

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:

```
"entity_type": "Impala",  
"image_path": "C  
:\\Users\\chjca\\Documents\\git_repos\\CS361  
-Group48-Microservice\\images\\wikimedia  
.org\\wikimedia\\commons\\dd5Red  
-billed_oxpecker_28Buphagus_erythrorhynchus29_on_im  
pala_28Aepyceros_melampus29.jpg"
```

- iii.
- 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/\[search for image\]](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:

```
"entity_type": "tacos",  
"image_path": "C  
:\\Users\\chjca\\Documents\\git_repos\\CS361  
-Group48-Microservice\\images\\wikimedia  
.org\\wikimedia\\commons\\996Tacos_-_Tulum_QR_Feb_2020  
.jpg"
```

- iii.

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.

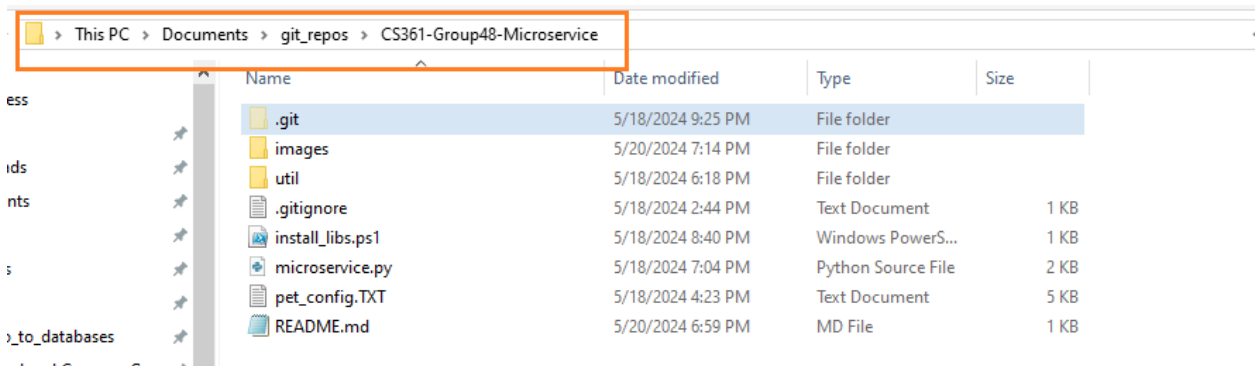
CS361-Group48-Microservice: Random Pet Generator

Launching the microservice

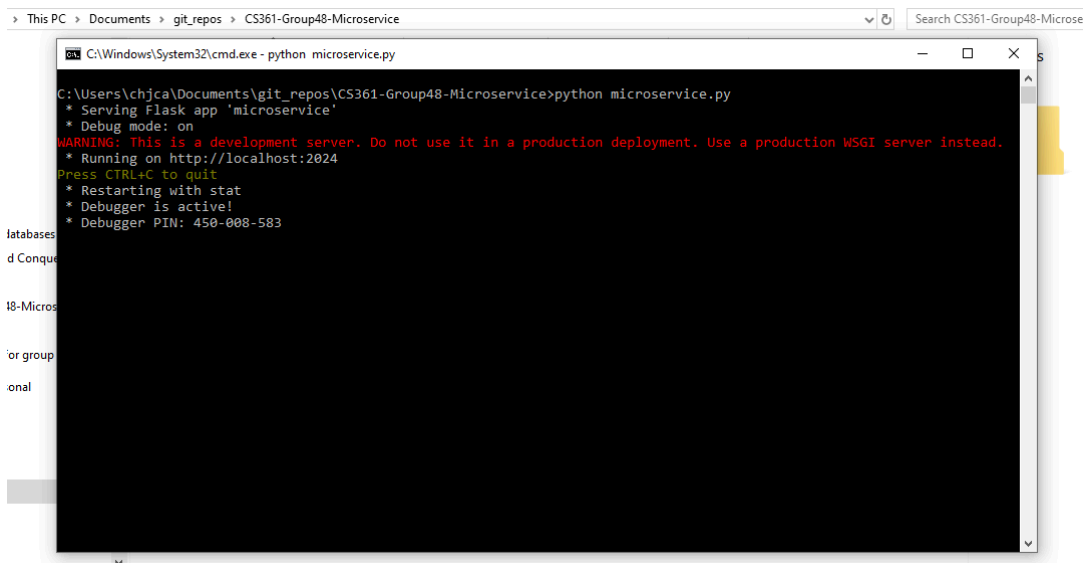
Once all the prerequisites are installed, from the app root folder run “python microservice.py” from the console.

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 bash or cmd most of the time.

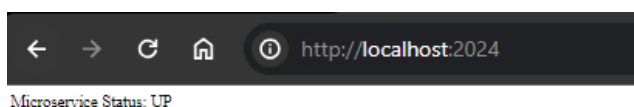
1. To run using cmd, go to the app root directory and type “cmd” where the orange box is located and hit enter:



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



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



- 6.

CS361-Group48-Microservice: Random Pet Generator

UML Sequence Diagram

