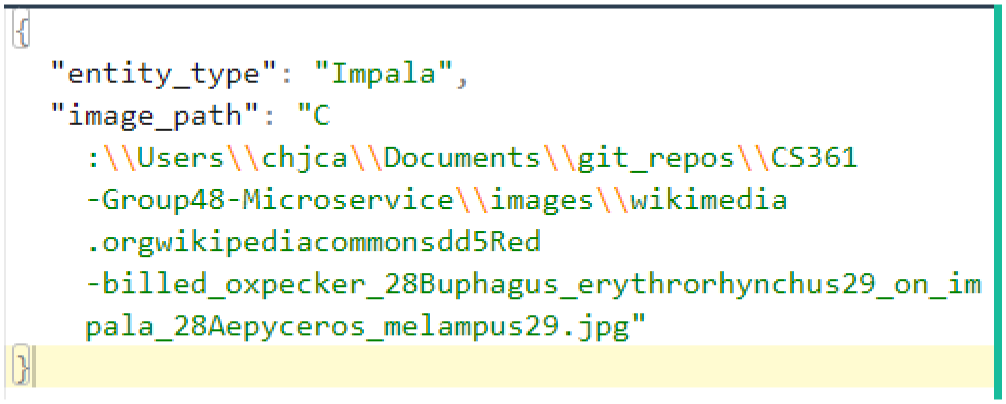
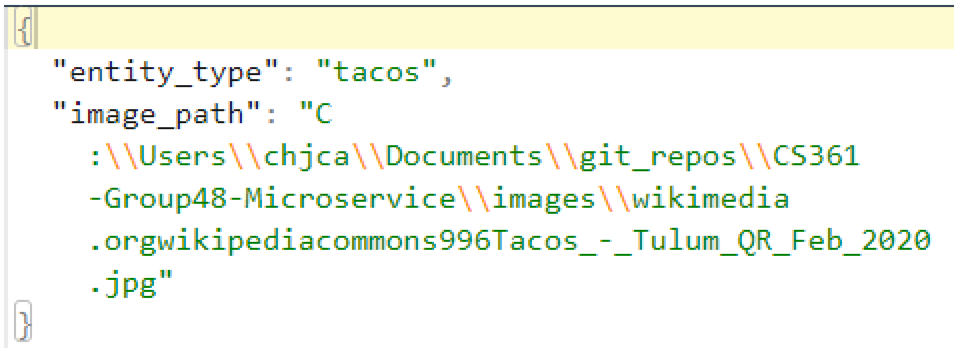
CS361-Group48-Microservice: Random Pet Generator

Description: A microservice that generates random animal values and images.

---

1. **Communication Contract:** There are 2 endpoints which you can use to generate images.
    a. **Endpoint #1:** http://localhost:2024/generateRandomPet
        i. **Requesting Data:** Use **GET** request for this endpoint to select a random animal from pet_config.txt and fetch an associated image from wikimedia commons.
        ii. Receiving Data: The microservice will return a JSON payload containing the “entity_type” and an “image_path” like such:
        iii. 

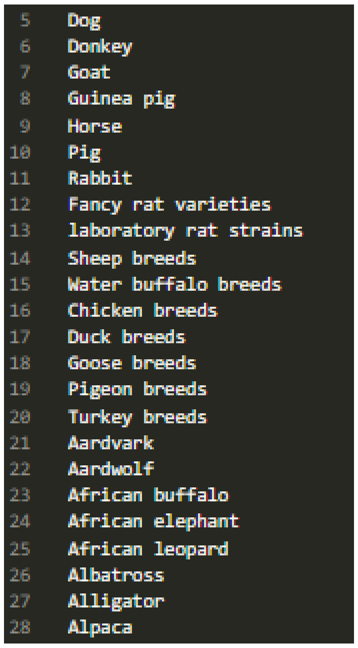
```
{
  "entity_type": "Impala",
  "image_path": "C
    :\\Users\\chjca\\Documents\\git_repos\\CS361
    -Group48-Microservice\\images\\wikimedia
    .orgwikipediacommonsdd5Red
    -billed_oxpecker_28Buphagus_erythrorhynchus29_on_im
    pala_28Aepyceros_melampus29.jpg"
}
```

        iv. Images will be downloaded from wikimedia commons to the microservice’s “image” directory. The file path is relative so the folder structure only needs the image directory in the repo folder like such: “CS361-Group48-Microservice\images\”
    b. **Endpoint #2:** http://localhost:2024/generateImage/***[search for image]***
        i. Requesting Data: Use GET request for this endpoint and replace “[search for image]” with the term you would like to search for. Example endpoint call: “*http://localhost:2024/generateImage/tacos*”
        ii. Receiving Data: Similarly to the first endpoint, the microservice will return a JSON payload containing the “entity_type” and “image_path” to the downloaded image:
        iii. 

```
{
  "entity_type": "tacos",
  "image_path": "C
    :\\Users\\chjca\\Documents\\git_repos\\CS361
    -Group48-Microservice\\images\\wikimedia
    .orgwikipediacommons996Tacos_-_Tulum_QR_Feb_2020
    .jpg"
}
```

CS361-Group48-Microservice: Random Pet Generator

How to set up the microservice

1. **Prerequisites:** You will need python version 3.12.0 or later. You might be able to get away with an earlier version but this is what I have on my local computer so to be safe use 3.12 or later.
2. **Required libraries:**
    a. *flask*
    b. *flask_cors*
    c. *colorama*
    d. *requests*
3. You can either manually install these libraries by running “pip install [lib name]”, example: “pip install flask”.
4. Alternatively, if you are on a windows OS machine, you can run the powershell script named “install_libs.ps1” that is given in the repo and it will run the pip installs all at once. Make sure to run the powershell script as administrator.
5. **pet_config.TXT**
    a. This is a text based configuration file that the microservice uses to “randomly” select an animal. It currently has a giant list of about 520 animals which are selected at random when using /generateRandomPet. This list can be changed to whatever you would like but it should be a non-empty list that is separated by a newline:
    b. Example:
    c. 

```
Dog
Donkey
Goat
Guinea pig
Horse
Pig
Rabbit
Fancy rat varieties
laboratory rat strains
Sheep breeds
Water buffalo breeds
Chicken breeds
Duck breeds
Goose breeds
Pigeon breeds
Turkey breeds
Aardvark
Aardwolf
African buffalo
African elephant
African leopard
Albatross
Alligator
Alpaca
```

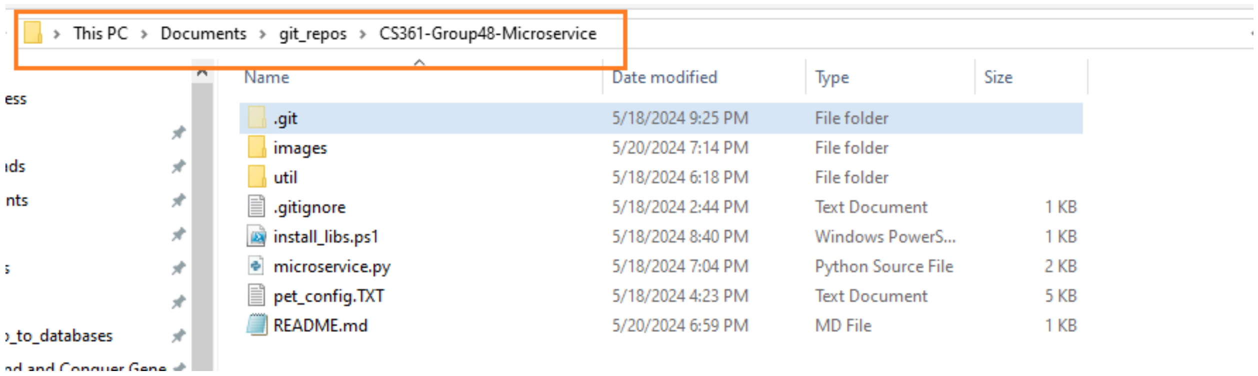
CS361-Group48-Microservice: Random Pet Generator

Launching the microservice

Once all the prerequisites are installed, from the app root folder run “python microservice.py” from the console.

I like to run my apps using cmd, but any terminal should work as long as it can reach the python.exe. I tend to have issues with running my vs code terminal so I use git bash or cmd most of the time.

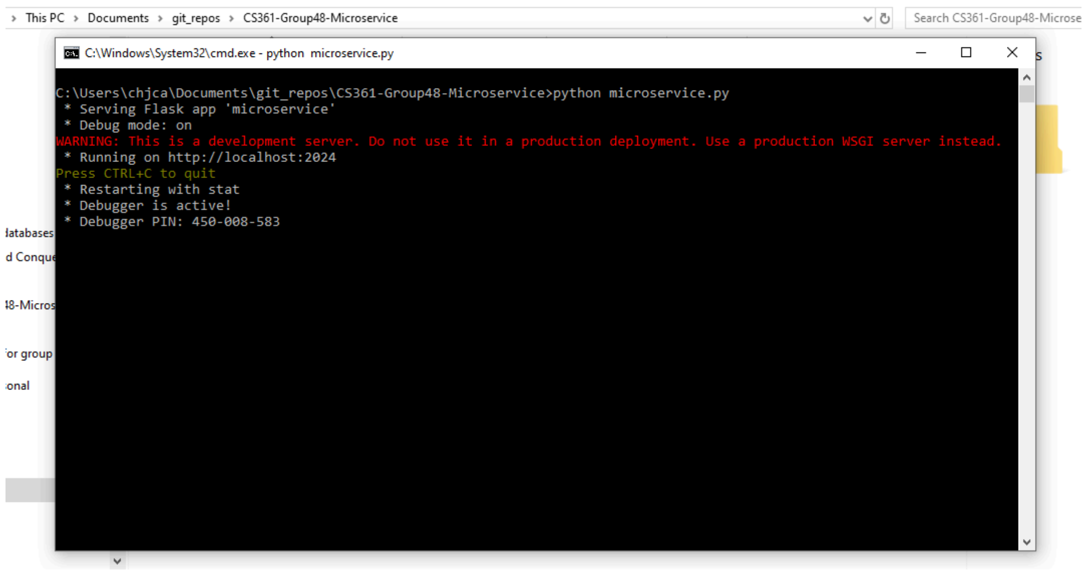
1. To run using cmd, go to the app root directory and type “cmd” where the orange box is located and hit enter:



This PC › Documents › git_repos › CS361-Group48-Microservice

Name	Date modified	Type	Size
.git	5/18/2024 9:25 PM	File folder	
images	5/20/2024 7:14 PM	File folder	
util	5/18/2024 6:18 PM	File folder	
.gitignore	5/18/2024 2:44 PM	Text Document	1 KB
install_libs.ps1	5/18/2024 8:40 PM	Windows PowerS...	1 KB
microservice.py	5/18/2024 7:04 PM	Python Source File	2 KB
pet_config.TXT	5/18/2024 4:23 PM	Text Document	5 KB
README.md	5/20/2024 6:59 PM	MD File	1 KB

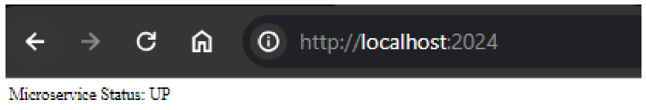
2.
3. This will open a cmd window. Type “python microservice.py” to start the flask application. Output should look similar to below:



C:\Windows\System32\cmd.exe - python microservice.py

```
C:\Users\chjca\Documents\git_repos\CS361-Group48-Microservice>python microservice.py
 * Serving Flask app 'microservice'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://localhost:2024
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 450-008-583
```

4.
5. Using your web browser, navigate to to “http://localhost:2024/”, the microservice should return a status of “UP” like shown below:



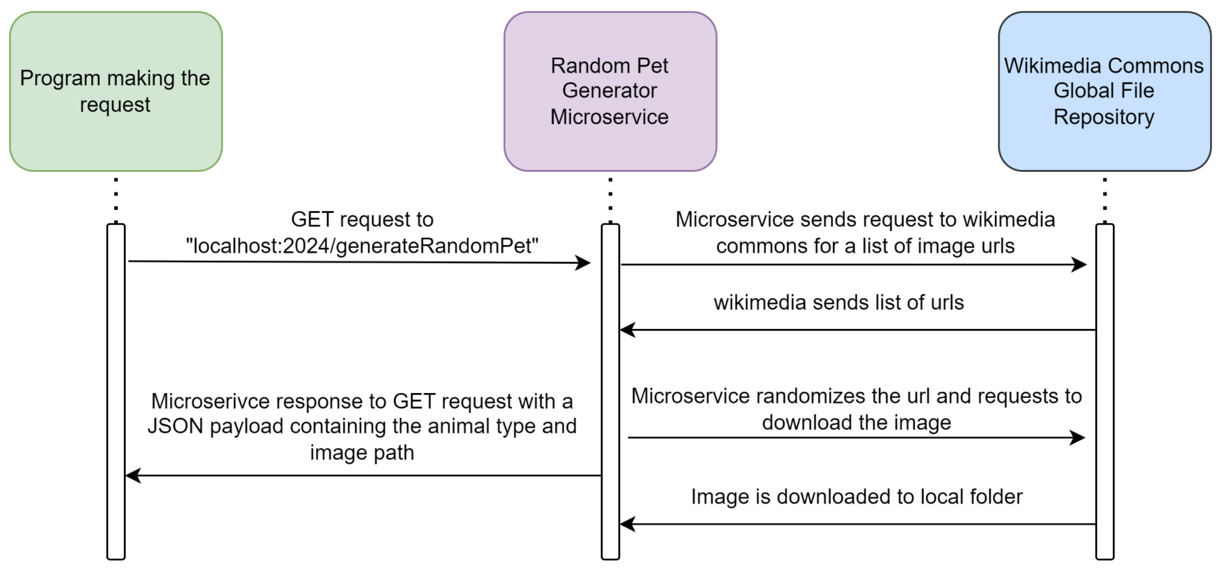
http://localhost:2024

Microservice Status: UP

6.

CS361-Group48-Microservice: Random Pet Generator

UML Sequence Diagram


Program making the request
Random Pet Generator Microservice
Wikimedia Commons Global File Repository
GET request to "localhost:2024/generateRandomPet"
Microservice sends request to wikimedia commons for a list of image urls
wikimedia sends list of urls
Microserivce response to GET request with a JSON payload containing the animal type and image path
Microservice randomizes the url and requests to download the image
Image is downloaded to local folder