API Documentation

This API provides information on cafes, allows users to search for cafes, add new cafes, and submit feedback. It uses a JSON file as a data store.

Submit user feedback

```
POST http://127.0.0.1:8000/submit-feedback
```

Submitted-feedback path lets users submit feedback with a `POST` request to `/submit-feedback`. The server reads `feedback.json`, adds the new feedback, and saves/writes the updated JSON file. It will return a 200 status and a successful message if the file has been updated . If there is an error along the way, it will send an Error 400 status.

Request body (example input):

```
{ "name": "Rise",
  "feedback": "Lovely place."}
```

name (string): name of the café the user is submitting feedback about

feedback (string): the content of the feedback the user has written

Responses

Successful:

Status: 200 (OK)

```
{ message: "submission has worked" }
```

Error: 400 (Bad Request)

```
{ message: 'feedback has not been saved' }
```

Viewing user feedback

```
GET http://127.0.0.1:8000/feedback
```

The /feedback route lets users receive all feedback by making a `GET` request to `/feedback`. The server reads the `feedback.json` file and sends back the stored feedback. If everything works, it returns a status 200 with the feedback data from the JSON file . If something goes wrong while reading the file, it sends an error 400 message.

Responses

Successful:

Status: 200 (OK)

```
[]
    { "name": "Pig & Pastry",
        "feedback": "Wonderful food and drink!"
    },
    { "name": "Costa",
    "feedback": "Really quick coffees."}
....
  (more feedback elements)
....
]
```

Error: 400 (Bad Request)

```
{ message: "error with reading the file" }
```

Requesting Cafes (and filtered Café options)

```
GET http://127.0.0.1:8000/cafes
GET (OPTIONAL) : http://127.0.0.1:8000/cafes?filter=york (YORK FILTER)
                 http://127.0.0.1:8000/cafes?filter=durham (DURHAM FILTER)
```

The cafes route users receive `cafes.json` details by making a GET request to `/cafes`. The server reads the `cafes.json` file and responds with café data. Users can filter cafes by location using the `filter` query parameter (`york` or `durham`). If everything works, it returns a status 200 message with the relevant filtered cafes from the JSON file. If something goes wrong while reading the file, it sends an error 400 message.

Responses

Cafes (successful):

Status: 200 (OK)

```
[
  {
    "id": "1",
    "name": "Rise Brunch Cafe",
    "location": "Fossgate",
    "coffeetypes": "Espresso, Latte, Cappuccino",
    "reviews": "4.2/5",
    "comments": "Rise Brunch Cafe is the perfect spot..."
  },
  ..... (rest of cafes json file)
  ]
```

York Cafes (successful):

Status: 200 (OK)

```
[
  {
    "id": "13",
    "name": "Coppergate Coffee",
    "location": "York",
    "coffeetypes": "Latte, Espresso, Iced Coffee",
    "reviews": "4.2/5",
    "comments": "Coppergate Coffee is a delightful ..."
  },
  .... (more objects with property location == york)
  ]
```

Durham Cafes:

Status: 200 (OK)

```
[
  {
    "id": "2",
    "name": "Flat White Kitchen",
    "location": "Durham",
    "coffeetypes": "Espresso, Mocha, Flat White",
    "reviews": "4.7/5",
    "comments": "Flat White Kitchen is a must-visit ..."
```

```
  },
  ....(more objects with property location == durham)
]
```

Error: 400 (Bad Request)

```
{ message: 'error reading file' }
```

Requesting Café Details

```
GET http://127.0.0.1:8000/cafe/:id
GET (EXAMPLE) http://127.0.0.1:8000/cafe/11
```

The cafe/:id route allows users to retrieve details on a specific café (when clicked on HTML page) by making a GET request to `/cafe/:id`. The server reads the `cafes.json` file and finds the cafe with the matching ID (each cafe has a unique ID from the addCafe cafe submision). If a matching cafe is found, it returns a status 200 with the cafe's details in JSON format. If the cafe does not exist, it sends a classic 404 error message. If something goes wrong while reading the file, it sends an error 400 message.

Responses

Café of id="11" (successful)

Status: 200 (OK)

```
{
  "id": "11",
  "name": "Leonard’s Coffee House",
  "location": "Durham",
  "coffeetypes": "Americano, Latte, Mocha",
  "reviews": "4.5/5",
  "comments": "Leonard’s Coffee House is a ..."
}
```

Error: 400 (Bad Request)

```
{ message: 'error with reading cafes.json file' }
```

Error: 404 (Cannot be found)

```
{ message: 'cafe cannot be found' }
```

Search Function

```
GET http://127.0.0.1:8000//search?names=query
GET (EXAMPLE) http://127.0.0.1:8000/search?names=flat
```

Request body:

The `/search` route lets users search for cafes by making a GET request to `/search?names=query`. The server reads the `cafes.json` file and filters cafes whose names include the search text. If a match is found, it returns a status 200 with the matching cafes as JSON. If nothing is provided, it will return an empty array. If something goes wrong while reading the file, it sends an error 400 message.

Responses

search query = "flat" (successful)

Status: 200 (OK)


```
[
  {
    "id": "2",
    "name": "Flat White Kitchen",
    "location": "Durham",
    "coffeetypes": "Espresso, Mocha, Flat White",
    "reviews": "4.7/5",
    "comments": "Flat White Kitchen is a must-visit for ..."
  },
  {
    "id": "7",
    "name": "Flat White Cafe",
    "location": "Durham",
```

```
    "coffeetypes": "Flat White, Cortado, Filter Coffee",
    "reviews": "4.8/5",
    "comments": "Flat White Cafe is an essential destination ..."
  }
]
```

Error: 400 (Bad Request)

```
{ message: 'Error reading cafes data' }
```

Adding Cafes

```
POST http://127.0.0.1:8000/addCafe
```

The `/addCafe` route allows users to add a new cafe by making a POST request to `/addCafe`. The server reads the `cafes.json` file, assigns a unique ID to the new cafe, and adds it to the list. If everything works, it returns a status 200 with a success message and the newly added cafe. If required data is missing, it sends an error 400 message. If something goes wrong while updating the file, it sends a different error 400 message.

Request body (example input):

```
{
    "name": "Strawberry Fields Cafe",
    "location": "Escrick, York",
    "coffeetypes": "All Coffees and breakfast",
    "reviews": "3",
    "comments": "Really sweet place, nice customers and lovely staff."
  }
```

name (string): name of a new café the user is submitting

location (string): location of the new café

coffeetypes (string): coffee types the new café offers

reviews (string): numerical rating the user gives for the café

comment (string): worded rating the user gives for the café

Responses

Successful:

Status: 200 (OK)

```
{
  message: "new cafe added",
  "cafe": {
    "id": "46",
    "name": "Strawberry Fields Cafe",
    "location": "Escrick, York",
    "coffeetypes": "All Coffees and breakfast"
    "reviews": "3/5",
    "comments": "Really sweet place, nice customers and lovely staff."
  }
}
```

Error: 400 (Form submission with insufficient data)

```
{ message: 'form is not fully filled out' }
```

Error: 400 (Error updating JSON file)

```
{ message: 'writing to cafe.json error' }
```