README:

This micro service generates a JSON array of random animal or color names of a user requested length.

Communication Contract:
This service is set to run forever on OSU flip servers and be accessible at any time. If at any time the service is not accessible for use via the instructions below, my partner can text me and I will plan to have it running again within 24 hours.

Requesting Data:
The micro service is hosted on OSU flip server 4 so any client must have access to that server. For an OSU student they must be logged in with the CISCO AnyConnect App at vpn.oregonstate.edu, by using their OSU credentials.

Then to request data from the micro service, the client must make a fetch request to: http://flip4.engr.oregonstate.edu:9105/animals

The client must set the number of random names requested as the query parameter: "amount". For example a fetch request to the url:

http://flip4.engr.oregonstate.edu:9105/animals?amount=5

Returns a JSON array of random animal names of length 5.

Similarly a fetch request to the url:

http://flip4.engr.oregonstate.edu:9105/color?amount=3

Returns a JSON array of random color names of length 3.

The maximum possible number of colors that can be requested is 13. The maximum number of animals that can be requested is 224.

If the amount parameter is not set, the entire array will be returned. If the amount parameter is set to 0, an empty array will be returned.

Fetch Requests with Node JS:
Here is an example of requesting data using node.js to request animal names within a node app.

```
async function getRandomAnimalNames(amount){

    const response = await fetch(`http://flip4.engr.oregonstate.edu:9105/animals?amount=${amount}`)

    animalNamesArray = await response.json();

    return animalNamesArray;

}


animalNames = getRandomAnimalNames(4).then((res) => {
    console.log(res);
    return res;
});
```

In the example above we create an asynchronous function that takes an amount argument and uses the fetch protocol to return an array of length amount. We then call the function as the value for the variable “animalNames”. When the promise is fulfilled animalNames is set to the returned array of random animal names:

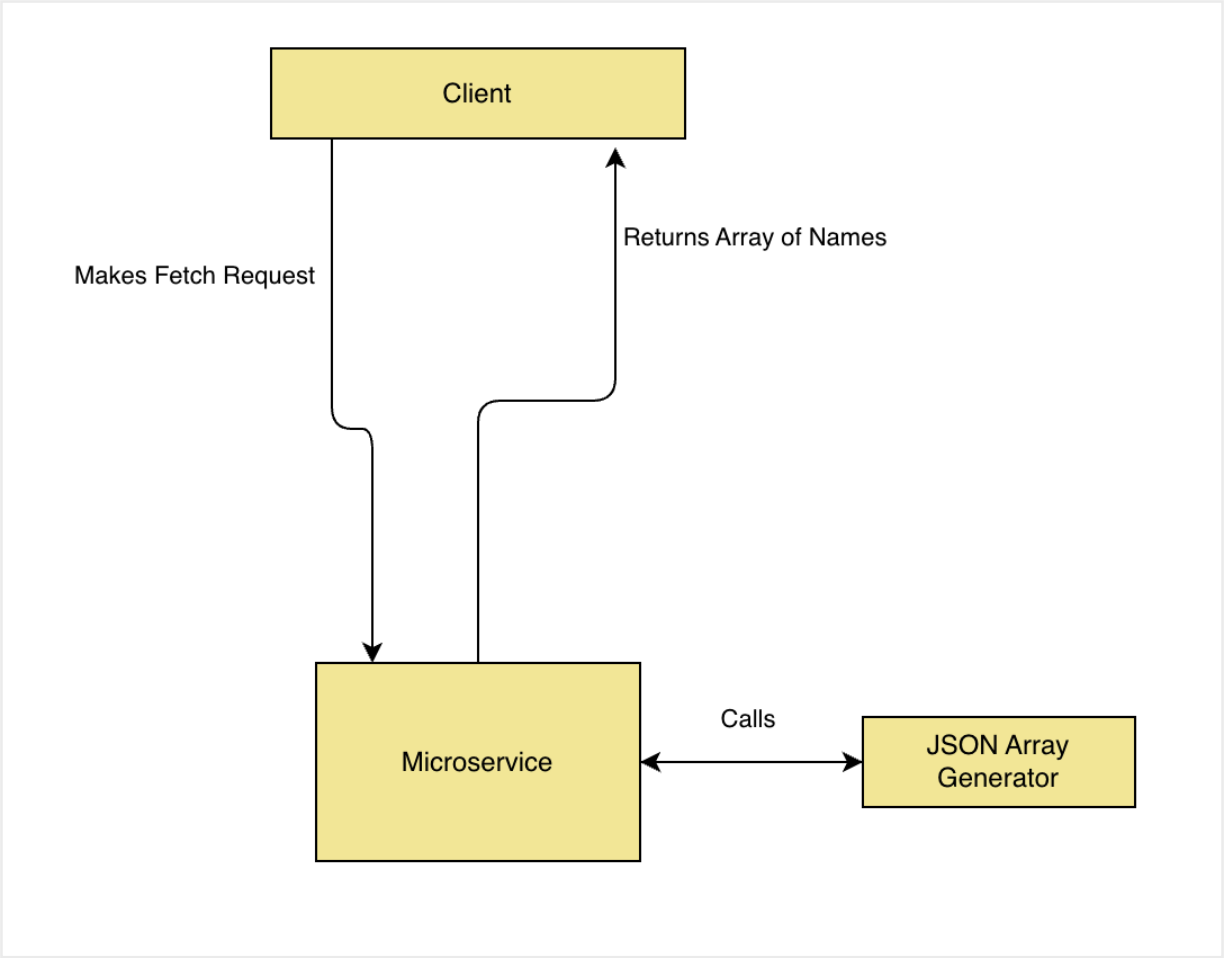
```
Express started on http://localhost:9105; press Ctrl-C to terminate.
[ 'Cobra', 'Mule', 'Rat', 'Baboon' ]
```

To use fetch in node follow the instructions here:
https://www.npmjs.com/package/node-fetch#installation

For me I terminal installed fetch:
npm install node-fetch@2

And used this statement to implement:
const fetch = require("node-fetch");

UML Diagram:


Client
Makes Fetch Request
Returns Array of Names
Microservice
Calls
JSON Array Generator