Repost API Test

This is a complete test for the Repost API. It's designed to run through every possible API endpoint and test all possible responses.

The test is designed to communicate with the API using HTTP calls. As such, to perform the test a deployed version of the API must be running. Every endpoint will be tested in an order that makes sense for resource dependencies. As such, the complete test will eventually create and test operations with all resources, until it finally deletes them all. All of the operations mentioned here are tested both positively and negatively.

Installation

Python 3 must be installed and accessible through the use of a terminal and the keyword `python` or `python3`. Below are the steps for a proper setup using VENV (Python Virtual Environment).

1. Clone the repository:

```
git clone https://github.com/pckv/repost-apitest.git
```

2. Navigate to the `repost-apitest` directory and create a new VENV:

```
cd repost-apitest
python -m venv venv
```

3 (**Linux**). Activate the venv (alternatively: run all commands after this step prefixed with `venv/bin/`)

```
source venv/bin/activate
```

3 (**Windows**). Activate the venv (alternatively: run all commands after this step prefixed with `venv\Scripts\`)

```
venv\Scripts\activate
```

4. Install the required packages

```
pip install -r requirements.txt
```

Running the test

To run the full API test, run the following command with the VENV activated (or use the prefix from step 3):

```
python main.py http://localhost:8000
```

Replace *localhost* and *port* with the address and port of your deployed API.

Running multiple tests for benchmarking

The test can be run more than once using the `--runs` argument. The result will then show stastitics of the tests, such as average test run time, standdard deviation etc.

NOTE: This is not a very good benchmark as it only runs the above test multiple times, which is not concurrent. The benchmarking is only included as a proof of concept, and might work better if multiple runs could be executed concurrently.

```
python main.py http://localhost:8000 --runs 100
```