DSCI 551 FINAL PROJECT REPORT

TEAM DETAILS

SINGLE PERSON TEAM - GROUP 80

NAME: Vedanvita Gudavalli
USC ID: 1741003325
EMAIL: vgudaval@usc.edu

PROJECT TOPIC

EMULATING FIREBASE

Firebase is a popular backend-as-a-service (BaaS) that provides a variety of services, including a realtime database. This project aims to build a prototype system to emulate some of the main functionalities of Firebase's Realtime Database using Flask, a micro web framework for Python, and MongoDB, a NoSQL database.

We implemented the following CRUD operations using Flask and MongoDB:

GET: Retrieve a listing or all listings with optional filtering and sorting.
POST: Create a new listing with a unique identifier.
PUT: Update a listing or create a new one if it doesn't exist.
PATCH: Update specific fields of a listing.
DELETE: Delete a listing by its identifier.

WORKFLOW

- **Appropriate Dataset is collected and Pre-processing is implemented.**
- **The Dataset is loaded into MongoDB using Python with PyMongo Library.**
- **The proper indexes are set to the MongoDB according to the requests.**
- **A server is created using Flask with appropriate endpoints and restful libraries that are connected with MongoDB.**
- **The windows command terminal is set up to support the CURL commands.**
- **As soon as we enter the command the results are fetched by the restful flask server by calling MongoDB and the results can be seen on the terminal.**

IMPLEMENTATION

- **The dataset used for my project is "New York City Airbnb Open Data" which contains Airbnb listings and metrics in NYC for 2019.**
  The dataset link :
  https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data

- **Sufficient time spent on finalizing the dataset so that the required functions can be applied on the dataset suitably. This dataset includes information about listings, such as the name, host, location, room type, price, minimum nights, number of reviews, last review date, reviews per month, calculated host listings count, and availability.**

- **Data Pre-processing and Data Cleaning is implemented by removing null values and unnecessary columns.**

- **The dataset is a csv file, so I implemented a python script "csv_to_json.py". csv_to_json.py converts the csv data into json format.**

- **All the required installations are done like Flask, PyMongo and MongoDB on my local system.**

- **The dataset is loaded into MongoDB using PyMongo library. Created a python script "import_to_mongo.py" which implements the data loading in MongoDB. This is done only once.**

- **The indices are set for the Data. These fields are chosen to create indexes because we can get efficient queries and more filtering options. So I implemented "indices.py" to achieve the same.**

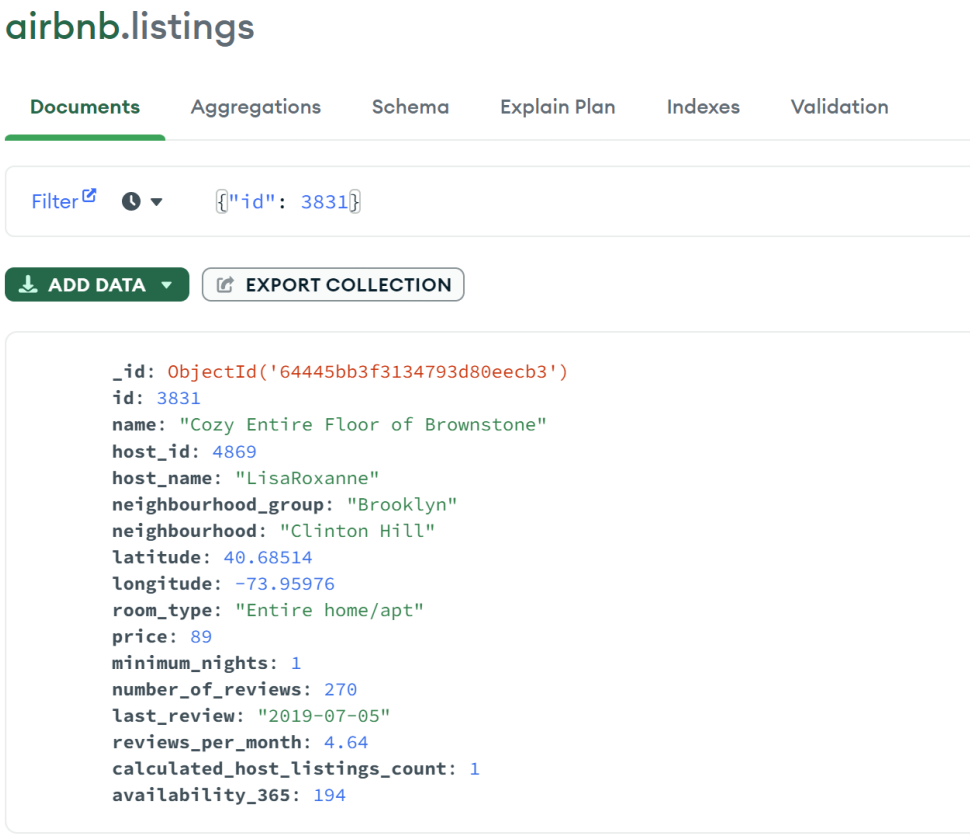
- **Flask server is implemented as "app.py" and all the CRUD operations are implemented like GET, POST, PUT, PATCH and DELETE.**

- **The windows command terminal is set up to support CURL commands and environment variables and system variables are set up accordingly to support the commands.**

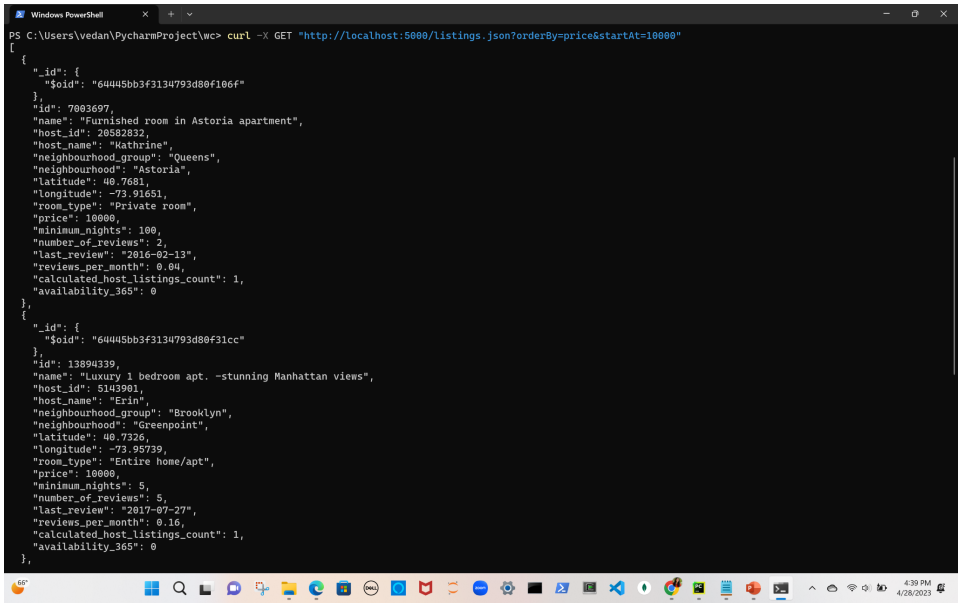
GET

```
PS C:\Users\vedan\PycharmProject\wc> curl -X GET "http://localhost:5000/listings/3831.json"
{
  "_id": {
    "$oid": "64445bb3f3134793d80eecb3"
  },
  "id": 3831,
  "name": "Cozy Entire Floor of Brownstone",
  "host_id": 4869,
  "host_name": "LisaRoxanne",
  "neighbourhood_group": "Brooklyn",
  "neighbourhood": "Clinton Hill",
  "latitude": 40.68514,
  "longitude": -73.95976,
  "room_type": "Entire home/apt",
  "price": 89,
  "minimum_nights": 1,
  "number_of_reviews": 270,
  "last_review": "2019-07-05",
  "reviews_per_month": 4.64,
  "calculated_host_listings_count": 1,
  "availability_365": 194
}
```

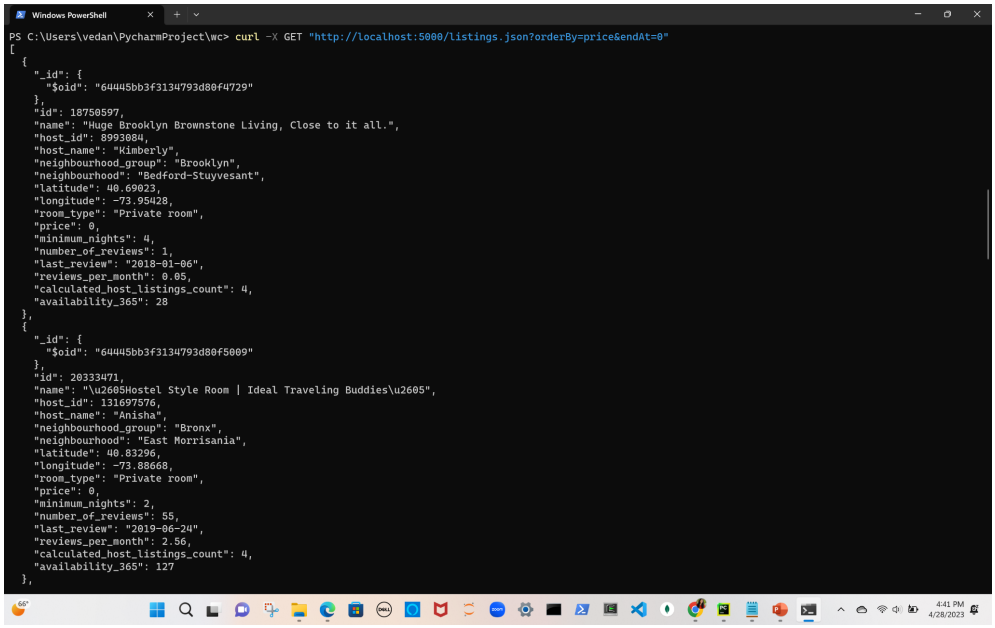
GET command used to get the details of listing id : 3831



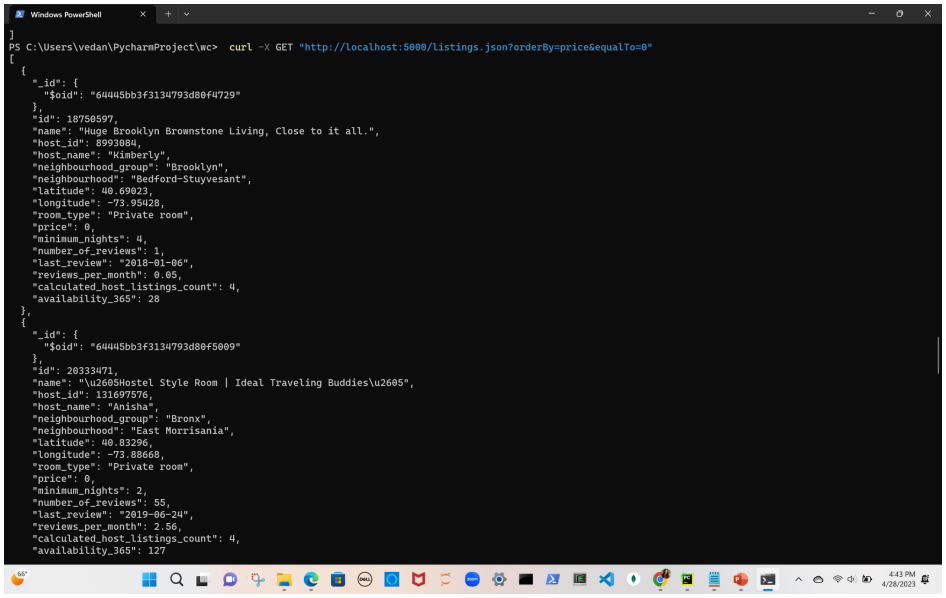
GET



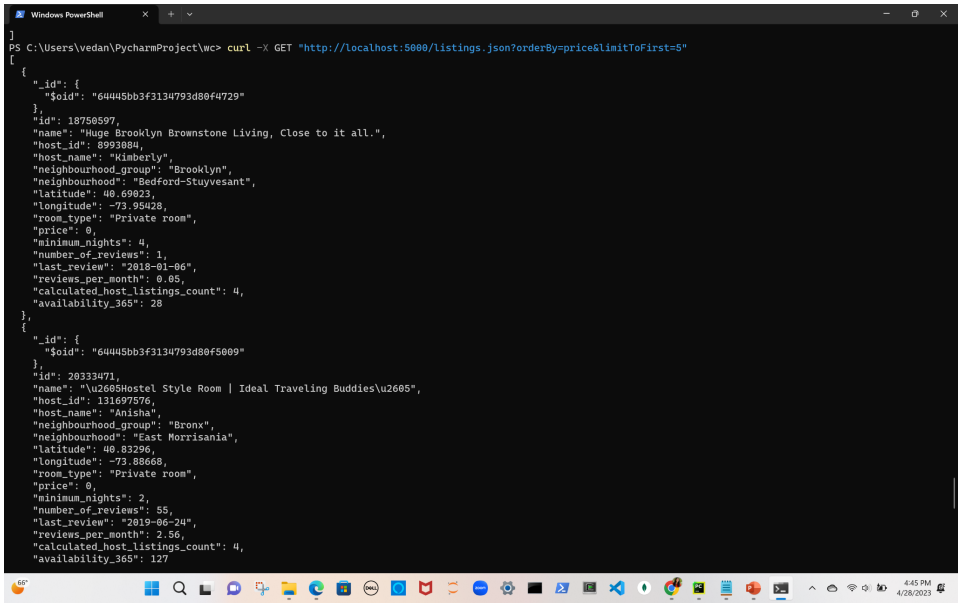
GET command implementation to get the listing which are ordered by price and starts at 10000



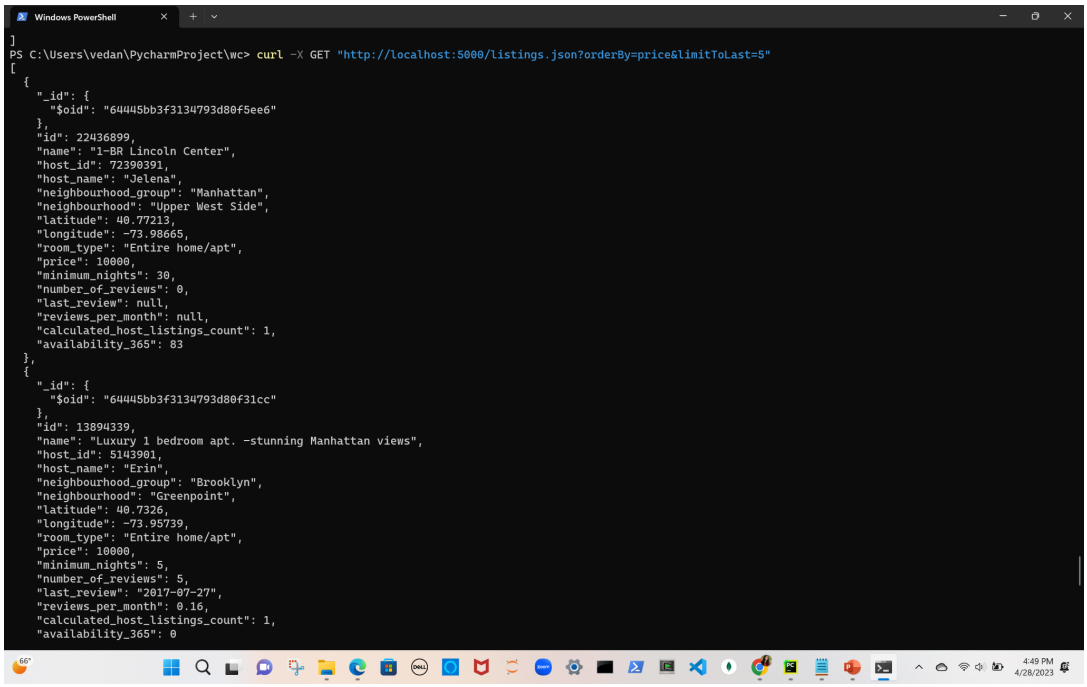
GET command implementation to get the listings which are ordered by price and ends at 0 (price is zero because of coupons)



GET command implementation to get the listings which are ordered by price and equals to 0 (price is zero because of coupons)

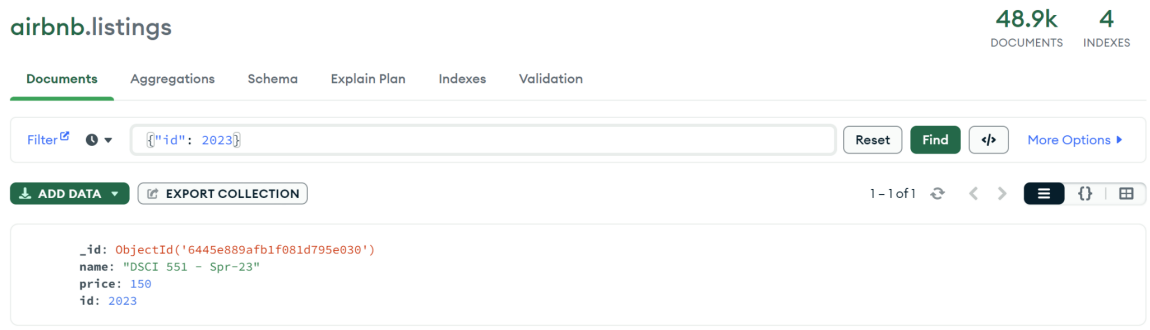


GET command implementation to get the first five listings(The 5 lease priced listings) in ascending order which are ordered by price (price is zero because of coupons).



GET command implementation to get the last five listings(The top 5 highest price listings) in descending order which are ordered by price (price is zero because of coupons).

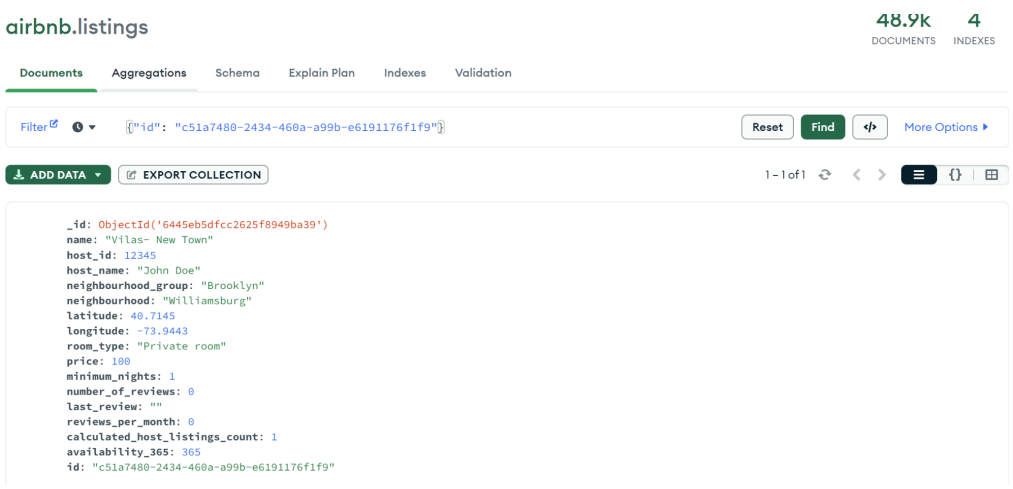
PUT



PUT command implementation to insert a new ID which is 2023.

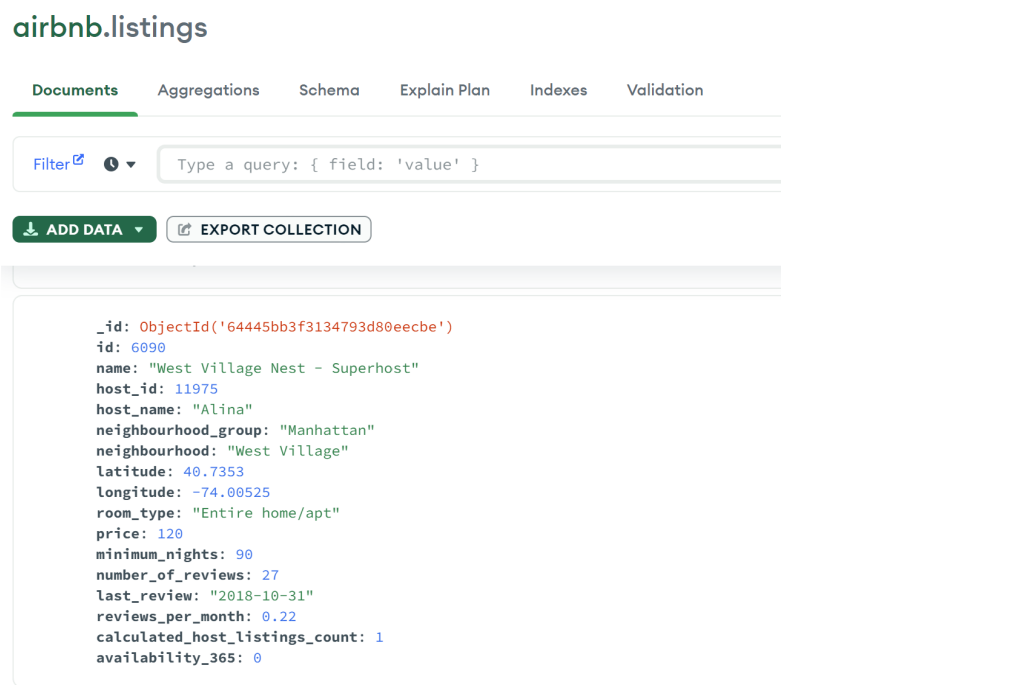
POST

```
PS C:\Users\vedan\PycharmProject\wc> curl -X POST -H "Content-Type: application/json" -d '{\"name\": \"Vilas- New Town\", \"host_id\": 12345, \"host_name\": \
"John Doe\", \"neighbourhood_group\": \"Brooklyn\", \"neighbourhood\": \"Williamsburg\", \"latitude\": 40.7145, \"longitude\": -73.9443, \"room_type\": \"Priv
ate room\", \"price\": 100, \"minimum_nights\": 1, \"number_of_reviews\": 0, \"last_review\": \"\", \"reviews_per_month\": 0, \"calculated_host_listings_count
\": 1, \"availability_365\": 365}' 'http://localhost:5000/listings.json'
{
  "listing_id": "c51a7480-2434-460a-a99b-e6191176f1f9",
  "message": "Listing created successfully."
}
```

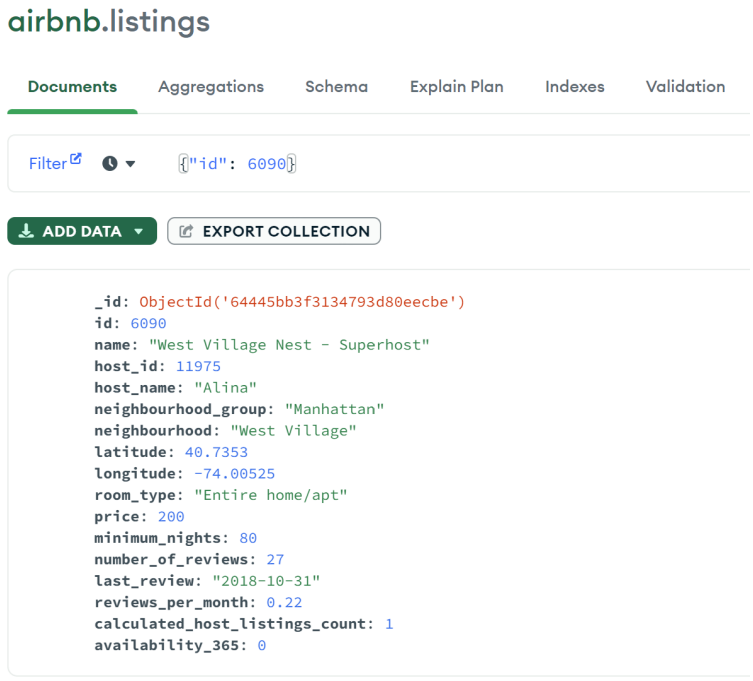


POST command implementation which creates a new listing id .

PATCH

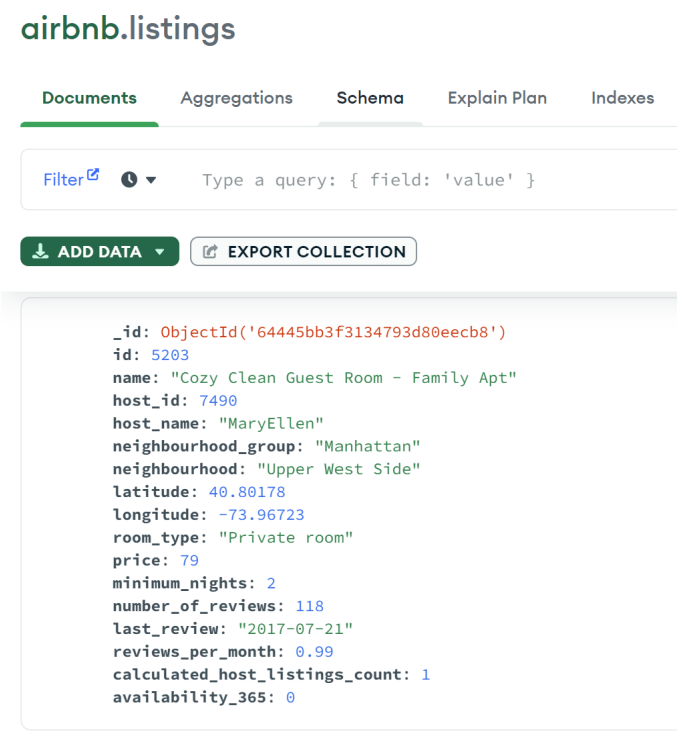


Before updating using PATCH we have a price as 120 and minimum_nights as 90.



After updating using PATCH we have a price as 200 and minimum_nights as 80.

DELETE



```
PS C:\Users\vedan\PycharmProject\wc> curl -X GET "http://localhost:5000/listings/5203.json"
{
  "_id": {
    "$oid": "64445bb3f3134793d80eecb8"
  },
  "id": 5203,
  "name": "Cozy Clean Guest Room - Family Apt",
  "host_id": 7490,
  "host_name": "MaryEllen",
  "neighbourhood_group": "Manhattan",
  "neighbourhood": "Upper West Side",
  "latitude": 40.80178,
  "longitude": -73.96723,
  "room_type": "Private room",
  "price": 79,
  "minimum_nights": 2,
  "number_of_reviews": 118,
  "last_review": "2017-07-21",
  "reviews_per_month": 0.99,
  "calculated_host_listings_count": 1,
  "availability_365": 0
}
PS C:\Users\vedan\PycharmProject\wc> curl -X DELETE 'http://localhost:5000/listings/5203.json'
{
  "result": "Listing deleted successfully."
}
PS C:\Users\vedan\PycharmProject\wc> curl -X GET "http://localhost:5000/listings/5203.json"
{
  "error": "Listing not found."
}
PS C:\Users\vedan\PycharmProject\wc>
```

We can see that the listing id 5203 has been deleted.

LEARNING EXPERIENCES

- **Understanding Firebase's Realtime Database**

By emulating Firebase's Realtime Database, I gained a deeper understanding of its features, capabilities, and limitations. This knowledge can be useful when working with Firebase or other backend-as-a-service platforms in the future.

- **Working with Flask**

This project provided hands-on experience with Flask, a popular micro web framework for Python. I learned how to set up a Flask application, define routes, handle HTTP requests and responses, and work with JSON data.

- **MongoDB and NoSQL Databases**

I learned about MongoDB, a widely-used NoSQL database, and how to interact with it using the pymongo library. Explored the benefits of using a document-based database like MongoDB, such as flexibility in data storage, scalability, and fast querying capabilities.

- **Implementing CRUD operations**

Creating, reading, updating, and deleting records are essential operations for most web applications. This project allowed me to implement and understand these CRUD operations using Flask and MongoDB, which can be applied to other web development projects in the future.

- **Implementing Filtering and Sorting**

Gained experience in implementing advanced filtering and sorting functionality in my RESTful API. This skill can be useful when building APIs that require complex data retrieval and manipulation.

- **Error Handling and Validation**

Throughout the project, I encountered various challenges related to error handling and input validation. I learned the importance of validating user input, handling different types of errors, and providing meaningful error messages to users.

- **Testing and Debugging**

To ensure the correctness and robustness of the restful server, I tested its functionality using various curl commands and analyzed its behavior under different conditions. This experience reinforced the importance of thorough testing and debugging in software development.

This process improved my ability to communicate technical concepts clearly and concisely, which is essential for working in teams and presenting our work to others.

GOOGLE DRIVE LINK

https://drive.google.com/drive/folders/1CJQzEle1WZfsa82C4FNghxP2kJZ9uvwt?usp=share_link

YOUTUBE VIDEO LINK

https://youtu.be/rokZmiU9Hx8