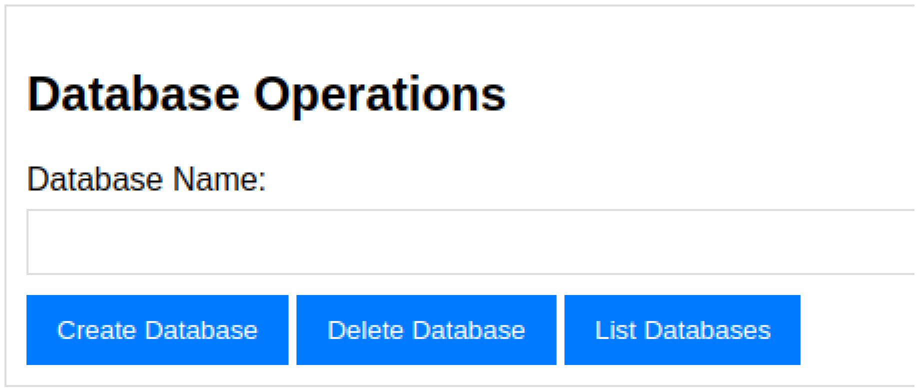
Working of UI

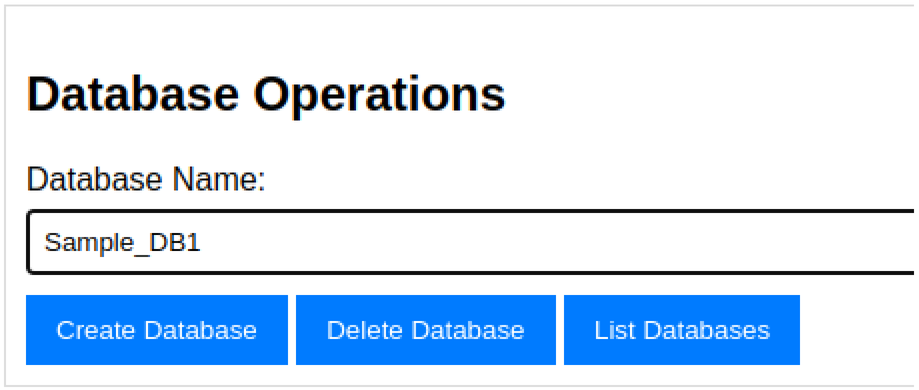
Group-14
Sawale Sumeet Shivaji - 22110234
Yash Patkar - 22110296
Neerja Kasture - 22110165
Anura Mantri - 22110144

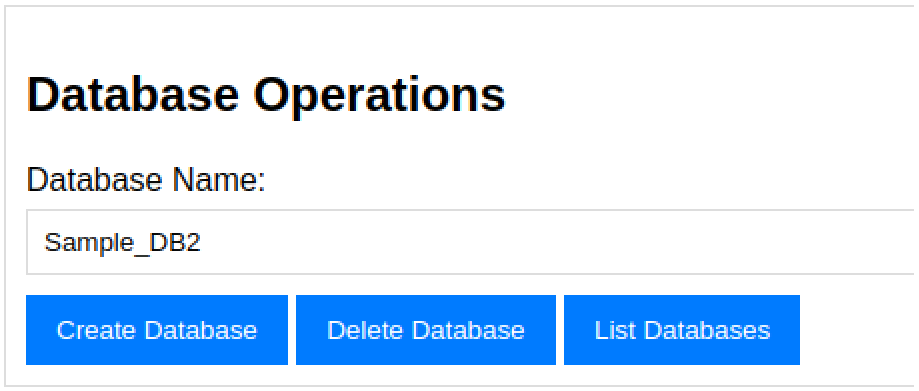
1. Database operations



- This is how database operations look.
- We have an input field for the name of the database and we can create or delete database with the given name or get a list of all databases.

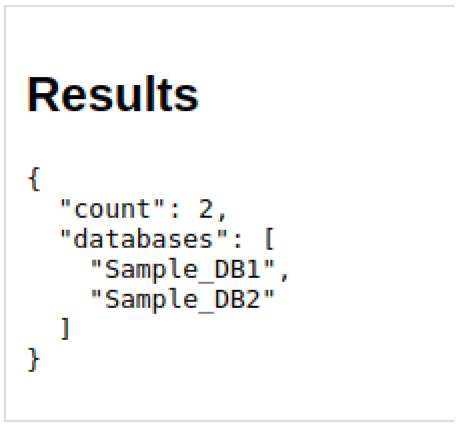
   a. Creating a database.





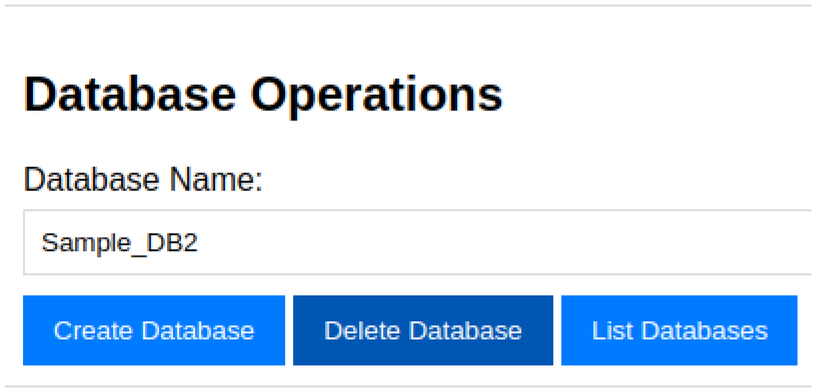
- We created 2 sample databases.

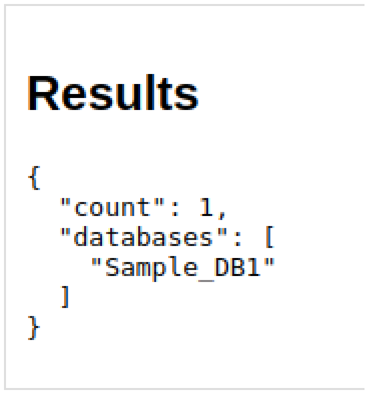
b. Viewing list of all databases.



```
{
  "count": 2,
  "databases": [
    "Sample_DB1",
    "Sample_DB2"
  ]
}
```

c. Deleting a database.





Results

```
{
  "count": 1,
  "databases": [
    "Sample_DB1"
  ]
}
```

- After deleting Sample_DB2, we are only left with Sample_DB1

2. Table operations:

- Here we can create or delete tables from a given database, or get a list of all tables in the database.

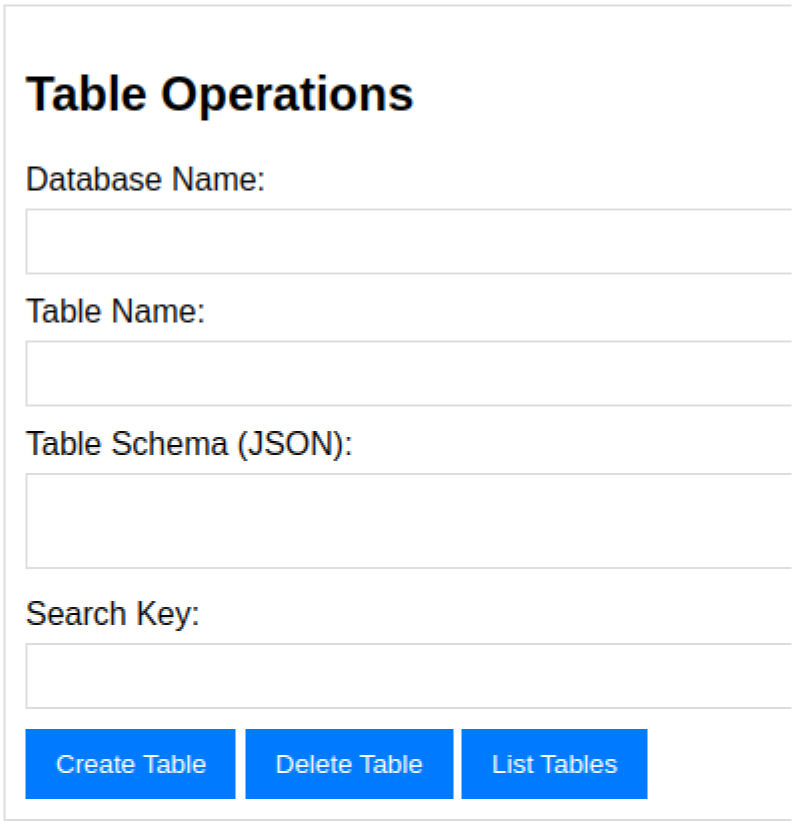


Table Operations

Database Name:

Table Name:

Table Schema (JSON):

Search Key:

a. Creating a table.

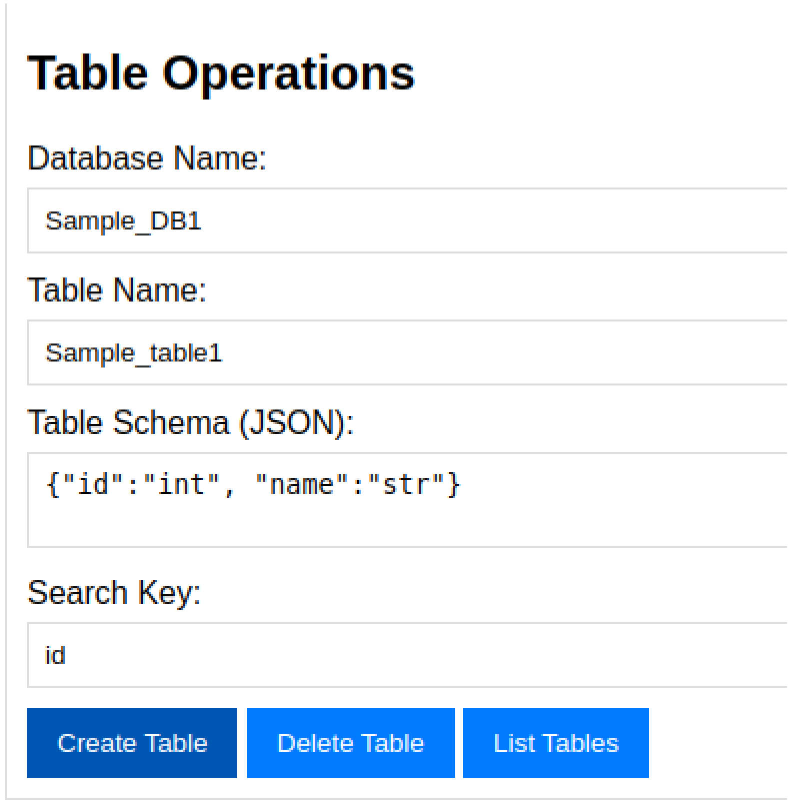


Table Operations

Database Name:

Sample_DB1

Table Name:

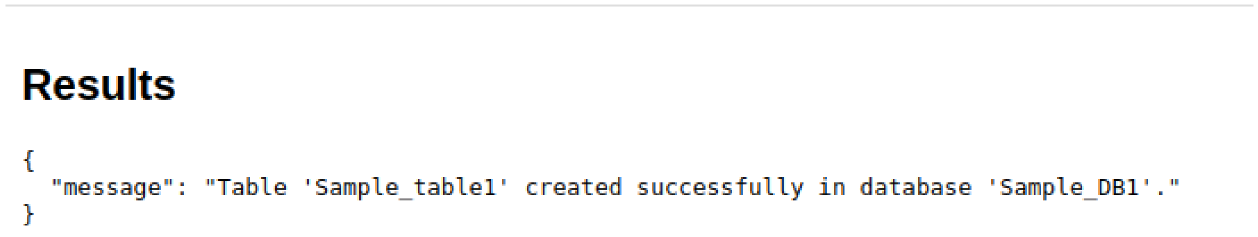
Sample_table1

Table Schema (JSON):

```
{"id":"int", "name":"str"}
```

Search Key:

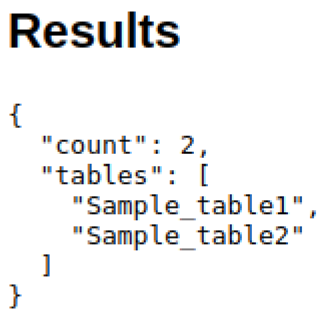
id



Results

```
{
  "message": "Table 'Sample_table1' created successfully in database 'Sample_DB1'."
}
```

b. Listing tables in a database.



Results

```
{
  "count": 2,
  "tables": [
    "Sample_table1",
    "Sample_table2"
  ]
}
```

c. Deleting a table

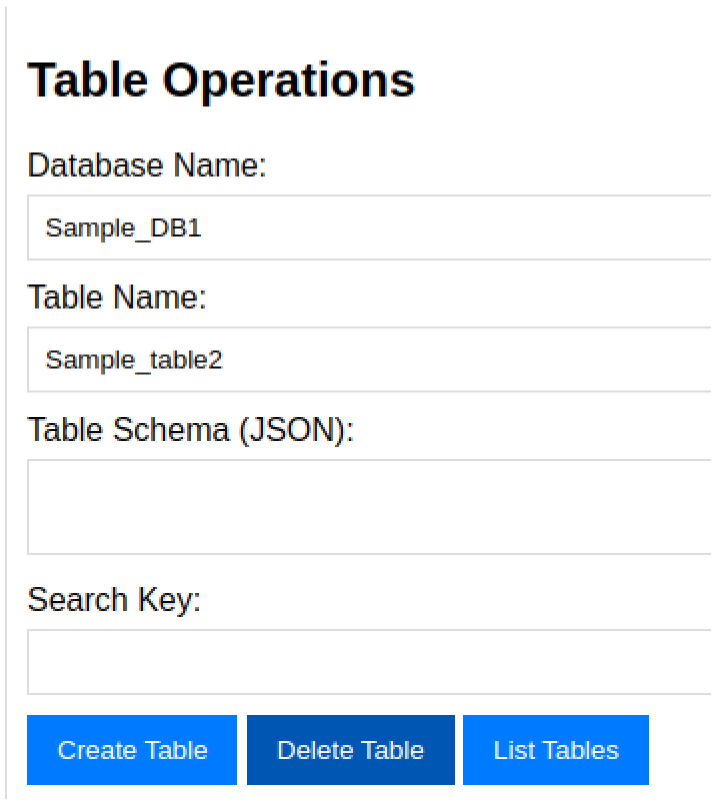


Table Operations

Database Name:

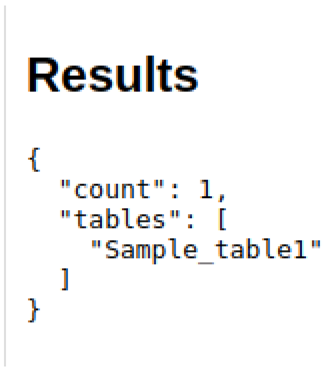
Sample_DB1

Table Name:

Sample_table2

Table Schema (JSON):

Search Key:



Results

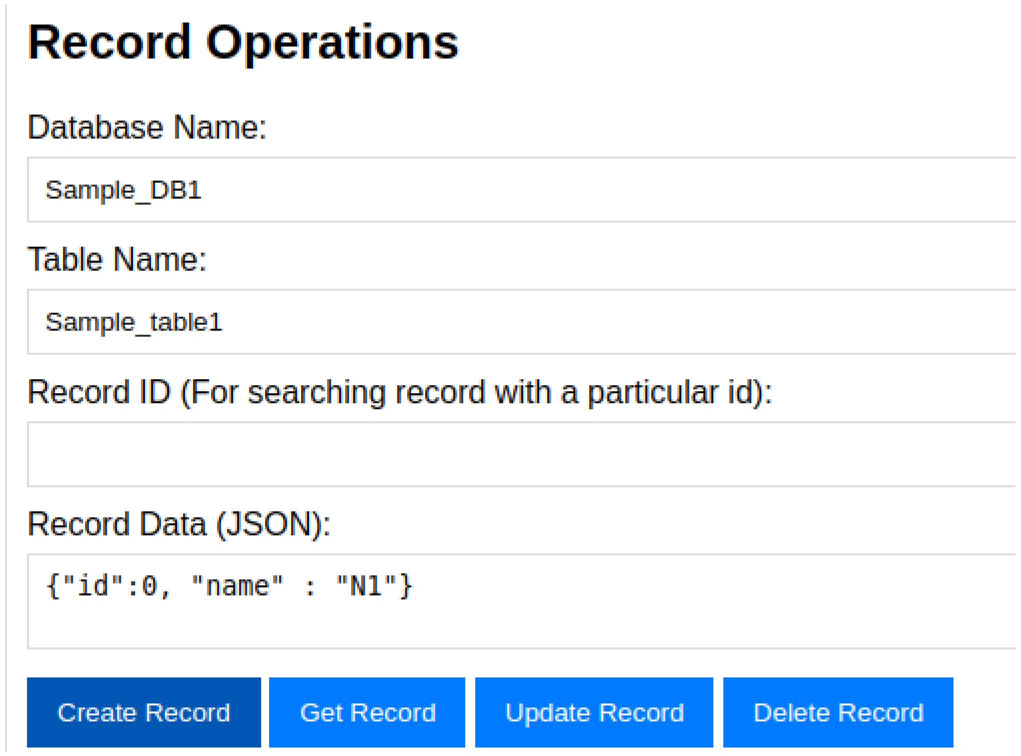
```
{
  "count": 1,
  "tables": [
    "Sample_table1"
  ]
}
```

- After deleting Sample_table2, we are left with only Sample_table1

3. Record operations.

a. Adding a record to a table.

- We will create some dummy entries.



Record Operations

Database Name:

Sample_DB1

Table Name:

Sample_table1

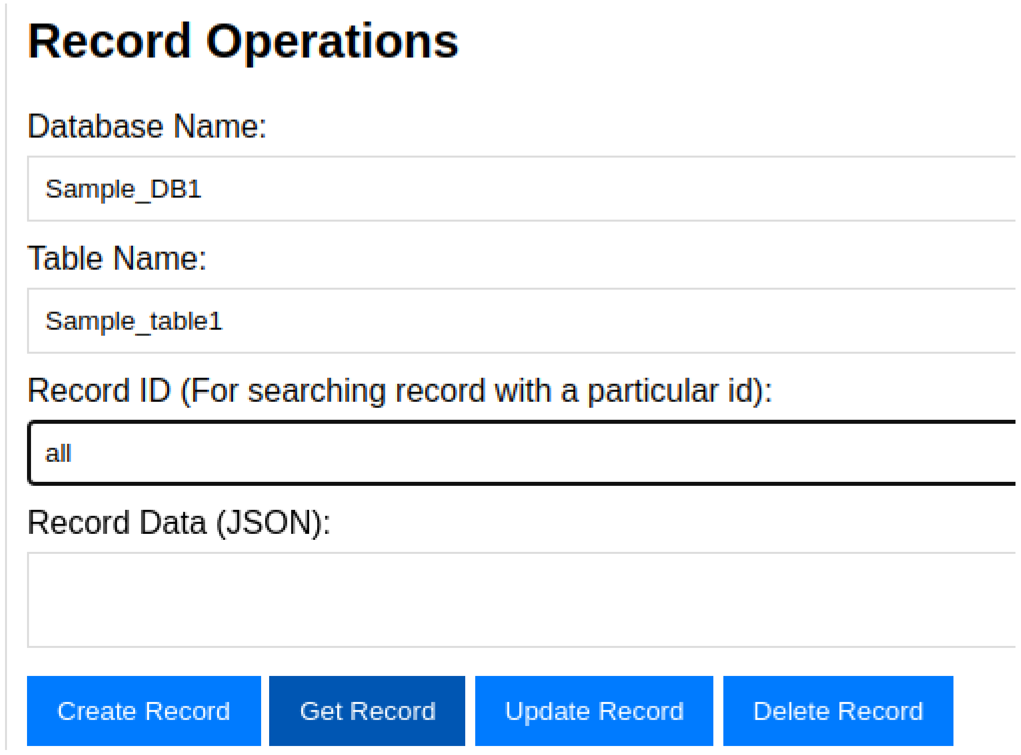
Record ID (For searching record with a particular id):

Record Data (JSON):

```
{"id":0, "name" : "N1"}
```

b. Listing all the records in a table.

- We use "all" in the Record ID field to get all the records in the table.



Record Operations

Database Name:

Sample_DB1

Table Name:

Sample_table1

Record ID (For searching record with a particular id):

all

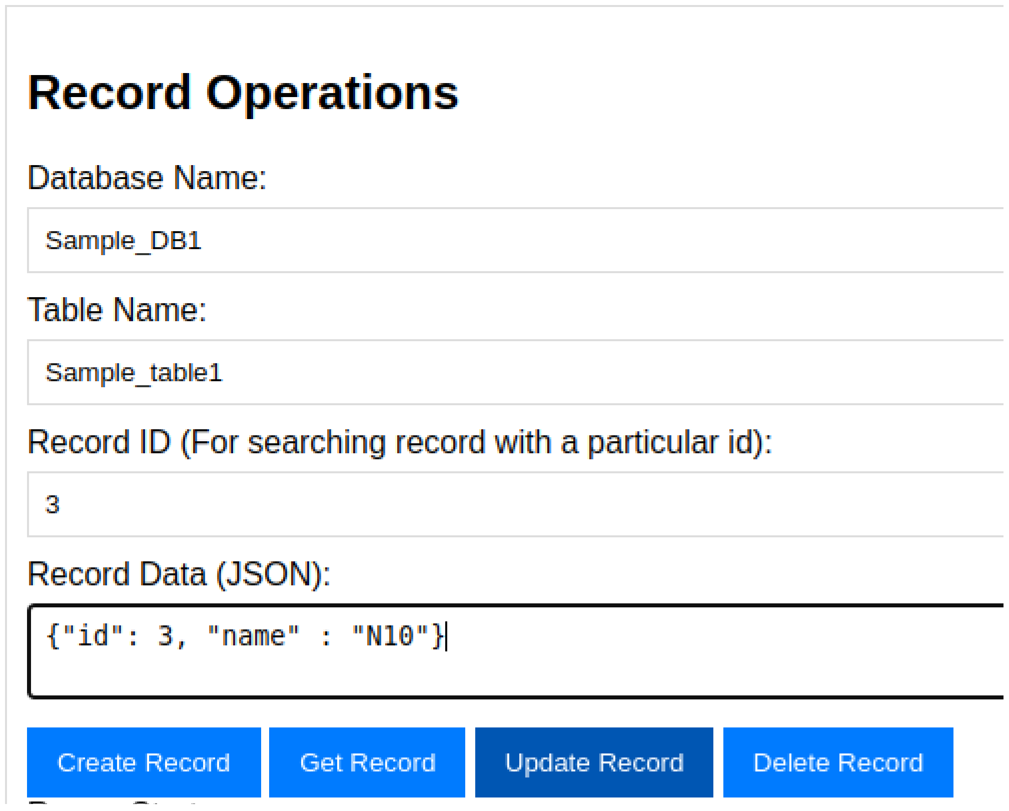
Record Data (JSON):

Results

```
{
  "count": 5,
  "records": [
    {
      "data": {
        "id": 0,
        "name": "N0"
      },
      "id": 0
    },
    {
      "data": {
        "id": 1,
        "name": "N1"
      },
      "id": 1
    },
    {
      "data": {
        "id": 2,
        "name": "N2"
      },
      "id": 2
    },
    {
      "data": {
        "id": 3,
        "name": "N3"
      },
      "id": 3
    },
    {
      "data": {
        "id": 4,
        "name": "N4"
      },
      "id": 4
    }
  ]
}
```

c. Updating a record.

- We can update a record by providing its “Record ID” and the new data.
- Here we are changing the name of the record with id 3 from “N3” to “N10”



Record Operations

Database Name:

Sample_DB1

Table Name:

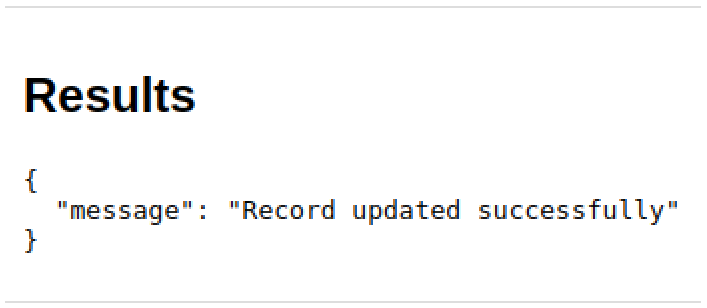
Sample_table1

Record ID (For searching record with a particular id):

3

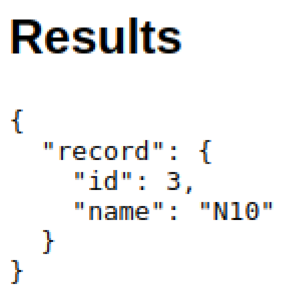
Record Data (JSON):

```
{"id": 3, "name" : "N10"}
```



Results

```
{
  "message": "Record updated successfully"
}
```

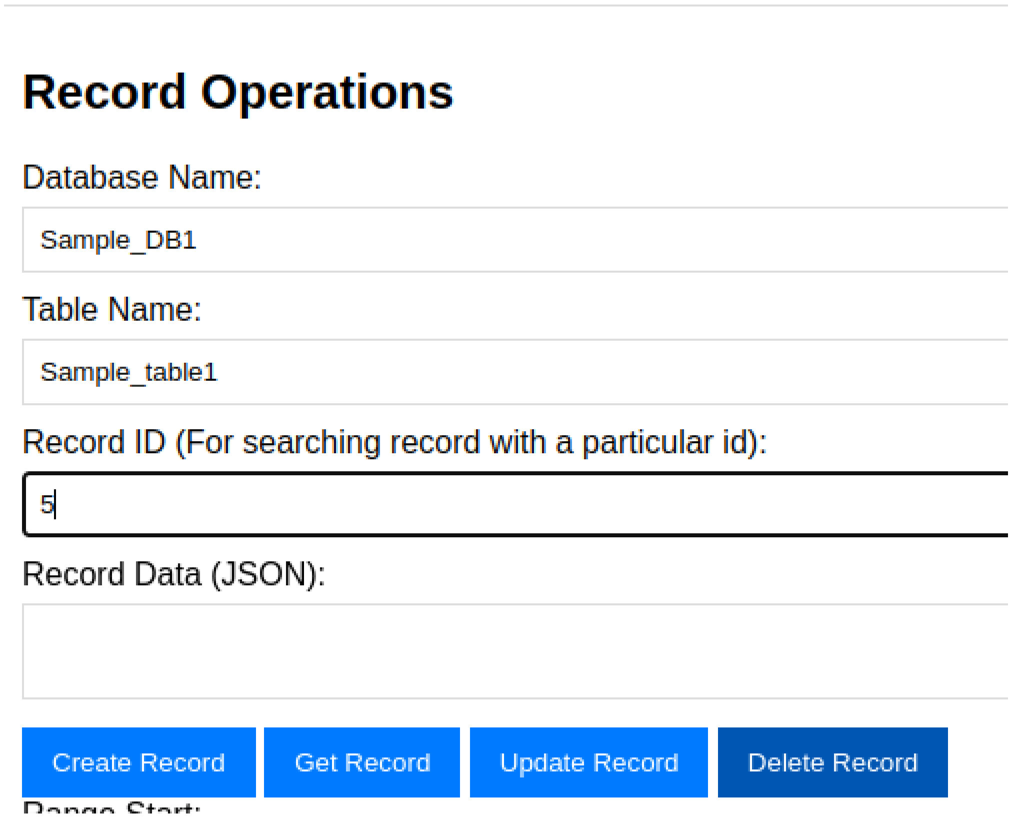


Results

```
{
  "record": {
    "id": 3,
    "name": "N10"
  }
}
```

d. Deleting a record.

- Here we are deleting the record with id 5.



Record Operations

Database Name:

Sample_DB1

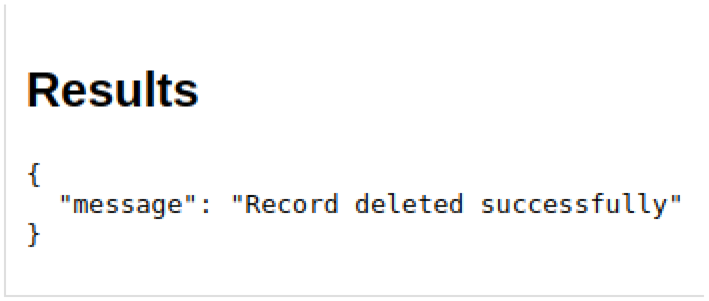
Table Name:

Sample_table1

Record ID (For searching record with a particular id):

5

Record Data (JSON):



Results

```
{
  "message": "Record deleted successfully"
}
```

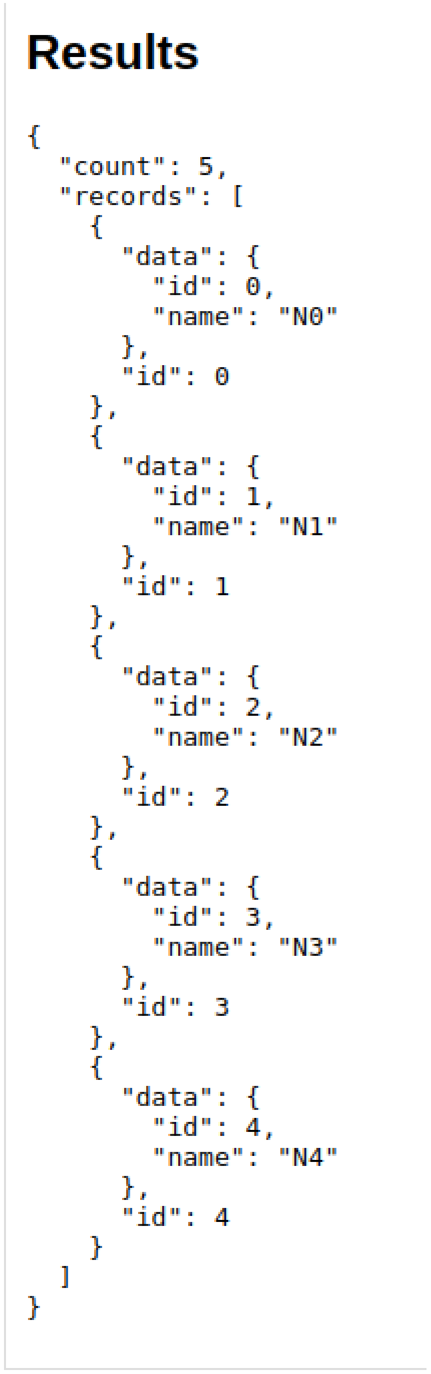
Results

```
{
  "count": 5,
  "records": [
    {
      "data": {
        "id": 0,
        "name": "N0"
      },
      "id": 0
    },
    {
      "data": {
        "id": 1,
        "name": "N1"
      },
      "id": 1
    },
    {
      "data": {
        "id": 2,
        "name": "N2"
      },
      "id": 2
    },
    {
      "data": {
        "id": 3,
        "name": "N3"
      },
      "id": 3
    },
    {
      "data": {
        "id": 4,
        "name": "N4"
      },
      "id": 4
    }
  ]
}
```

4. Search queries:

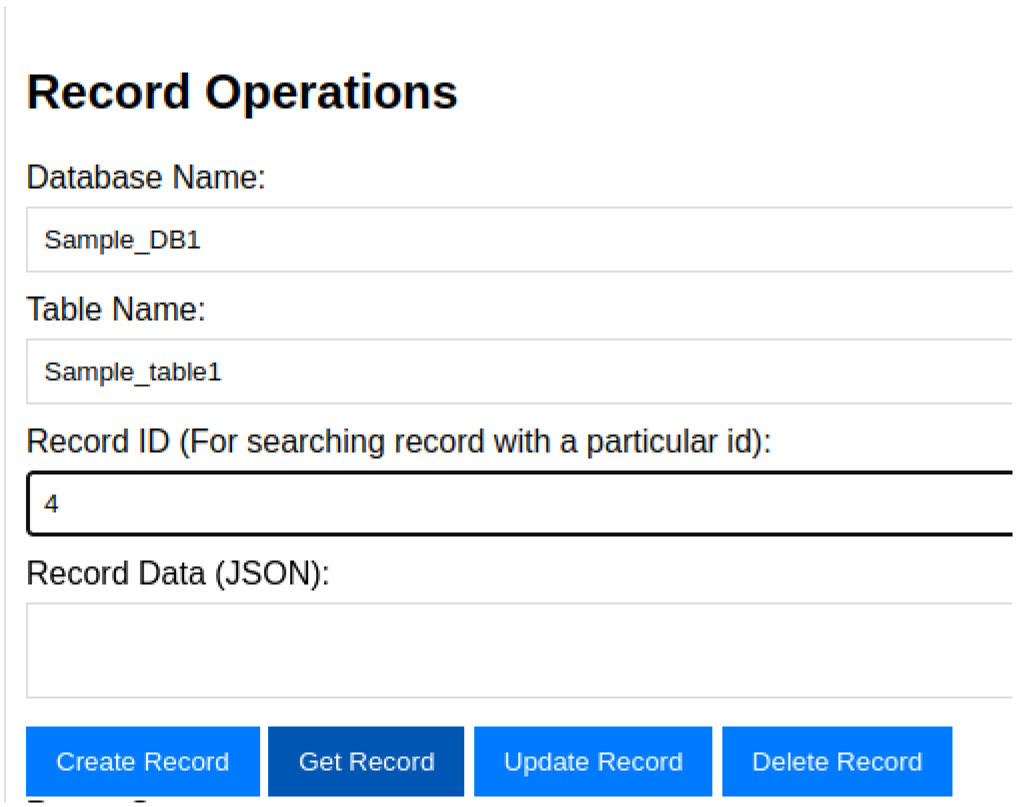
a. Exact value query.

- We have the following 5 records in our table.
- We will search for record with ID 4.



Results

```
{
  "count": 5,
  "records": [
    {
      "data": {
        "id": 0,
        "name": "N0"
      },
      "id": 0
    },
    {
      "data": {
        "id": 1,
        "name": "N1"
      },
      "id": 1
    },
    {
      "data": {
        "id": 2,
        "name": "N2"
      },
      "id": 2
    },
    {
      "data": {
        "id": 3,
        "name": "N3"
      },
      "id": 3
    },
    {
      "data": {
        "id": 4,
        "name": "N4"
      },
      "id": 4
    }
  ]
}
```



Record Operations

Database Name:

Sample_DB1

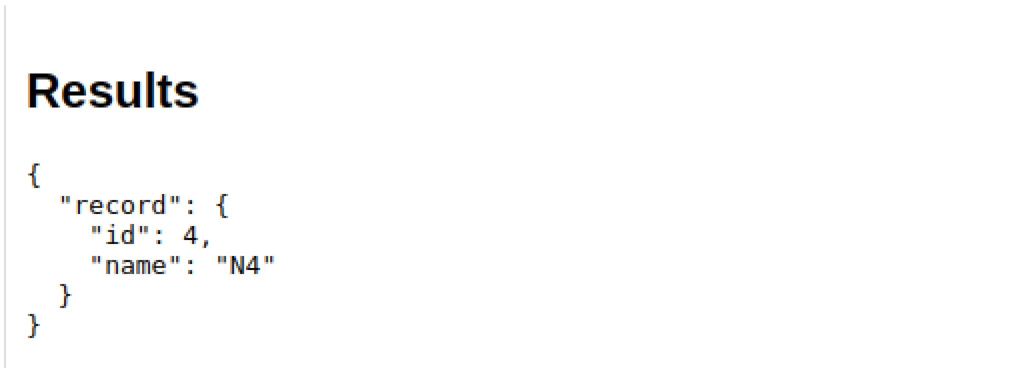
Table Name:

Sample_table1

Record ID (For searching record with a particular id):

4

Record Data (JSON):

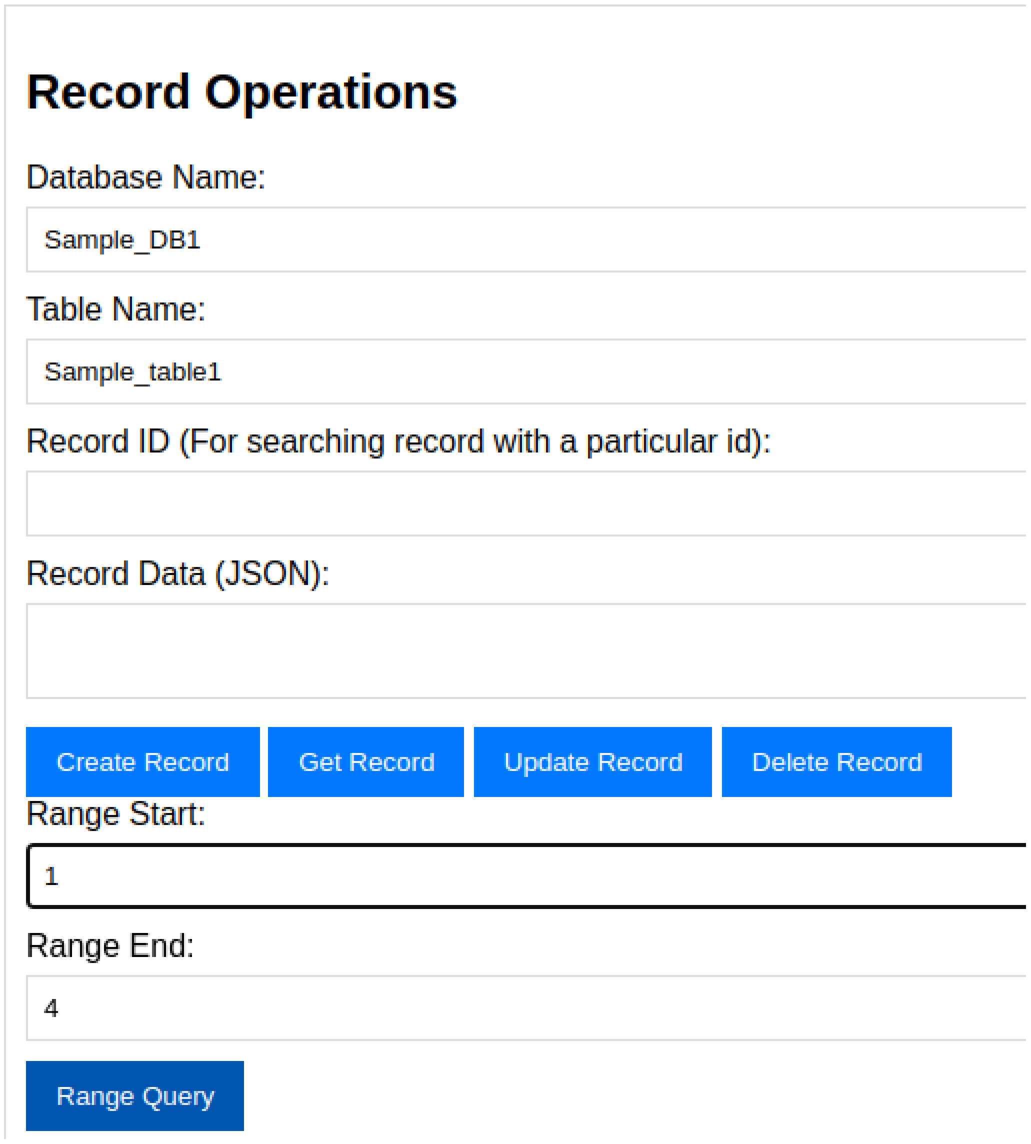


Results

```
{
  "record": {
    "id": 4,
    "name": "N4"
  }
}
```

b. Range query

- Searching for all records with 1 <= id <= 4.
- There is “id” in the output at two places.
- Inside data, it is the id attribute of the table.
- Outside the “data” field, it is the index of the returned data.
- We got 4 results with id = 1, 2, 3, and 4



Record Operations

Database Name:

Sample_DB1

Table Name:

Sample_table1

Record ID (For searching record with a particular id):

Record Data (JSON):

Range Start:

1

Range End:

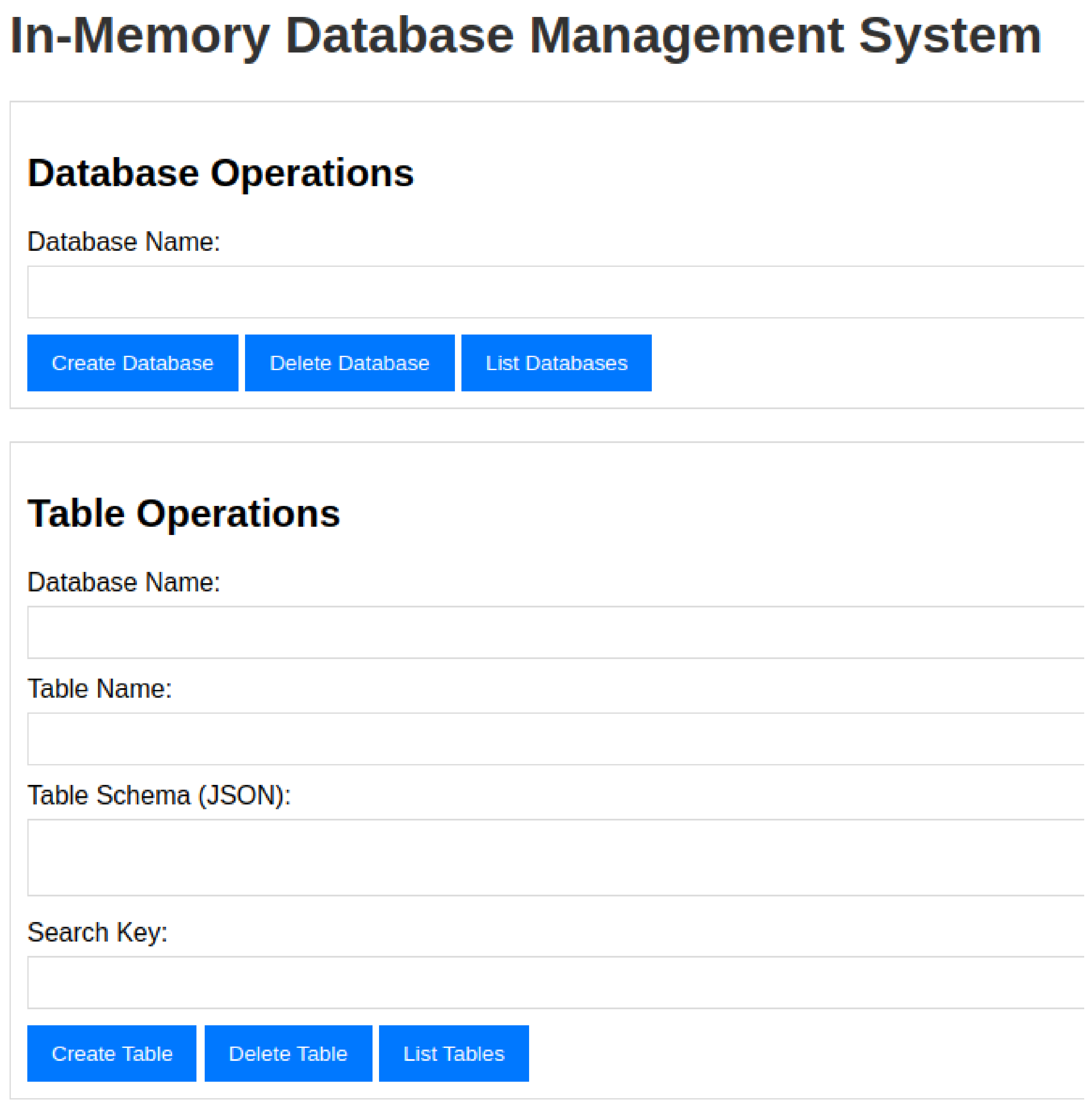
4

Results

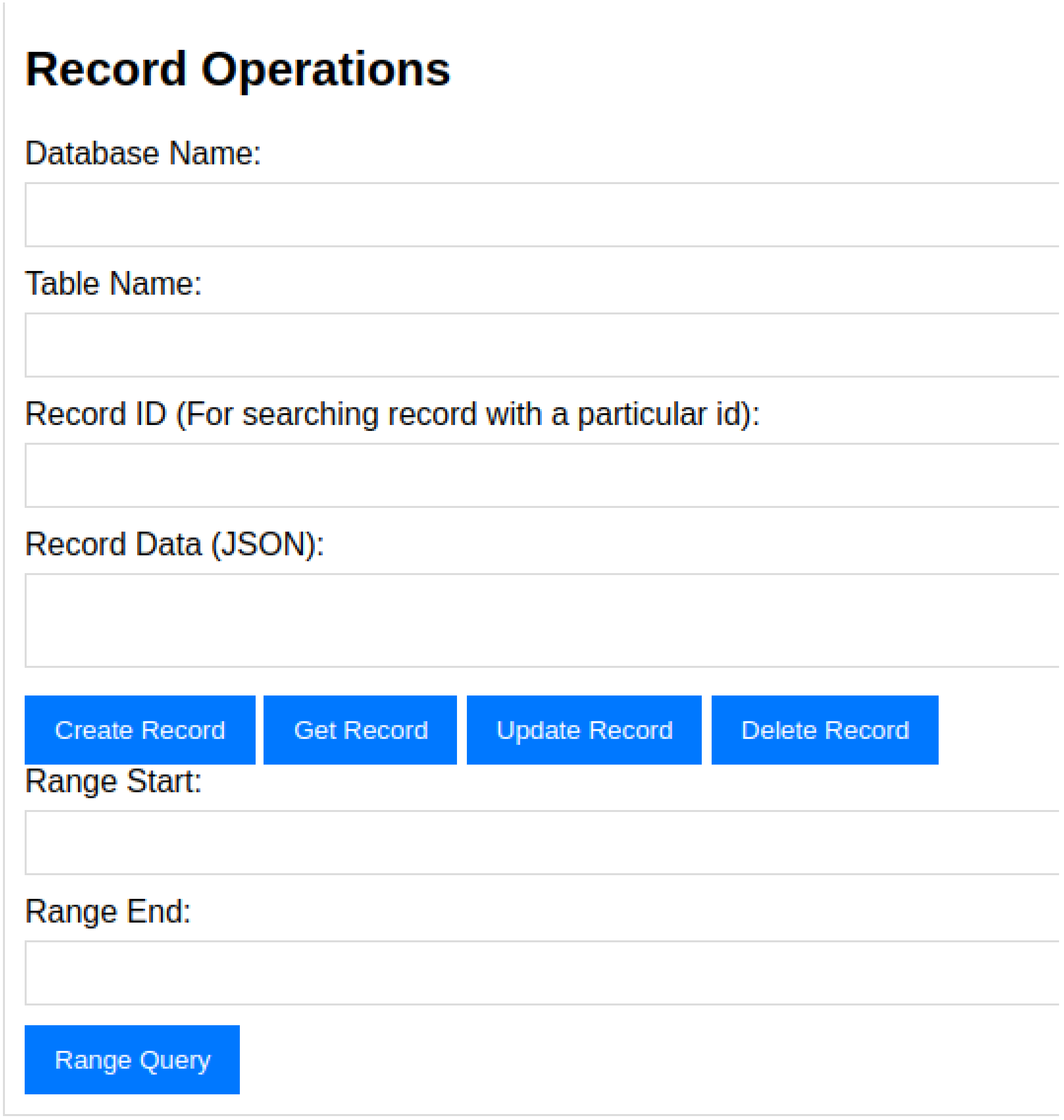
```
{
  "count": 4,
  "results": [
    {
      "data": {
        "id": 1,
        "name": "N1"
      },
      "id": 0
    },
    {
      "data": {
        "id": 2,
        "name": "N2"
      },
      "id": 1
    },
    {
      "data": {
        "id": 3,
        "name": "N3"
      },
      "id": 2
    },
    {
      "data": {
        "id": 4,
        "name": "N4"
      },
      "id": 3
    }
  ]
}
```

Overall format of the UI.

1. Database operations followed by table operations.



2. Record operations.



Record Operations

Database Name:

Table Name:

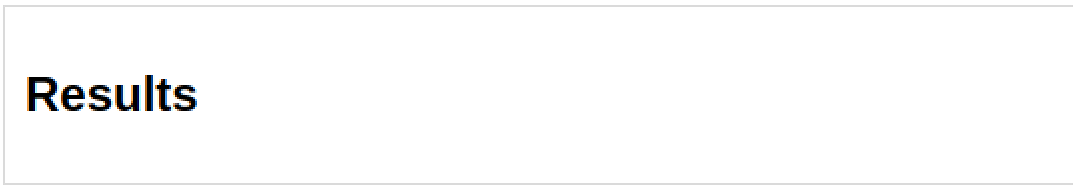
Record ID (For searching record with a particular id):

Record Data (JSON):

Range Start:

Range End:

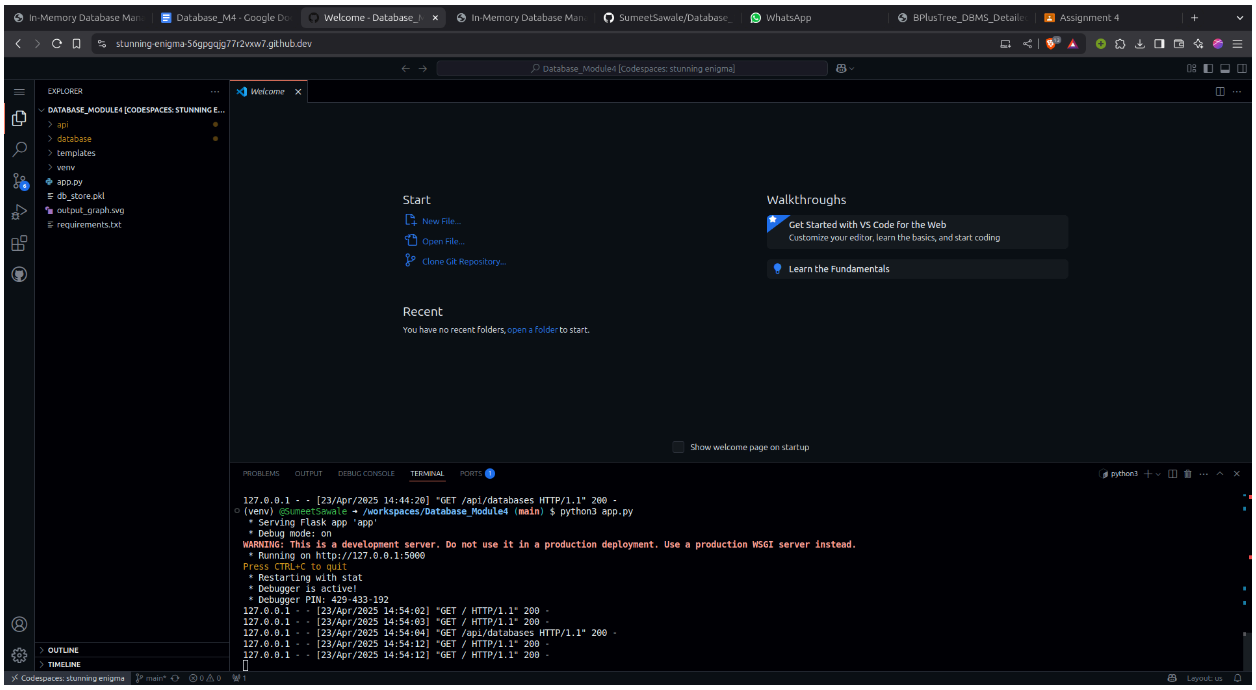
3. Results at the end of the page.



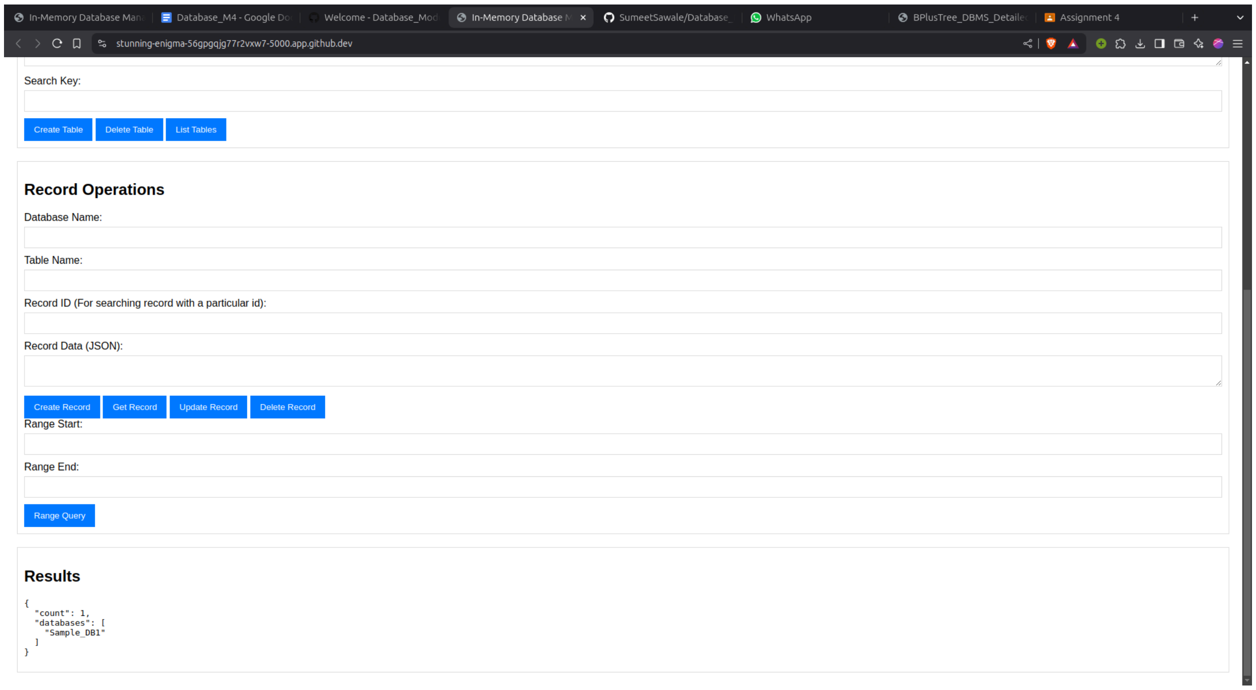
Results

I tried running it on github codespaces and it does work. First activate the virtual environment and then run “python3 app.py”.

If it does not run try “pip install flask” and “pip install graphviz”


stunning-enigma-56gpgqjg77r2vxw7.github.dev
EXPLORER
DATABASE_MODULE4 [CODESPACES: STUNNING E...
api
database
templates
venv
app.py
db_store.pkl
output_graph.svg
requirements.txt
Welcome
Start
New File...
Open File...
Clone Git Repository...
Recent
You have no recent folders, open a folder to start.
Walkthroughs
Get Started with VS Code for the Web
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Show welcome page on startup
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```
127.0.0.1 - - [23/Apr/2025 14:44:20] "GET /api/databases HTTP/1.1" 200 -
(venv) @SumeetSawale → /workspaces/Database_Module4 (main) $ python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 429-433-192
127.0.0.1 - - [23/Apr/2025 14:54:02] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [23/Apr/2025 14:54:03] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [23/Apr/2025 14:54:04] "GET /api/databases HTTP/1.1" 200 -
127.0.0.1 - - [23/Apr/2025 14:54:12] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [23/Apr/2025 14:54:12] "GET / HTTP/1.1" 200 -
```



Search Key:

Record Operations

Database Name:

Table Name:

Record ID (For searching record with a particular id):

Record Data (JSON):

Range Start:

Range End:

Results

```
{
  "count": 1,
  "databases": [
    "Sample_DB1"
  ]
}
```