Setup Instruction

Within the main folder, there are 2 sh files, run.sh and setup.sh. Run the setup.sh file to setup the server database and create a default superuser.

The default superuser has the username PBAdmin and password PBAdmin. This account can be used to access the Admin page at [web_url]/admin/. It is recommended that the password of this account be changed immediately either through the website admin page or through the Edit Profile API call below.

Once the setup.sh file has been run, the run.sh file can then be run to start the server. A token for the Admin User can be generated through the Login API call or in the Admin Panel.

API Documentation

Account Authorization

General Notes:

- With Exception to Set User as Coach, the authentication token is optional, however if it is provided, an invalid token will be rejected. If a user attempts a second login, their existing token will be revoked.

Register:

- Endpoint: /accounts/register/
- Methods:
  - POST
- Headers:
  - Authorization: Token {{AuthToken}}
- Field/Payload
  - username
  - password1
  - password2
  - first_name
  - last_name
  - email
  - phone_num
- Response
  - If the account was successfully created, a status 200 will be returned, otherwise an status 400 will be returned.
- Notes
  - password1 and password2 must be at least 8 characters.

    - email must be a valid email format.
    - phone_num must be a valid phone number

Login:

- Endpoint: /accounts/login/
- Methods
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - username
    - password
- Response
    - Status 400 - request malformed
    - Status 401 - Login Failed (Username or Password invalid)
    - Status 200 - Success, a User Token will be returned.
- Notes
    - If the authentication is successful, a user token will be returned.

Logout:

- Endpoint: /accounts/logout/
- Methods
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Token was not provided
    - Status 200 - Account logged out successfully (token deleted).
- Notes
    - This request has a get request. The corresponding AuthToken will be deleted

Set User As Coach:

- Endpoint: /accounts/setcoach/<int:account_id>/
- Methods
    - GET, DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response]
    - Status 401 - authorization was not provided or invalid.
    - Status 404 - account_id was not found

    - Status 200 - User set or removed as a coach.
- Notes
    - The client must be a site Admin, otherwise the request will be denied
    - The GET request sets the account as a coach, the DELETE request removes the coach status.

Account General

View Profile:

- Endpoint: /accounts/view/
- Methods
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Authorization not provided or invalid
    - Status 200 - Success
- Notes
    - Gets information about the currently authenticated user.

Edit Profile:

- Endpoint: /accounts/edit/
- Methods
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - password1
    - password2
    - first_name
    - last_name
    - email
    - phone_num
- Response
    - Status 400 - Request Malformed
    - Status 401 - Authorizations not provided or invalid
    - Status 200 - Success
- Notes

    - Behavior: If the field or payload exist, it is treated as if the data contained is intended change, ie: if a client provides an empty string, the corresponding field will be changed to the empty string.
    - For passwords, if the password1 field is provided, the password validator is performed. However if password1 is not provided, password2 will be ignored.

Set Profile Picture:

- Endpoint: /accounts/icon/set/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload:
- Response:
    - Status 400 - Malformed request, usually image file is invalid
    - Status 200 - Success
- Notes:
    - On success, the name of the image will be returned, the image can be retrieved at /accounts/icon/<image_name>

Get Profile Picture:

- Endpoint: /accounts/icon/<image_name>
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload:
- Response:
    - Status 404 - Image was not found
    - Status 200 - Image found
- Notes
    - On success, the image is returned.
    - The name of the image can be retrieved when the image was first set as the profile picture in Set Account Picture.
    - Alternatively the path to the image can be obtained from View Profile.

Remove Profile Picture:

- Endpoint: /accounts/icon/clear
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}

- Fields/Payload:
- Response:
    - Status 401 - Authorization invalid or not provided
    - Status 200 - Request successful
- Notes:

View Enrolled Classes:

- Endpoint: /accounts/enrolledclasses/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - sort
    - filter
- Response:
    - Status 401 - Authorization not provided or invalid
    - Status 200 - Request successful
- Notes
    - Returns a paginated list of all class sessions the user has been enrolled in.
    - sorting has 2 options
        - asc: sort by date, ascending order
        - des: sort by date, descending order (default)
    - filter has three options:
        - all: default option, view all enrolled classes
        - past: view only past and inactive subscriptions
        - future: view current and future subscriptions
    - If any of the payload data is malformed, it is ignored and the default value used.

Add Subscription:

- Endpoint: /accounts/subscriptions/add/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - subscription_id
    - any additional payment information requirements, such as the code on the back of the card.
    - do_not_renew
- Response
    - Status 401 - Authorization invalid or not provided, or the requested subscription is not available for users, or if the user's payment method is invalid.

    - Status 404 - Requested subscription was not found
    - Status 200 - Request Successful
- Notes
    - JSON data containing whether, whether payment was successful (if the subscription is immediate) or not.
    - If the payment was not successful, an extra return field contains the reason.
    - If a user does not have a payment option in their profile, payment will fail unless they already have an active subscription.
    - if do_not_renew exists in the payload, then the automatic renewal of subscription is not done.

Get User Subscriptions:

- Endpoint: /accounts/subscriptions/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - sort
    - filter
- Response
    - Status 401 - Authentication is invalid or not provided
    - Status 200 - Request Successful
- Notes
    - The response is a paginated list of subscriptions by the user.
    - sorting has 2 options
        - asc: sort by date, ascending order
        - des: sort by date, descending order (default)
    - filter has three options:
        - all: default option, view all subscriptions
        - past: view only past and inactive subscriptions
        - future: view current and future subscriptions

Get User Subscription:

- Endpoint: /accounts/subscriptions/[subscription_id:int]/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Authentication is invalid or not provided

    - Status 404 - User subscription not found
    - Status 200 - Request Successful
- Notes
    - Response is the details on a specific user subscription, if it exists.

Cancel Subscription:

- Endpoint: /accounts/subscriptions/[subscription_id:int]/
- Methods:
    - DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Authentication invalid or not provided, or subscription has already begun
    - Status 404 - User subscription not found
    - Status 200 - Request Successful
- Notes
    - Cancels the specific subscription
    - Subscription must not have started yet..
    - Subsequent subscriptions will be shifted down

Cancel All Subscriptions:

- Endpoint: /accounts/subscriptions/cancel/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Authentication invalid or not provided
    - Status 200 - Request Successful
- Notes
    - Cancels all future subscriptions

Add Payment Method:

- Endpoint: /accounts/payment/add/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}

- Fields/Payload
  - card_type
  - card_num
  - card_name
  - exp_month
  - exp_year
- Response
  - Status 400 - Request body malformed
  - Status 401 - Authentication invalid or not provided
  - Status 200 - Request Successful
- Notes
  - Adds a new payment method

Remove Payment Method:

- Endpoint: /accounts/payment/remove/
- Methods:
  - DELETE
- Headers:
  - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
  - Status 401 - Authentication invalid or not provided
  - Status 200 - Request successful
- Notes
  - Removes the current active payment method.

View Payment Method:

- Endpoint: /accounts/payment/
- Methods:
  - GET
- Headers:
  - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
  - Status 401 - Authentication invalid or not provided
  - Status 404 - User does not have an active payment method
  - Status 200 - Request Successful.
- Notes
  - View the current active payment method if it is available.

Studio

Create Studio:

- Endpoint: /studios/create/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - name
    - address
    - post_code
    - phone_num
    - images(can be empty)
- Response
    - Status 200:
        - Studio creation success
    - Status 400:
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing
    - Status 401:
        - client without web admin privileges attempts to access this endpoint
- Notes:
    - Auth token correspond to an admin

Delete Studio:

- Endpoint: /studios/[studio_id:int]/delete/
- Methods:
    - DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 200:
        - Studio deletion successful
    - Status 401:
        - Client without web admin privileges attempts to access this endpoint

    - Status 404:
        - non-existing studio is inputted
- Notes:
    - Deletes Amenities relating to studio
    - Auth token correspond to an admin

Edit Studio:

- Endpoint: /studios/[studio_id:int]/edit/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - name
    - address
    - post_code
    - phone_num
    - images
- Response
    - Status 200:
        - Studio edit successful
    - Status 401:
        - Client without web admin privileges attempts to access this endpoint
    - Status 400
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing:
    - Status 404:
        - non-existing studio is inputted
- Notes
    - Fields with no value are ignored.
    - Auth token correspond to an admin

Get Studio:

- Endpoint: /studios/[studio_id:Int]
- Methods:
    - GET
- Headers:
- Fields/Payload
- Response

    - Status 200:
        - Studio view successful
    - Status 401:
        - Client without web admin privileges attempts to access this endpoint
    - Status 404:
        - non-existing studio is inputted
    -
- Notes
    - If a studio of a non-existing studio is inputted, a message is sent

Get Studios:

- Endpoint: /studios/
- Methods:
    - GET
- Headers:
- Parameters
    - n
    - a
    - cln
    - chn
- Fields/Payload
    - LocationData
- Response
    - Status 401 - Authentication invalid
    - Status 200 - Request successful
- Notes
    - The response is a paginated list of studios sorted by proximity
    - Authorization is not required, however if provided it must be valid.
    - If no Location Data is provided, sort by name
    - LocationData in the form of a coordinate:
        - ie: 43.66535212044863,-79.38639192519997
    - n is studio name
    - a is amenity type
    - cln is class name
    - chn is coach name

Add Amenities:

- Endpoint: /studios/[studio_id:int]/amenities/add/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload

    - type
    - quantity
- Response
    - Status 200:
        - Amenity add successful
    - Status 401:
        - Client without web admin privileges attempts to access this endpoint
    - Status 400
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing:
    - Status 404:
        - non-existing amenity is inputted
- Notes
    - Auth token correspond to an admin

Edit Amenity:

- Endpoint: /studios/[amenity_id:int]/amenities/edit
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - type
    - quantity
- Response
    - Status 200:
        - Amenity edit successful
    - Status 401:
        - Client without web admin privileges attempts to access this endpoint
    - Status 400
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing:
    - Status 404:
        - non-existing amenity is inputted
- Notes
    - Auth token correspond to an admin

Delete Amenity:

- Endpoint: /studios/[int:amenity_id]amenities/delete/
- Methods:
  - DELETE
- Headers:
  - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
  - Status 200:
    - Amenity delete successful
  - Status 401:
    - Client without web admin privileges attempts to access this endpoint
  - Status 404:
    - non-existing amenity is inputted
- Notes
  - Auth token correspond to an admin

Get Amenities:

- Endpoint: /studios/[studio_id:int]/amenities/
- Methods:
  - GET
- Headers:
- Fields/Payload
- Response
  - Status 200:
    - A list of amenities returned
  - Status 401:
    - Client without authorization attempts to access this endpoint
- Notes
  - If studio id of a non-existing studio is inputted then empty list is provided

Get Studio Image:

- Endpoint: /studios/studioimages/[image_name:str]/
- Methods:
  - GET
- Headers:
- Fields/Payload
- Response
  - Status 200:
    - Image returned
  - Status 404:
    - Image does not exist

- Notes
  - If studio id of a non-existing studio is inputted then empty list is provided

GymClass

Create Classes:

- Endpoint: /classes/<int:studio_id>/create/
- Methods:
  - POST
- Headers:
  - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
  - name
  - coach
  - description
  - keywords
  - earliest_date
  - last_date
  - day
  - start_time
  - end_time
  - enrollment_capacity
- Response
  - Status 200:
    - GymClass Add successful
  - Status 401:
    - Client without web admin privileges attempts to access this endpoint
  - Status 400
    - A Payload keys are missing
    - A Payload Key is Invalid
    - Chronological error in start time, end time,earliest_date, and last_date
  - Status 404:
    - non-existing studio/coach is inputted

- Notes
  - Auth token correspond to an admin

Delete Classes:

- Endpoint: /classes/<int:gym_class>/delete/
- Methods:
    - DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
    - delete(optional): If this payload is set to true then the
- Response:
    - Status 200:
        - GymClasss Delete successful
    - Status 401:
        - Client without web admin privileges attempts to access this end
    - Status 404:
        - gym_class was not found
- Notes:
    - If delete param is set to True then the gym class is deleted else canceled
    - Auth token correspond to an admin

Delete ClassesSchedule:

- Endpoint: /classes/schedule/<int:gym_schedule>/delete/
- Methods:
    - DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
    - delete(optional): If this payload is set to true then the
- Response
    - Status 200:
        - GymClasss Delete successful
    - Status 401:
        - Client without web admin privileges attempts to access this end
    - Status 404:
        - gym_class was not found
- Notes
    - If delete param is set to True then the gym class is deleted else canceled
    - Auth token correspond to an admin

ClassSchedule of Studio:

- Endpoint: /classes/studio/<int:studio_id>/list/
- Methods:
  - GET
- Headers:
  - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
  - Status 200:
    - List Found successful
  - Status 404:
    - studio/gym class schedule was not found
- Response
- Notes

Search/Filter ClassesSchedule:

- Endpoint: /classes/
- Methods:
  - POST
- Headers:
  - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
  - name
  - coach_name
  - date
  - time_range
- Response
  - Status 200:
    - List Found successful
  - Status 400:
    - A Payload Key is Invalid
- Notes

Edit GymClass

- Endpoint: /classes/<int:gymclass_id>/edit/
- Methods:
  - POST
- Headers:
  - Authorization: Token {{AuthToken}}
- Parameters

- Fields/Payload
    - studio
    - name
    - description
    - keywords
    - earliest_date
    - last_date
    - day
    - start_time
    - end_time
- Response
    - Status 200:
        - Edit Successful
    - Status 400:
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing:
    - Status 404:
        - studio/Gym Class not found
    - Status 401:
        - Client without web admin privileges attempts to access this end
- Notes
    - Auth token correspond to an admin

Edit GymClass Schedule

- Endpoint: /classes/schedule/<int:gymclass_schedule_id>/edit/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Parameters
- Fields/Payload
    - date
    - coach
    - enrollment_capacity
    - enrollment_count
    - start_time
    - end_time
    - is_cancelled
- Response
    - Status 200:

        - Edit Successful
    - Status 400:
        - If any of the fields is missing a message is sent defining which keys is missing
        - If any of the inputted fields is invalid a message is sent as to which key is missing:
    - Status 404:
        - studio/Gym Class schedule not found
    - Status 401:
        - Client without web admin privileges attempts to access this end
- Notes
    - Auth token correspond to an admin

Enroll In Class Session:

- Endpoint: /classes/session/<int:session_id>/signup/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}} (Optional)
- Parameters
- Fields/Payload
- Response
    - Status 401 - Authentication invalid or not provided, or the target class is not available for enrollment
    - Status 404 - Class does not exist
    - Status 200 - Request Successful
- Notes

Drop Class Session:

- Endpoint: /classes/session/<int:class_id>/drop/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}} (Optional)
- Parameters
- Fields/Payload
- Response
    - Status 401 - Authentication invalid or not provided, or the target class is not available for enrollment
    - Status 404 - Class does not exist
    - Status 200 - Request Successful
- Notes

Enroll In Class:

- Endpoint: /classes/<int:class_id>/signup/
- Methods:
  - GET
- Headers:
  - Authorization: Token {{AuthToken}} (Optional)
- Parameters
- Fields/Payload
- Response
  - Status 401 - Authentication invalid or not provided, or the target class is not available for enrollment
  - Status 404 - Class does not exist
  - Status 200 - Request Successful
- Notes
  - Enrolls in all available sessions for a class

Drop Class:

- Endpoint: /classes/<int:class_id>/drop/
- Methods:
  - GET
- Headers:
  - Authorization: Token {{AuthToken}} (Optional)
- Parameters
- Fields/Payload
- Response
  - Status 401 - Authentication invalid or not provided, or the target class is not available for enrollment
  - Status 404 - Class does not exist
  - Status 200 - Request Successful
- Notes
  - Drops all sessions associated with a class.

Drop All Classes:

- Endpoint: /classes/dropclasses/
- Methods:
  - GET
- Headers:
  - Authorization: Token {{AuthToken}} (Optional)
- Parameters
- Fields/Payload
- Response

    - Status 401 - Authentication invalid or not provided, or the target class is not available for enrollment
    - Status 404 - Class does not exist
    - Status 200 - Request Successful
- Notes
    - Drops all future class sessions.

Website Subscriptions

View Subscriptions:

- Endpoint: /subscriptions/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 200 - Request successful
- Notes
    - A paginated list of available subscriptions
    - Authorization header is optional but must be valid if provided

Add Subscription:

- Endpoint: /subscriptions/add/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - name
    - description
    - price
    - days
    - hours
- Response
    - Status 400 - Request body malformed
    - Status 401 - Authentication invalid or not provided
    - Status 200 - Request successful
- Notes
    - The auth token must correspond with an Admin user.

Edit Subscription:

- Endpoint: /subscriptions/[subscription_id:int]/edit/
- Methods:
    - POST
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - name
    - description
    - price
    - days
    - hours
    - available
- Response
    - Status 400 - Request body malformed
    - Status 401 - Authentication invalid or not provided
    - Status 404 - Requested subscription does not exist
    - Status 200 - Request successful
- Notes
    - missing fields are ignored. If you wish to not modify a field, do not include the field. If a field is an empty string, it is treated as such or as 0
        - However if only day is included, then the hour field is treated as 0 and vice versa.

View Subscription:

- Endpoint: /subscriptions/[subscription_id:int]/
- Methods:
    - GET
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
- Response
    - Status 401 - Authentication invalid
    - Status 404 - Requested subscription does not exist
    - Status 200 - Request successful
- Notes
    - The details of the subscription
    - Authorization is not required, however if provided it must be valid.

Remove Subscription:

- Endpoint: /subscriptions/[subscription_id:int]/
- Methods:

    - DELETE
- Headers:
    - Authorization: Token {{AuthToken}}
- Fields/Payload
    - remove
- Response
    - Status 401 - Authentication invalid or not provided
    - Status 404 - Requested subscription does not exist
    - Status 200 - Request successful
- Notes
    - The authorization must be from an admin.
    - The remove payload is optional, by default removing a subscription simply makes it unavailable (invisible) to normal users, users who already have the subscription will keep the subscription. If the remove payload is included and is true, then the subscription will be deleted entirely.