# Restaurant Scheduler on Cloud

Group: 404NotFound

# Project Idea

"Where Should We Eat?" is a timeless, exhausting, and frustrating struggle, especially when you’re going out with a group of people.

We want to build an application to help users select restaurants for group dining

# Microservices

- User Microservice
- Restaurant Microservice
- Schedule Microservice

# Databases

- User Database
- Restaurant Database
- Schedule Database

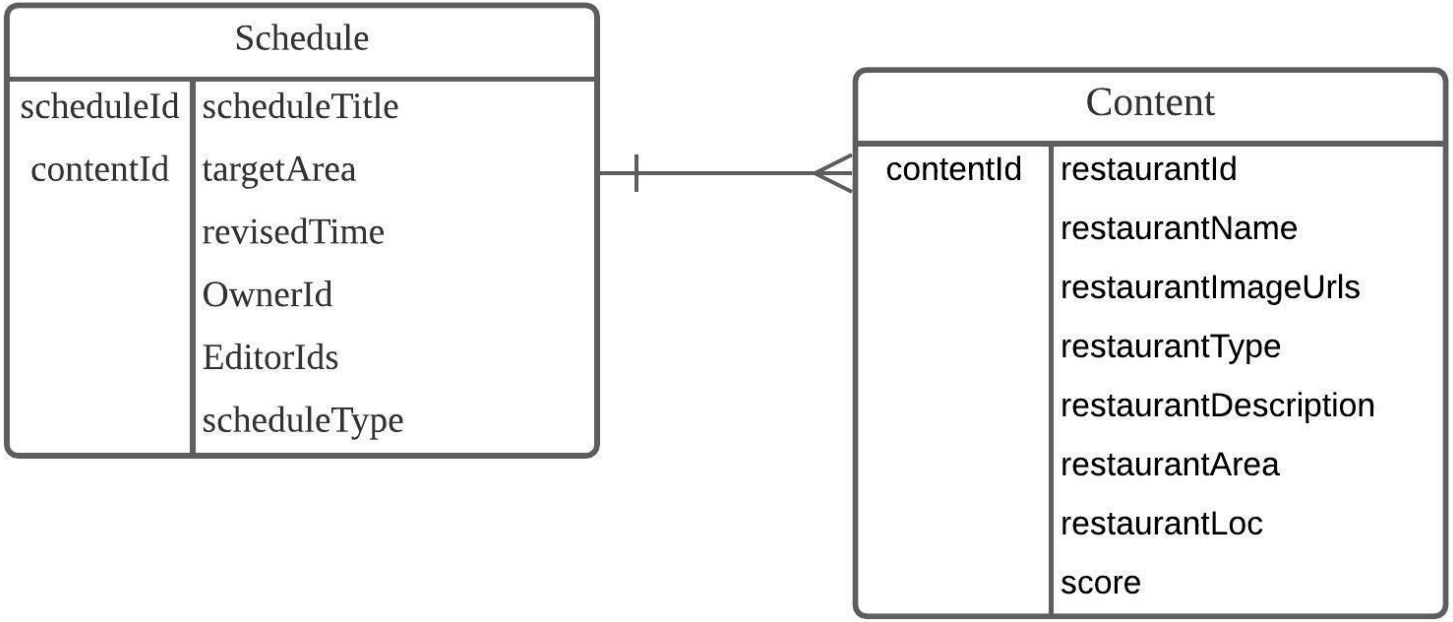
| userID | userName | userEmail | userPWD | address | Preference | Links |
|---|---|---|---|---|---|---|
| 1 | Wenqing Zhong | wz2557@columbia.edu | 12345 | NY10027 | {"1": "Chinese", "2": "Japanese", "3": "Korean"} | {"rel": "self", "href": "/users/1"} |
| 2 | Xiyuan Zhao | xz2994@columbia.edu | 123 | NY10025 | {"1": "Chinese", "2": "Korean", "3": "Thai"} | {"rel": "self", "href": "/users/2"} |
| 3 | Moxin Xu | mx2237@hotmail.com | 123 | NY10025 | {"1": "Chinese", "2": "Korean", "3": "Japanese"} | {"rel": "self", "href": "/users/3"} |
| 4 | Zhejian Jin | zj2324@columbia.edu | 123 | NY10025 | {"1": "American", "2": "Indian", "3": "Italian", "4": "Congo"} | {"rel": "self", "href": "/users/4"} |
| 6 | Aandy | 11@qq.com | 11 | WA1111 | {} | {} |

| rid | name | owner | type | location | links |
|---|---|---|---|---|---|
| 1 | New_name_for_restaurant1 | Manson Wudy | Pizza | 10029 | {"rel": "self", "href": "/restaurants/1"} |
| 2 | Massawa | Wooden Wuddy | Seafood | 10027 | {"rel": "self", "href": "/restaurants/2"} |
| 3 | 12 | Stephen Wang | Chinese | 10025 | {"rel": "self", "href": "/restaurants/3"} |

| contentId | restaurantId | restaurantName | restaurantImageUrls | restaurantType | restaurantDescription | restaurantArea | restaurantLoc | score |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Thai Noodle | www.baidu.com | Thai | This restaurant sells Thai noodle | NY10027 | 50,127 | 70 |
| 2 | 2 | Shanghai Cai | www.google.com | Shanghai | This is a tasty Shanghai Food | NY10001 | 30,50 | 80 |

| scheduledId | contentId | scheduledTitle | revisedTime | targetArea | ownerId | editorIds |
|---|---|---|---|---|---|---|
| 1 | 1 | schedule1 | 2021-12-22 20:19:59 | NY10027 | 1 | 2 |
| 1 | 2 | schedule1 | 2021-12-22 20:24:27 | NY10027 | 1 | 2 |
| 2 | 3 | schedule2 | 2021-12-22 20:41:27 | NY10014 | 3 | 1 |

# Schedule Database



# User Microservice

Recommend Just For You!

User | Create

Continue

Url path(inserting/updating/deleting tuples):

-GET(/users/?name=M, /users/?name=Moxin&limit=1&offset=1)

-POST, DELETE(/delete_user) {‘uid’:10}

Interface:

First page(sign in or sign up):

**Login from here:**

User Name:

Password:

Submit

Please enter the user name:

Please enter the user id:

Please enter the user email:

Please enter the user password:

Please enter the user address:

Please enter the user preferences:

Create

**Create schedule:**

Schedule Title:

Target Area:

Restaurant Name:

Restaurant Type:

Submit

Second page(chosen user/schedule and create):

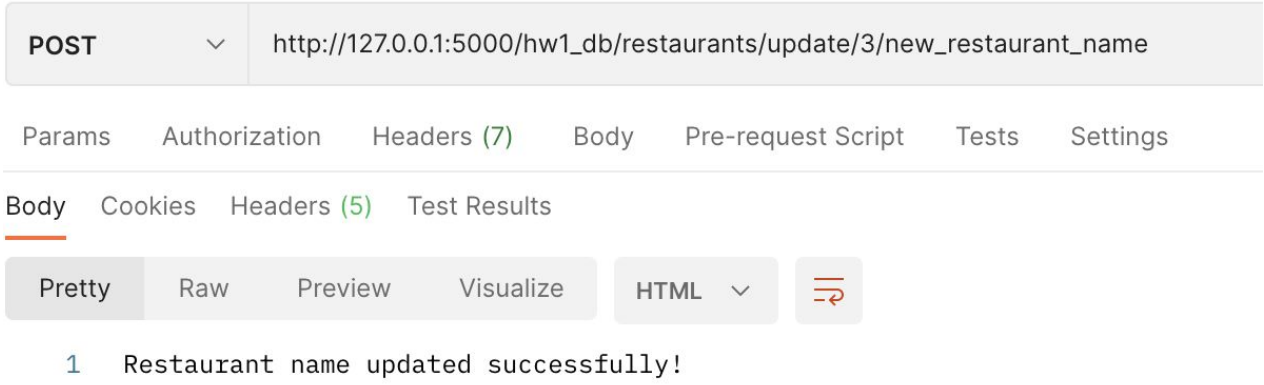
Link: http://18.222.171.216:5000/

# Restaurant Microservice

Listing all the restaurants to choose

And inserting/searching/updating/deleting tuples of the restaurants:

E.g. Update the restaurants name based on restaurant id.



# Schedule Microservice

Endpoints:

- **GET /schedule**: retrieve the schedule list by user id
- **POST /schedule**: create a preselected type schedule
- **GET /schedule/{scheduleId}**: retrieve a schedule
- **POST /schedule/{scheduleId}**: update a schedule
- **DELETE /schedule/{scheduleId}**: delete a schedule
- **GET schedule/{scheduleId}/restaurant/{restaurantId}**: retrieve a like info of a restaurant in the schedule
- **PUT schedule/{scheduleId}/restaurant/{restaurantId}**:  initialize a like info of a restaurant in the schedule
- **DELETE schedule/{scheduleId}/restaurant/{restaurantId}**: delete a like info of a restaurant in the schedule
- **GET /schedule/{scheduleId}/submit**: change the stage of the schedule from  to COMPLETED
- **GET /schedule/{scheduleId}/finish**: change the stage of the schedule from EDITING to COMPLETED

# What we accomplished

- Microservices that support CRUD operations for users

GET http://18.216.70.122:5000/hw1_db/users/7/Helen Eillish/123@fire.com/12123/CA1222/{}/{}

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (5) Test Results 200 OK 143 ms 1.46 KB

Pretty Raw Preview Visualize JSON


```
        "address": "NY10025",
        "Preference": "{\"1\": \"American\", \"2\": \"Indian\", \"3\": \"Italian\", \"4\":
        "Links": "{\"rel\": \"self\", \"href\": \"/users/4\"}"
    },
    {
        "userID": "6",
        "userName": "Aandy",
        "userEmail": "11@qq.com",
        "userPWD": "11",
        "address": "WA1111",
        "Preference": "{}",
        "Links": "{}"
    },
    {
        "userID": "7",
        "userName": "Helen Eillish",
        "userEmail": "123@fire.com",
        "userPWD": "12123",
        "address": "CA1222",
        "Preference": "{}",
        "Links": "{}"
    }
]
```

# What we accomplished

- Microservices that support CRUD operations for restaurants

GET http://127.0.0.1:5000/hw1_db/restaurants/

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON


```
[
    {
        "rid": "1",
        "name": "New_name_for_restaurant1",
        "owner": "Manson Wudy",
        "type": "Pizza",
        "location": "10029",
        "links": "{\"rel\": \"self\", \"href\": \"/restaurants/1\"}"
    },
    {
        "rid": "2",
        "name": "Massawa",
        "owner": "Wooden Wuddy",
        "type": "Seafood",
        "location": "10027",
        "links": "{\"rel\": \"self\", \"href\": \"/restaurants/2\"}"
    },
    {
        "rid": "3",
        "name": "12",
        "owner": "Stephen Wang",
        "type": "Chinese",
        "location": "10025",
        "links": "{\"rel\": \"self\", \"href\": \"/restaurants/3\"}"
    },
```


GET http://127.0.0.1:5000/hw1_db/restaurants/search/rid/3

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON


```
[
    {
        "rid": "3",
        "name": "12",
        "owner": "Stephen Wang",
        "type": "Chinese",
        "location": "10025",
        "links": "{\"rel\": \"self\", \"href\": \"/restaurants/3\"}"
    }
]
```

# What we accomplished

- Microservices that support CRUD operations for schedule

18.220.208.27:5003/schedules

POST 18.220.208.27:5003/schedules

Params Authorization Headers (8) Body Pre-request Script

none form-data x-www-form-urlencoded raw binary

```
{
    "scheduledId": "4",
    "targetArea":"NY999",
    "ownerId":"3",
    "editorId":"1"
}
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1
```

18.220.208.27:5003/schedules/2

GET 18.220.208.27:5003/schedules/2

Params Authorization Headers (8) Body Pre-

Query Params

| KEY | |
|---|---|
| Key | |

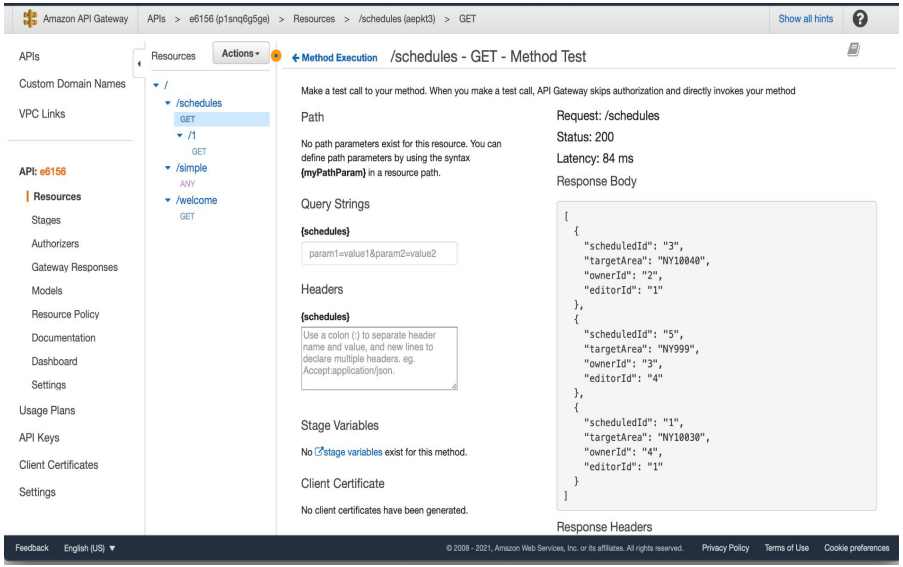
Body Cookies Headers (5) Test Results

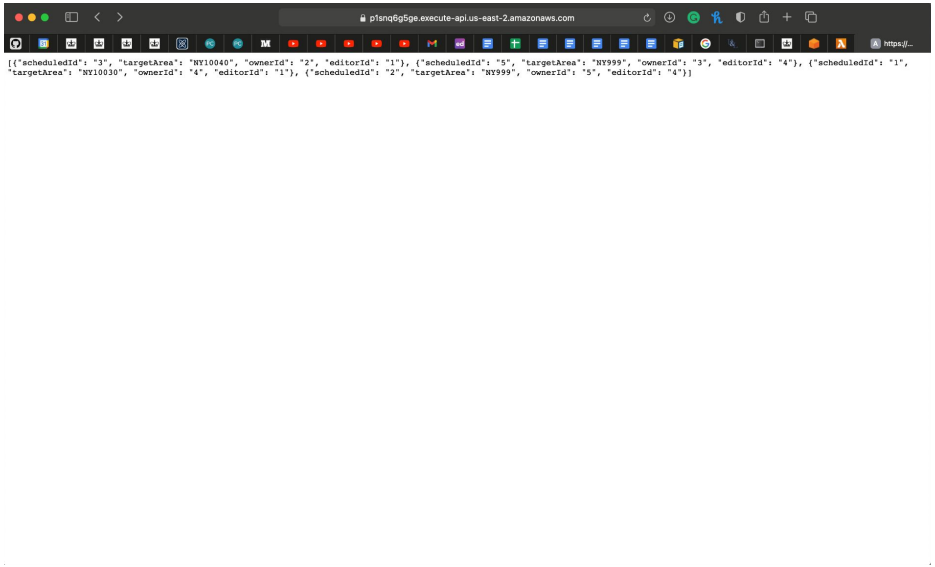
Pretty Raw Preview Visualize JSON

```
[
    {
        "scheduledId": "2",
        "targetArea": "NY001",
        "ownerId": "5",
        "editorId": "3"
    }
]
```

# What we accomplished

- API Gateway



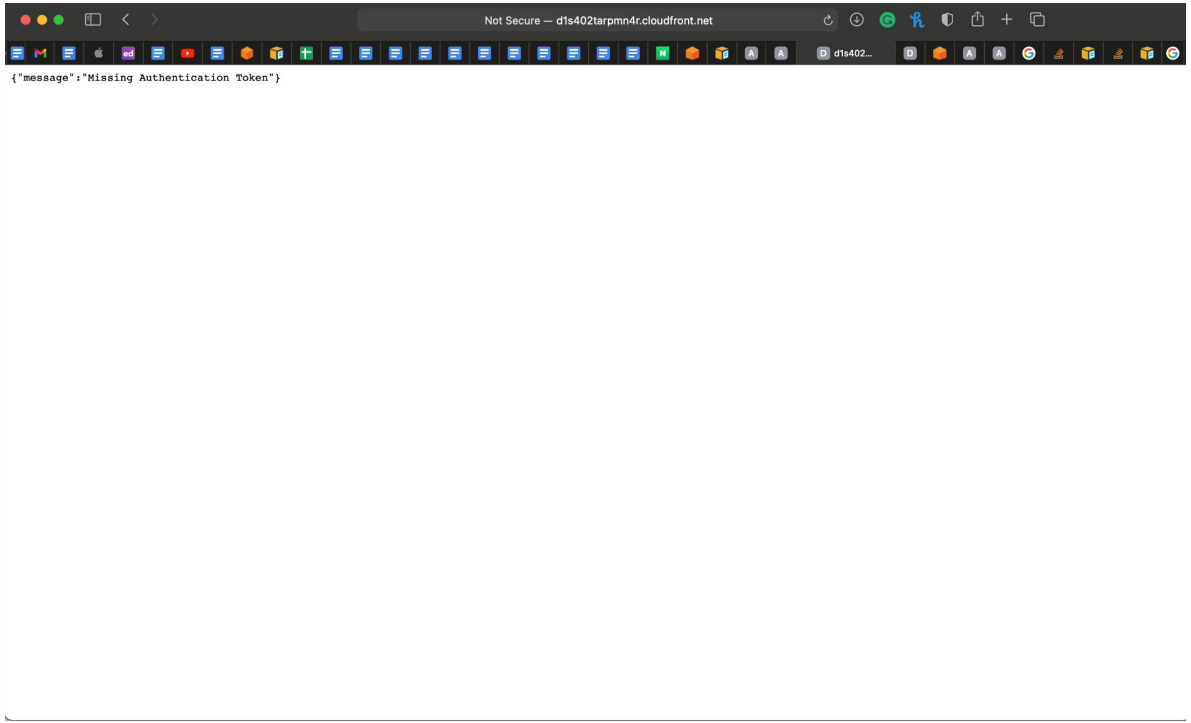
# What we accomplished

- S3 on CloudFront


| Restaurants Name | Type | Location |
|---|---|---|
| ELysian Fields Cafe | Pizza | 10029 |
| Massawa | Seafood | 10027 |
| Szechuan Garden | Chinese | 10025 |

# What we accomplished

- Problem we faced



# What we accomplished

- Pagination, Query support

GET http://160.39.152.122:5000/hw1_db/users/?name=Moxin&limit=0&offset=1

Params Authorization Headers (8) Body Pre-request Script Tests Settings

| | | |
|---|---|---|
| ☑ name | Moxin | |
| ☑ limit | 0 | |
| ☑ offset | 1 | |
| Key | Value | Description |

Body Cookies Headers (5) Test Results 200 OK 181 ms 428 B

Pretty Raw Preview Visualize JSON


```
[
    {
        "userID": "3",
        "userName": "Moxin Xu",
        "userEmail": "mx2237@hotmail.com",
        "userPWD": "123",
        "address": "NY10025",
        "Preference": "{\"1\": \"Chinese\", \"2\": \"Korean\", \"3\": \"Japanese\"}",
        "Links": "{\"rel\": \"self\", \"href\": \"/users/3\"}"
    }
]
```


```python
@app.route('/hw1_db/users/', methods=['GET'])
def get_users_by_name():
    name, limit, offset = request.args.get('name'), request.args.get('limit'), request.args.get('offset')
    if limit is None and offset is None:
        limit, offset = '0', '100'
    print(name, limit, offset)
    res = IMDBUsersResource.get_by_name(name, limit, offset)
    if res == False:
        rsp = Response('Find 0 result.', status=404, content_type="application/json")
    else:
        rsp = Response(json.dumps(res), status=200, content_type="application/json")
    return rsp
```