Accounts

Models

UserProfile

- first_name: CharField for user's first name, maximum of 150 characters
- last_name: CharField for user's last name, maximum of 150 characters
- password: CharField for user's password
- email: EmailField for user's email
- phone_num: CharField for user's phone number, maximum of 20 characters
- avatar: ImageField for user's avatar, optional
- subscribed: BooleanField indicating if user is subscribed to a plan, default is False

Signing Up a User

Anyone can register to create a user account. Password must be at least 8 characters and confirmation password must match. Phone number must be a standard 10-digit phone number.

Endpoint: /accounts/signup/ **Method:** POST

Payload:

```
{
    "first_name": "string",
    "last_name": "string",
    "password": "string",
    "password2": "string",
    "email": "string@string.string",
    "phone_num": "string",
    "avatar": null
}
```

Response:

```
{
    "pk": int,
    "first_name": "string",
    "last_name": "string",
    "password": "password hash",
    "email": "string@string.string",
    "phone_num": "string",
    "avatar": null
}
```

Logging In a User

A registered user can log in using their email and password. If credentials are valid, an authorization Bearer token will be returned. This token should be passed for all views involving authenticated users/

Endpoint: /accounts/token/ **Method:** POST

Payload:

```
{
    "email": "string",
    "password": "string"
}
```

Response:

```
{
    "refresh": "string",
    "access": "string"
}
```

Editing a User

An authorized user can edit their account details (first name, last name, email, phone number, and avatar). If successful, the full user account details will be returned.

Endpoint: /accounts/<user_id>/ **Method:** PATCH

Authorization: Bearer <token>

Payload:

```
{
    "first_name": "string",
    "last_name": "string",
    "email": "string@string.string",
    "phone_num": "string",
    "avatar": null
}
```

Response:

```
{
    "pk": int,
    "first_name": "string",
    "last_name": "string",
    "password": "password has",
    "email": "string@string.string",
    "phone_num": "string",
    "avatar": null
}
```

Studios

Models

Studio

- name: CharField for studio's name, maximum of 150 characters
- address: CharField for studio's address, maximum of 150 characters
- location_lat: DecimalField for studio's latitude, maximum of 9 digits and 6 decimal places, default is 0
- location_long: DecimalField for studio's longitude, maximum of 9 digits and 6 decimal places, default is 0
- postal_code: CharField for studio's postal code, maximum of 10 characters
- phone_num: CharField for studio's phone number, maximum of 20 characters

Amenity

- studio: ForeignKey referencing the studio in which the amenity belongs to
- type: CharField for amenity's type, maximum of 50 characters
- quantity: PositiveIntegerField for amenity's quantity

Image

- studio: ForeignKey referencing the studio in which the image belongs to
- image: ImageField for studio image, optional

Listing Studios by Distance

Any user can view studios sorted by distance by entering their desired latitudinal and longitudinal coordinates.

Endpoint: /studios/distancelist/ **Method:** PUT

Authorization: Bearer <token>

Payload:

```
{
    "lat": "string",
    "long": "string"
}
```

Response:

```
[]
```

Viewing Studio Details

Any user can view a studio's details by providing the studio id in the endpoint.

Endpoint: /studios/<studio_id>/ **Method:** GET

Authorization: Bearer <token>

Response:

```
{
    "name": "string",
    "address": "string",
    "location_lat": float,
    "location_long": float,
    "postal_code": "string",
    "phone_num": "string",
    "amenities": [amenities],
    "images": [images],
    "directions": "string"
}
```

Searching & Filtering Studios

A user can search and filter through studios that have their desired characteristics.

Endpoint: /studios/filter-studios/ **Method:** GET

Authorization: Bearer <token>

Params:

```
{
    "studio_name": "string",
    “amenities”: "string",
    “class_name” : ”string”,
    “coach” : “string”
}
```

Response:

```
{[
  {
      "id": 1,
      "name": "dfsf",
      "address": "202-180 Mississauga Valley Blv",
      "location_lat": "0.000001",
      "location_long": "0.000001",
      "postal_code": "L5A3M2",
      "phone_num": "647-620-1076"
  }
]
}
```

Classes

Models

ClassSet

- studio: ForeignKey referencing the studio in which the class set belongs to
- name: CharField for class set's name, maximum of 200 characters
- description: CharField for class set's description, maximum of 200 characters
- coach: CharField for class set's coach, maximum of 200 characters
- start: DateTimeField for class set's start date and time
- end_time: TimeField for class set's end time
- reoccur_until: DateField for class set's last recurring date
- capacity: PositiveIntegerField for class set's capacity

ClassSession

- name: ForeignKey referencing the class set in which the class session belongs to
- description: CharField for class session's description, maximum of 200 characters
- coach: CharField for class session's coach, maximum of 200 characters
- start_date_time: DateTimeField for class session's start date and time
- end_time: TimeField for class session's end time
- enrolled: PositiveIntegerField for class session's enrolled users, default is 0

Keyword

- classSession: ForeignKey referencing the class session in which the keyword belongs to
- word: CharField for keyword, maximum of 200 characters

EnrolledUser

- user: ForeignKey referencing the user in which to enrol
- class_session: ForeignKey referencing the class session in which enrol the user to

Enrol into Class Session

Users are able to enrol in a class session based on the class set and class session id. The response will return whether the insert was successful and any errors.

Endpoint: /classes/<int:id>/enrol-class-session/	**Method:** POST
Authorization: Bearer <token>	

Payload:
{
"class_session": "string",
}

Response:
{
"You have successfully enrolled in this class session"
}

Enrol into Class Set

Users are able to enrol in a class set based on the class set id. The response will return whether the insert was successful and any errors.

Endpoint: /classes/<int:id>/enrol-class-set/ **Method:** GET

Authorization: Bearer <token>

Response:
{
"You have successfully enrolled in this class set"
}

Drop into Class Session

Users are able to drop in a class session based on the class set and class session id. The response will return whether the delete was successful and any errors.

Endpoint: /classes/<int:id>/drop-class-session/ **Method:** DELETE

Authorization: Bearer <token>

Payload:
{
"class_session": "string",
}

Response:
{
"You have successfully dropped in this class session"
}

Drop into Class Set

Users are able to drop in a class set based on the class set id. The response will return whether the delete was successful and any errors.

Endpoint: /classes/<int:id>/drop-class-set/ **Method:** DELETE

Authorization: Bearer <token>

Response:

```
{
   "You have successfully dropped in this class set"
}
```

Filter Class Sessions

Users are able to filter through class sessions by passing in either class name, coach, start time and date or end time. The response will then result in class sessions based on the filters.

Endpoint: /classes/1/filter-sessions/ **Method:** GET

Authorization: Bearer <token>

Params:

```
{
   "class_name": "string",
   “coach”: "string",
   “start_date_time” : ”2022-11-24T23:45:11Z”,
   “end_time” : “23:45:14”
}
```

Response:

```
{
  [
   {
      "name": "309",
      "description": "sda",
      "coach": "sad",
      "start_date_time": "2022-11-24T23:45:11Z",
      "end_time": "23:45:14",
      "enrolled": 0
   }
]
}
```

User Schedule

User are able to view all their class history in chronological order.

Endpoint: /classes/user-schedule/ **Method:** GET

Authorization: Bearer <token>

Response:


```
{
   [
  {
     "name": "309",
     "description": "sda",
     "coach": "sad",
     "start_date_time": "2022-11-24T23:45:11Z",
     "end_time": "23:45:14",
     "enrolled": 1
  }
]
}
```


Class List

Users will be able to view class list based on their studio of choice. The response will return all the class sessions available for that particular studio.

Endpoint: /classes/<int:id>/class-list/ **Method:** GET

Authorization: Bearer <token>

Response:


```
{
   [
  {
     "name": "309",
     "description": "sda",
     "coach": "sad",
     "start_date_time": "2022-11-24T23:45:11Z",
     "end_time": "23:45:14",
     "enrolled": 2
  }
  ]
}
```

Subscriptions

Models

Subscription

- plan_name: CharField for subscription plan's name, primary_key, maximum of 500 characters
- plan_period: CharField for subscription plan's period with 2 choices defined by PLAN_PERIOD, maximum of 32 characters, default is MONTHLY
- price: CharField for subscription plan's price, maximum of 100 characters

UserPayment

- plan: ForeignKey referencing the subscription
- user: ForeignKey referencing the user
- amount: CharField for user payment amount, maximum of 100 characters
- payment_card_number: CharField for user's card number, maximum of 25 characters
- payment_security_code: CharField for user's card security code, maximum of 25 characters
- payment_exp_date: CharField for user's card expiry date, maximum of 25 characters
- payment_date: DateTimeField for payment date
- future_payment_date: DateTimeField for the future payment date

Get All Subscriptions/Plans

Users can get all subscriptions which admins create. The response has the name of the name of the plan, whether the plan is yearly or monthly, and the price of the plan.

Endpoint: /subscriptions/all/ **Method:** GET

Authorization: Bearer <token>

Response:

```
[
  {
     "plan_name": "string",
     "plan_period": "string",
     "price": "string"
  },
 {
     "plan_name": "string",
     "plan_period": "string",
     "price": "string"
```

```
  },
]
```

Get a Specific Subscription/Plan

Users can view specific subscriptions which admins create. The response has the name of the name of the plan, whether the plan is yearly or monthly, and the price of the plan.

Endpoint: subscriptions/<str:plan_name>/ **Method:** GET

Authorization: Bearer <token>

Response:

```
[
  {
     "plan_name": "string",
     "plan_period": "string",
     "price": "string"
  }
]
```

Subscribe to a Subscription/Plan

Users can subscribe to a plan. They must enter their card details (card number, security code, and expiry date). The response is their chosen plan, their userid, the amount of the plan, their card details, their current payment date and future payment date.

Endpoint: /subscriptions/<str:plan_name>/subscribe/ **Method:** POST

Authorization: Bearer <token>

Payload:

```
[
  {
  "payment_card_number": "string",
  "payment_security_code": "string",
  "payment_exp_date": "string",
   }
]
```

Response:

```
[
  {
  "plan": "string",
  "user": id,
```

```
  "amount": "string",
  "payment_card_number": "string",
  "payment_security_code": "string",
  "payment_exp_date": "string",
  "payment_date": "datetime",
  "future_payment_date": "datetime"
   }
]
```

Update Payment

Users can update their card details. The response is their chosen plan, their userid, the amount of the plan, their updated card details, their current payment date and future payment date.

Endpoint: /subscriptions/payment/update/ **Method:** PUT

Authorization: Bearer <token>

Payload:

```
[
  {
  "payment_card_number": "card number",
  "payment_security_code": "card security code",
  "payment_exp_date": "card expiry date",
   }
]
```

Response:

```
[
  {
  "plan": "string",
  "user": id,
  "amount": "string",
  "payment_card_number": "string",
  "payment_security_code": "string",
  "payment_exp_date": "string",
  "payment_date": "datetime",
  "future_payment_date": "datetime"
   }
]
```

View Payment History

Users can see their payment history. The response is their chosen plan, their userid, the amount of the plan, their card details, their payment date and future payment date for that period.

Endpoint: /subscriptions/payment/history/ **Method:** GET

Authorization: Bearer <token>

Response:

```
[
  {
  "plan": "string",
  "user": id,
  "amount": "string",
  "payment_card_number": "string",
  "payment_security_code": "string",
  "payment_exp_date": "string",
  "payment_date": "datetime",
  "future_payment_date": "datetime"
   },
  {
  "plan": "string",
  "user": id,
  "amount": "string",
  "payment_card_number": "string",
  "payment_security_code": "string",
  "payment_exp_date": "string",
  "payment_date": "datetime",
  "future_payment_date": "datetime"
   }
]
```

Update Subscription/Plan

Users can update their plan. The response is the name of the updated plan, the amount it costs, and the future payment date.

Endpoint: /subscriptions/<int:id>/update/ **Method:** PUT

Authorization: Bearer <token>

Payload:

```
 {
  "plan": "string",
  }
```

Response:

```
[
  {
  "plan": "string",
  "amount": "string",
  "future_payment_date": "datetime”
  }
]
```

View your Subscription/Plan Details

Users can view their plan details. The response is the name of the updated plan, the amount it costs, and the future payment date.

Endpoint: /subscriptions/<int:id>/details/ **Method:** GET

Authorization: Bearer <token>

Response:

```
[
  {
  "plan": "string",
  "amount": "string",
  "future_payment_date": "datetime”
  }
]
```

Unsubscribe from a Subscription/Plan

Users can unsubscribe from a plan, thus ceasing further payments. The response indicates that they have unsubscribed from their plan.

Endpoint: /subscriptions/<int:id>/cancel/ **Method:** DELETE

Authorization: Bearer <token>

Response:

```
[
  You have unsubscribed.
]
```