

Endpoint: /user/logout/
Methods: POST
Additional notes: log the user out. If the user is unauthenticated, ignore the request.

Endpoint: /user/update/
Methods: POST
Fields/payload: first_name, last_name, avatar, phone_number, password
Additional notes: If any field is empty or missing, the corresponding value should not be updated. Email cannot be updated.

Endpoint: /user/classes/
Methods: GET
Fields: name, description, coach, capacity, time
Additional notes: Returns a list the user's class schedule and history in chronological order

Endpoint: /user/profile/payment_info/create/
Methods: POST
Fields/payload: cardnum, expiry, CVV
Additional notes: all fields are required. Add a new payment method to the user's account.

Endpoint: /user/profile/payment/history/
Methods: GET
Output: amount, cardinfo (last 4 digits of cardnum), datetime
Additional notes: return a JSON string of all payments made in the past, and future payments

Endpoint: /user/unsubscribe/
Methods: POST
Additional notes: Remove the user's subscription. If the user does not have a subscription, send an error message

Subscriptions

Endpoint: /subscriptions/<sub_id>/subscribe/
Methods: POST
Additional notes: add the subscription with id sub_id, to the currently logged in user's profile. A user must be logged in and authenticated. If given ID is not associated with any subscriptions, return 404 NOT FOUND. The user must not have a currently active subscription. The user must also have payment_info filled in already before a subscription is added. When user is subscribed, the first payment is made immediately, and each upcoming payment will be automatically made in the beginning of their period

Endpoint: /subscriptions/list/
Methods: Get
Additional notes: Returns a list of subscriptions that are available to subscribe to.

Studios

Endpoint: /studios/list/
Methods: GET
Parameters: latitude, longitude
Additional notes: Parameters are required. Returns a JSON string of a list of studios based on latitude and longitude(parameters). Studios are listed starting from the closest one. Each listed studio contains the name and location. Assuming geographical proximity to a specific location (e.g., my current location, a pinpoint on map, or a postal code) is converted into latitude and longitude in frontend.

Endpoint: /studios/<studio_id>/details/
Methods: GET
Additional notes: If given ID is not associated with any studios, return 404 NOT FOUND. Returns a list of details of the studio, which include name, phone_number, location, images, amenities, and a link to get the directions.

Endpoint: /studios/search/
Methods: GET
Parameters: studio_name, amenity, class_name, coach

Additional notes: All parameters are optional. Amenity, coach are searched by their names.Returns a JSON string of a list of studios that contains the associated value given in Parameters. These are filtered by AND conditions(i.e, name__icontains=gym and amenity__icontains=rope and ...).
Each listed studio contains the name and location. Note that /studios/search/ without any parameters returns all studios.

Endpoint: /studios/<studio_id>/classes/schedule/
Methods: GET
Additional notes: If the given ID is not associated with any studios, return 404 NOT FOUND.
Classes appear in the order of their start time (from now), and the class information(name, description, capacity, coach_name, and time) is shown. Past or canceled classes are not listed.

Endpoint: /studios/<studio_id>/classes/<class_id>/enrol_all/
Methods: POST
Additional notes: Enroll the user in all instances of the class that user hasn't already enrolled in. A user must be logged in. The user must have an active subscription, if not return an error. If given ID is not associated with any studios/class, return 404 NOT FOUND.

Endpoint: /studios/classtime/<classtime_id>/enrol/
Methods: POST
Additional notes: Enroll the user in the single instance of the class. A user must be logged in. The user must have an active subscription, if not return an error. If the user is already enrolled in this class, return an error. If given ID is not associated with any studios/class, return 404 NOT FOUND.

Endpoint:/studios/<studio_id>/classes/<class_id>/drop_all/
Methods: POST
Additional notes: Drop all class times of the class the user is currently enrolled in. If none is enrolled in, do nothing and return none dropped. A user must be logged in, otherwise return 401 UNAUTHORIZED. If the given ID is not associated with any class, return 404 NOT FOUND. Remove this class from the user's class list.

Endpoint: /studios/classtime/<classtime_id>/drop/
Methods: POST
Additional notes: Drop the user from this class. If not enrolled in it, do nothing and return none dropped. A user must be logged in. If the user is already enrolled in this class, return an error.

Endpoint: /studios/classes/search/
Methods: GET
Parameters: class_name, coach_name, date, time_start, time_end
Additional notes: All parameters are optional. A user must be logged in, otherwise return 401 UNAUTHORIZED. Search for classes that satisfy non-empty fields: if class_name parameter is not empty,

search for classes whose names contain the input parameter; if coach_name parameter, search for classes whose coaches' names contains the input parameter; if date is not empty, search for class that started before this date and ends after this date; if time_start is not empty, search for classes that start at or after time_start; if time_end is not empty, search for classes that end at or before time_end.

Admin panel

@admin_only
Endpoint: /admin/studios/coach/add/
Methods: GET, POST
Fields/payload: name
Additional notes: name is required

@admin_only
Endpoint: /admin/studios/coach/<coach_id>/change/
Methods: GET, POST
Fields/payload: name
Additional notes: name is required

@admin_only
Endpoint: /admin/studios/coach/<coach_id>/delete/
Methods: POST

@admin_only
Endpoint: /subscriptions/subscription/add/
Methods: POST
Fields/payload: price, length
Additional notes: all fieldsy are required

@admin_only
Endpoint: /subscriptions/subscription/<sub_id>/change/
Methods: GET, POST
Fields/payload: price, length
Additional notes: all fields are required

@admin_only

Endpoint: /subscriptions/subscription/<sub_id>/delete/
Methods: POST

@admin_only
Endpoint: /admin/studios/location/add/
Methods: GET, POST
Fields/payload: latitude, longitude, address, postal_code
Additional notes: all fields are required

@admin_only
Endpoint:/admin/studios/location/<location_id>/change/
Methods: GET, POST
Fields/payload: latitude, longitude, address, postal_code
Additional notes: all fields are required

@admin_only
Endpoint:/admin/studios/location/<location_id>/delete/
Methods: POST
Additional notes:

@admin_only
Endpoint: /admin/studios/studio/add/
Methods: GET, POST
Fields/payload: name, phone_number, location, images(image file), amenities(type and quantity)
Validation: All fields except images and amenities are required.
Additional notes: Location is One To One field. Name, phone_number, location are required.

@admin_only
Endpoint:/admin/studios/studio/<studio_id>/change/
Methods: GET, POST
Fields/payload: name, phone_number, location, images(image file), amenities(type and quantity)
Additional notes: Location is One To One field. Name, phone_number, location are required.

@admin_only
Endpoint: /admin/studios/studio/<studio_id>/delete/
Methods: POST

@admin_only

Endpoint: /admin/studios/class/add/

Methods: GET, POST

Fields/payload: name, description, capacity, range_date_start, range_date_end, day, start_time, end_time, studio, coach, keywords, class_times(status, time)

Validation: All fields are required except keywords and class_times. Capacity must be a non-negative integer. range_date_end cannot be earlier than range_date_start and end time cannot be earlier than end time.

Additional notes: All fields except for keywords and class_time are required. range_date_start means the earliest date that the class can start, so it's not necessarily the start date of a class. Same goes for range_date_end. It will create instances of class_time with a given day, start_time, and end_time that falls within range_date_start and range_date_end. We are assuming class occurs weekly.

@admin_only

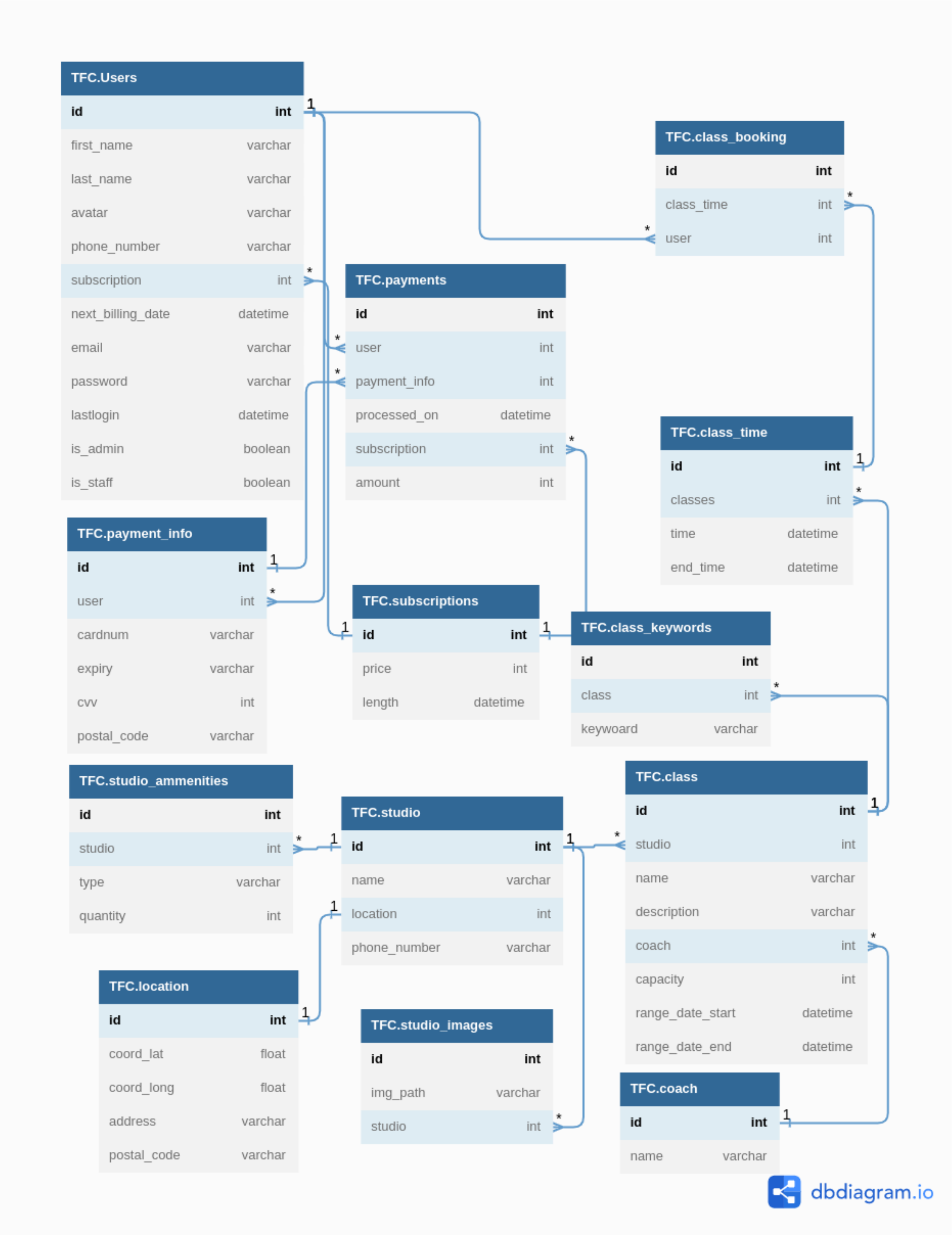
Endpoint: /admin/studios/class/<class_id>/change/

Methods: GET, POST

Fields/payload: name, description, capacity, range_date_start, range_date_end, day, start_time, end_time, studio, coach, keywords, class_times(status, time)

Validation: Do the same validations as /admin/studios/class/add/

Additional notes: All fields except for keywords and class_time are required. New instances of class_time would be created if new range_date_start and range_date_end contain the dates that previous range didn't include. On the other hand, if the new range_date_start and range_date_end doesn't contain the dates that the previous range did include, then the corresponding class is canceled.



TFC.Users	
id	int
first_name	varchar
last_name	varchar
avatar	varchar
phone_number	varchar
subscription	int
next_billing_date	datetime
email	varchar
password	varchar
lastlogin	datetime
is_admin	boolean
is_staff	boolean

TFC.class_booking	
id	int
class_time	int
user	int

TFC.payments	
id	int
user	int
payment_info	int
processed_on	datetime
subscription	int
amount	int

TFC.class_time	
id	int
classes	int
time	datetime
end_time	datetime

TFC.payment_info	
id	int
user	int
cardnum	varchar
expiry	varchar
cvv	int
postal_code	varchar

TFC.subscriptions	
id	int
price	int
length	datetime

TFC.class_keywords	
id	int
class	int
keyword	varchar

TFC.studio_ammenities	
id	int
studio	int
type	varchar
quantity	int

TFC.studio	
id	int
name	varchar
location	int
phone_number	varchar

TFC.class	
id	int
studio	int
name	varchar
description	varchar
coach	int
capacity	int
range_date_start	datetime
range_date_end	datetime

TFC.location	
id	int
coord_lat	float
coord_long	float
address	varchar
postal_code	varchar

TFC.studio_images	
id	int
img_path	varchar
studio	int

TFC.coach	
id	int
name	varchar