

Cleaning Service

- 1) Here, this project uses Python programming language with virtual environment for creating API's.
- 2) Postman is used for API testing and verifying.
- 3) PostgreSQL database is used.

Using Elephant SQL (<https://www.elephantsql.com/>) , PostgreSQL is implemented.

- 4) GitHub link for this project: [priyanka-garach/CleaningService \(github.com\)](https://github.com/priyanka-garach/CleaningService)
- 5) Zip file also submitted through email.
- 6) Below are the commands needs to be executed in a sequence.

Create virtual environment (write in terminal)

```
python -m venv ./venv
```

#Activate virtual environment

```
./venv/Scripts/activate
```

#Run requirements.txt

```
pip install -r requirements.txt
```

Run file

```
python app.py
```

- 7) For output verification:-

Using Postman and GET

http://127.0.0.1:5000/add_ons/Uppsala

http://127.0.0.1:5000/add_ons/Stockholm

Output:-

The screenshot displays a Postman interface for a GET request. The URL bar shows `http://127.0.0.1:5000/add_ons/Uppsala`. Below the URL bar, tabs for Params, Authorization, Headers (7), Body, Pre-request Script, Tests, and Settings are visible. The 'Body' tab is selected, showing a message: 'This request does not have a body'. Below this, another section shows the response details. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response is a JSON object with keys for 'Balcony_Cleaning', 'Dusting', 'Garbage_Removal', 'Window_Cleaning', and 'message'. The status bar at the bottom right indicates 'Status: 201 C'.

```
1  {
2    "Balcony_Cleaning": "550",
3    "Dusting": "280",
4    "Garbage_Removal": "80",
5    "Window_Cleaning": "300",
6    "message": "These are available addons."
7  }
```

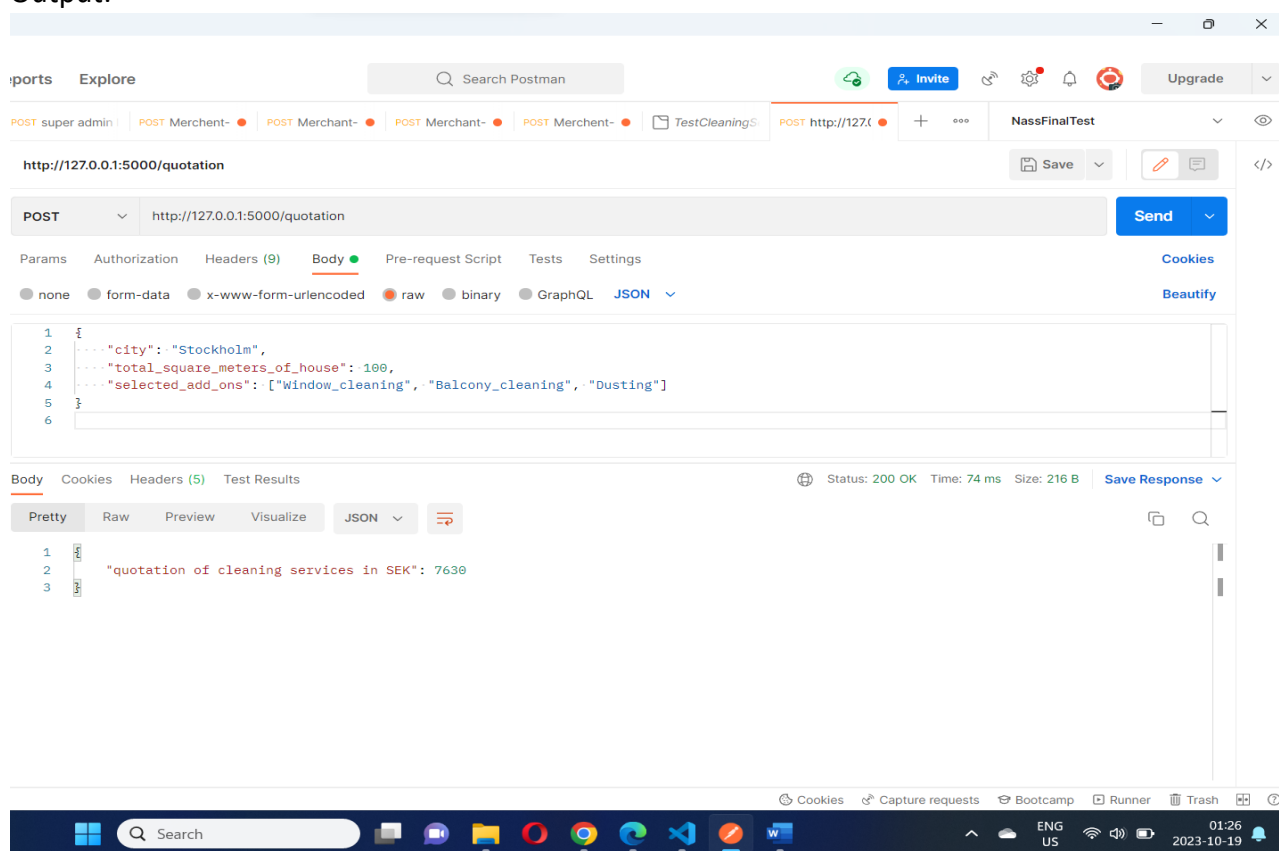
8) # Using Postman and POST execute with payload json

http://127.0.0.1:5000/quotation

Payload-1 (Json)

```
{
  "city": "Stockholm",
  "total_square_meters_of_house": 100,
  "selected_add_ons": ["Window_cleaning", "Balcony_cleaning", "Dusting"]
}
```

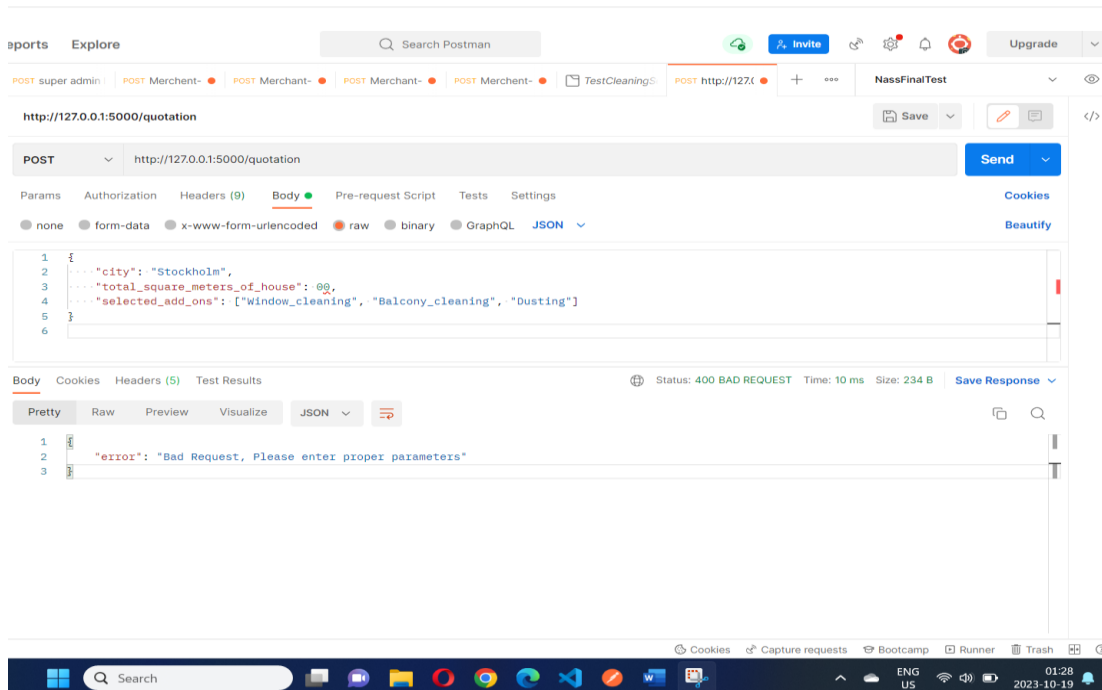
Output:-



Payload-2 (Json)

```
{
  "city": "Stockholm",
  "total_square_meters_of_house": 00,
  "selected_add_ons": ["Window_cleaning", "Balcony_cleaning", "Dusting"]
}
```

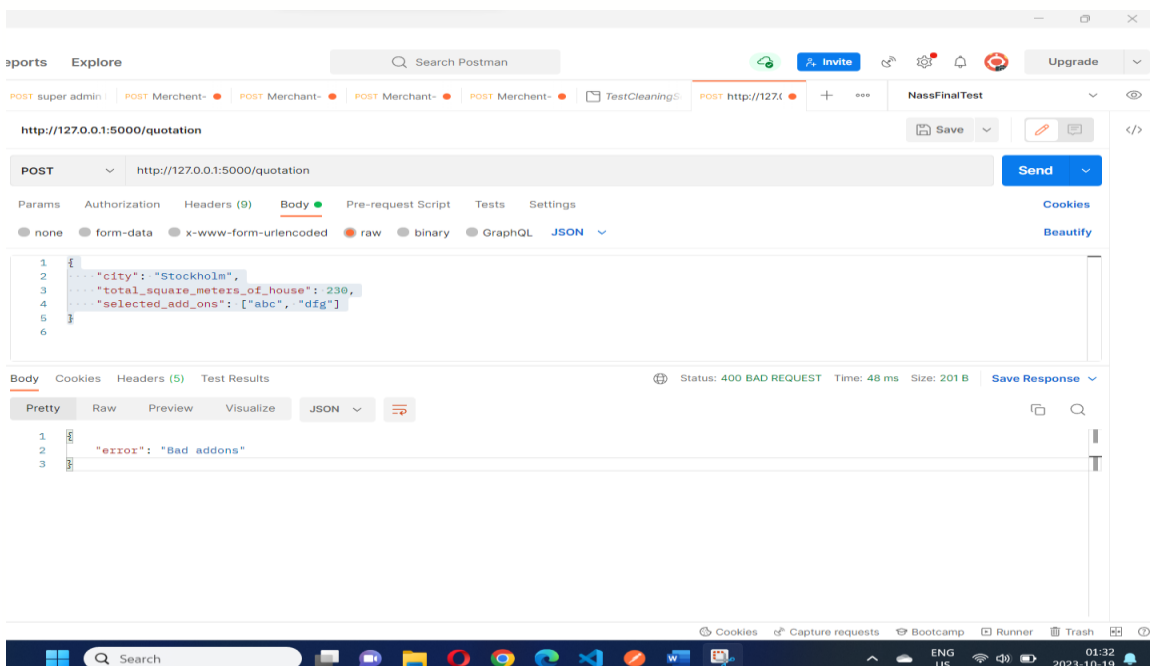
Output:-



Payload-3 (Json)

```
{
  "city": "Stockholm",
  "total_square_meters_of_house": 230,
  "selected_add_ons": ["abc", "dfg"]
}
```

Output:-



9)

http://127.0.0.1:5000/q, 500 Internal Server Error

Output:-

http://127.0.0.1:5000/q

The screenshot displays a REST client interface for a POST request to `http://127.0.0.1:5000/q`. The request body is a JSON object:

```
1 {
2   "city": "Stockholm",
3   "total_square_meters_of_house": 230,
4   "selected_add_ons": ["abc", "dfg"]
5 }
6
```

The response is shown in the 'Body' tab, indicating a 500 Internal Server Error with the message 'Not found':

```
1 {
2   "error": "Not found"
3 }
```