MyShelf API

Expected Functionality

Get all books

Request: `GET /api/books/`
- A book can only have one course at the moment.
- Images are not implemented

Response:

```
{
  "success": True,
  "data": [
          {
            "id" : 1,
            "title" : "Introduction to Calculus",
            "course" : "MATH 1110",
            "image" : "",
            "listings" : [0, 3, 43, 192]
          },
          {
            "id" : 2,
            "title" : "Algorithm Design",
            "course" : "CS 4820",
            "image" : "",
            "listings" : [29, 323, 4, 13]
          }
        ]
}
```

Get books by course

Request: `GET /api/books/course/{string:course name}/`

- By course name, the method expects "CS%201110", not "Introduction to Computing Using Python". The &20 stands in for a space.
- If there are no books for that course, the method returns an empty list rather than an error.

Response:

```
{

  "success": True,

  "data": <List of dict representations of books, as above>

}
```

Get book by title

Request: `GET /api/books/book/{string: book title}/`

- Here the title would be "Introductory%20Calculus"
- Always a one element list.

Response:

```
{

  "success": True,

  "data": [{

      "id" : 1,

      "title" : "Introductory Calculus",

      "course" : "MATH 1110",

      "image" : "",

      "listings" : [12, 224, 23]

    }]

}
```

Get book by ID

Request: `GET /api/books/book/id/{int: book id}/`

- Here the id would be 0.
- Always a one element list

Response:

```
{
  "success": True,
  "data": [{
      "id" : 1,
      "title" : "Introductory Calculus",
      "course" : "MATH 1110",
      "image" : "",
      "listings" : [12, 224, 23]
    }]
}
```

Get user by ID

Request: `GET /api/user/{string: net ID}/`
- This method gets the user by their numeric ID in the database or their net ID, depending on whether whether the string passed to the method contains letters in it.
- Profile picture not currently implemented
- Note that the listings for a user are stored by their ID, not their full dict representation

Response:
```
{
  "success": True,
  "data": [{
          "id" : 1,
          "name" : Hartek Sabharwal,
          "netid" : hs786,
          "pfp" : "",
          "listings" : [0, 10, 283, 392]
        }]
}
```

Get listing by ID

Request: `GET /api/listing/{int: listing ID}/`
- Error if the listing does not exist.

Response:

```
{

  "success": True,

  "data": [{

          "id" : 3,

          "title" : "Introduction to Statistics",

          "price" : "27.83",

          "condition": "good",

          "notes" : "My dog ate the front cover.",

          "image" : "",

          "course" : "STSCI 2100",

          "seller" : 10,

          "book" : 273

       }]

}
```

Get listings by user

Request: `GET /api/listings/user/{string: net ID}/`
- Error if the user does not exist. Empty list if the user is not selling anything.

Response:

```
{

  "success": True,

  "data": [

          {

            "id" : 1,

            "title" : "Introduction to Statistics",

            "price" : "27.83",
```

```
            "condition": "good",
            "notes" : "My dog ate the front cover.",
            "image" : "",
            "course" : "STSCI 2100",
            "seller" : 10,
            "book" : 273
        },
        {
            "id" : 29,
            "title" : "Algorithm Design",
            "price" : "3.23",
            "condition": "okay",
            "notes" : "My friend drew a thing on a lot of th
e pages.",
            "image" : "",
            "course" : "CS 4820",
            "seller" : 10,
            "book" : 932
        }
    ]
}
```

Get listings by book title

Request: `GET /api/listings/book/{string: book title}/`
- Error if the title does not exist in the database. Empty list if there are no active listings for the book, but the book is in the database.

Response:

```
{
    "success": True,
    "data": [
```

```json
        {
          "id" : 1,
          "title" : "Algorithm Design",
          "price" : "2.30",
          "condition": "pretty bad ngl",
          "notes" : "I had a nosebleed on page 80.",
          "image" : "",
          "course" : "CS 4820",
          "seller" : 111,
          "book" : 932
        },
        {
          "id" : 29,
          "title" : "Algorithm Design",
          "price" : "3.23",
          "condition": "okay",
          "notes" : "My friend drew a thing on a lot of th
e pages.",
          "image" : "",
          "course" : "CS 4820",
          "seller" : 10,
          "book" : 932
        }
      ]
}
```

Create a user

Request: `POST /api/users/`
- The `pfp` argument in the body is optional

Body:
```
{
```

```
   "name": "Hartek Sabharwal",

   "netid": "hs786",

   "pfp" : "/images/users/hs786.png"

 }
```

Response:
```
{

   "success": True,

   "data": {

            "id" : 1,

            "name" : Hartek Sabharwal,

            "netid" : hs786,

            "pfp" : "/images/users/hs786.png",

            "listings" : []

          }

 }
```

Add a book

Request: `POST /api/book/`
- The `pfp` argument in the body is optional
- Returns error if a book with this title already exists.

Body:
```
{

   "title": "Slaughterhouse Five",

   "course": "ENGL 2060",

   "image" : "/images/books/slaughterhousefive.png"

 }
```

Response:
```
{

   "success": True,
```

```
    "data": {
        "id": 4,
        "title": "Slaughterhouse Five",
        "course": "ENGL 2060",
        "image": "/images/books/slaughterhousefive.png",
        "listings": []
    }
}
```

Add a listing

Request: `POST /api/listings/`
- The condition, image, and notes fields are optional.
- Might add author and edition field at some point?
- Error if the user does not exist.
- Automatically adds the book to the book database if not already present.

Body:
```
{
  "title" : "Algorithm Design",
  "netid" : "hs786",
  "course" : "CS 4820",
  "price" : "27.29",
  "condition" : "ehh",
  "notes" : "My friend drew a thing on some of the pages.",
  "image" : "/images/books/algorithm_design.png"
}
```

Response:
```
{
  "success": True,
  "data": {
      "id" : 1,
```

```
        "title" : "Algorithm Design",

        "course" : "CS 4820",

        "price" : "27.29",

        "condition": "ehh",

        "notes" : "My friend drew a thing on some of the page
s.",

        "image" : "/images/books/algorithm_design.png",

        "seller" : 1,

        "book" : 2

    }

}
```

Delete listing by ID

Request: `DELETE /api/listing/{int: listing ID}/`
- Error if the listing doesn't exist to begin with.
- The book and the user stay in the database.

Response:

```
{

  "success": True,

  "data": {

        "id" : 29,

        "title" : "Algorithm Design",

        "course" : "CS 4820",

        "price" : "27.29",

        "condition": "ehh",

        "notes" : "My friend drew a thing on some of the page
s.",

        "image" : "/images/books/algorithm_design.png",

        "seller" : 1,

        "book" : 2

    }
```

```
    }
```

Delete user by ID

Request: `DELETE /api/user/{int: listing ID}/`

- Error if the user doesn't exist to begin with.
- The listings the user made are deleted as well.

Response:

```json
{
   "success": true,
   "data": {
      "id": 1,
      "name": "Hartek Sabharwal",
      "netid": "hs786",
      "pfp": "/images/users/hs786.png",
      "listings": [1]
   }
}
```

Delete book by ID

Request: `DELETE /api/book/{int: listing ID}/`

- Error if the book doesn't exist to begin with.
- The listings of the book are deleted as well.

Response:

```json
{
   "success": true,
   "data": {
      "id": 5,
      "title": "Pride and Prejudice",
      "course": "CS 4820",
      "image": "",
```

```
    "listings": []
  }
}
```