Account app:

Account:

- Everything from django default user.
- phone : a field for phone number
- avatar : a field for image

APIs for Account:

1. **/accounts/register/** - signup account

   method: POST
   header: none
   payload: username, first_name, last_name, phone, email, avatar, password
   , password2

2. **/accounts/login/** - obtain login token

   method: POST
   header: none
   payload: username, password

3. **/accounts/profile/** - view profile, and edit profile

   method: GET, PUT
   header: token auth
   payload: first_name, last_name, phone, email, avatar, password, password2

Subscription app:

Plan:

- amount: amount to charge each time
- duration: two choices, Month or Year

Subscription:

- account: point to account model
- plan: point to plan model
- card: card number
- date: start date of subscription

Payment:

- account: point to account
- datetime: time of payment
- amount: amount paid
- card: card number

APIs for Subscription:

1. /subscription/plans/ - list all available plans

   method: GET
   header: none
   payload: none

2. /subscription/manage - view, add, edit, and cancel my subscription, in case of cancellation, you will be removed from all classes you enrolled that start after your subscription expires.

   method: GET, POST, PUT, DELETE
   header: token auth
   payload: card, plan

3. /subscription/payment/history - list all payment history

   method: GET
   header: token auth
   payload: none

4. /subscription/payment/future - show my payment date and schedule

   method: GET
   header: token auth
   payload: none

5. /subscription/payment/update - admin only endpoint, check all subscriptions and to see if the users need to pay today or not, and make corresponding changes. Notice that this request should only be sent once a day otherwise the user might be charged twice

   method: GET
   header: token auth
   payload: none

Classes App:

Class (General information of a class)
- name: name of this class
- studio: studio where this class is held (Foreign key to **Studio**)
- description: description for this class
- coach: name of the coach running this class
- keywords: list of keywords
- capacity: capacity of this class
- recurrences: recurrence rule with an end date (e.g. Wednesday every week until 2022-12-01)
- start_time: start time of this class
- end_time: end time of this class

Event (Event of a class)
- classInfo: general information of this class event (Foreign key to **Class**)
- date: date of this class event
- start_time: start time of this event
- end_time: end time of this event
- cancelled: whether this event is cancelled or not
- enrolled_user: list of users enrolled in this class event

Creating a class automatically creates events with each date of future occurrences.

APIs for Classes:

1. **/studios/<studio_id>/classes** - list all upcoming class events in a studio. Searching and filtering on name, coach, date, and time is available if the parameter is provided.

   method: GET
   header: none
   payload: none
   parameter: (optional)
   - search: keyword to search for a class that has a related class name, coach, keyword, date, and time.
   - name: Filter class events with this name.
   - coach: Filter class events with this coach.
   - date: Filter class events that are on this date.
   - start_date: Filter class events that are after the start_date.
   - end_date: Filter class events that are before the end_date.
   - start_time: Filter class events that have start time after the start_time.
   - end_time: Filter class events that have start time before the end_time.
   - keyword: Filter class events with the given keyword.

2. **/classes/enroll** - Enroll all future events of a class that are uncancelled and have available space. Account must have a valid subscription to enroll.

   method: POST
   header: token auth
   payload:
   - class_id (required): Id of a class want to enroll

3. **/classes/event/enroll** - Enroll in an upcoming class event if is uncancelled and has available space. Account must have a valid subscription to enroll.

   method: POST
   header: token auth
   payload:
   - event_id (required): Id of an event want to enroll

4. **/classes/drop** - Drop all future events of a class the user is enrolled in.

   method: POST
   header: token auth
   payload:
   - class_id (required): Id of a class want to drop

5. **/classes/event/drop** - Drop an upcoming class event if the user has enrolled.

   method: POST
   header: token auth
   payload:
   - event_id (required): Id of an event want to drop

6. **/classes/schedule** - List upcoming uncancelled class events the user is enrolled in.

   method: GET
   header: token auth
   payload: none

7. **/classes/history -** List past class events the user has enrolled.

   method: GET
   header: token auth
   payload: none

Studios App:

Studio (general information for a studio)
- name: name of this studio
- address: address of the studio
- longitude: longitude of the studio
- latitude: latitude of the studio
- postal_code: the postcode of the studio
- phone_number: the phone number of the studio
- distance: the distance between user and this studio, it is empty when it is initialized

Amenity (amenity of a class)
- studio: studio where this amenity is held (Foreign key to **Studio**)
- type: the type of the amenity
- quantity: the quantity of the amenity

Images (image of a class)
- studio: studio where this image is held (Foreign key to **Studio**)
- image: imageField for the image

APIs for Studios:

1. **/studios/all** - List all studios based on the distance between user
   method: GET
   header: none
   payload: none
   parameter:
   - longitude (required): input of the user's longitude
   - latitude (required): input of the user's latitude
   - search: keyword to search for a studio that has a related studio name, amenity type, class name, and coach
   - name: Filter studio with this name.
   - class_name: Filter studios contains this class
   - class_coach: Filter studios with this coach.
   - amenity_type: Filter studio contains this type of amenity

2. **/studios/<studio_id>/details** - View the detail of the studio
   method: GET
   header: none
   payload: none