REST-API-Restaurant

Name: Tejas Panchal (tejaspanchal49@gmail.com)

Technology	Version
npm	5.5.5
Node.js	6.11.5
MongoDB	3.4.9
Redis	4.0.8

Steps to Run the Project

1.Install NodeJs, MongoDB, Redis.

2.Clone or download the github repository - https://github.com/panchaltej/REST-API-Restaurant.

3.Start mongodb and redis servers.

4.Install node modules

```
npm install
```

6.Start node server

```
npm start
```

7.To run unit tests

```
npm test
```

Database Schema

```
Collection: restaurants


{
    "_id"        : <restaurant_id>,
    "name"        : <restaurant_name>,
    "location"    : <restaurant_location>,
    "cuisines": [<cuisine1>, <cuisine2>],
    "ratings": <restaurant_ratings>,
    "phone_number": <restaurant_contact_number>,
    "avg_cost": <restaurant_cost>,
    "menus":[
        {
            "menu_type": <menu type>,
            "menu_items": [
                {
                    "item_name": <item name>,
                    "item_price": <item price>,
                    "desc": <item description
                }
            ]
        }
    ]
}
```

REST API Endpoints

BaseURL : localhost:3001/

Endpoint	HTTP Verb	Functionality	Example	Success Response Code
/restaurants	POST	Create new restaurant in the database.	localhost:3001/restaurants	201
/restaurants/:restaurant_id	GET	Get Restaurant details with the generated restaurant ID	localhost:3001/restaurants/:restaurant_id	200
/restaurants/:restaurantID	DELETE	Delete Restaurant with the given restaurant ID	localhost:3001/restaurants/:restaurant_id	200
/restaurants/:resta	POST	Add menu to the restaurant	localhost:3001/restaurants/:restaurant_id/menu	201

Endpoint	HTTP Verb	Functionality	Example	Success Response Code
urant_id/menu		with name.		
/restaurants/:restaurantID/menu	GET	Get all menus of a restaurant	localhost:3001/restaurants/:restaurant_id/menu	200
/restaurants/:restaurant_id/menu/:menutype	GET	Get a menu detail of a restaurant with menutype.	localhost:3001/restaurants/:restaurant_id/menu/drinks	200
/restaurants/:restaurant_id/menu/:menutype	DELETE	Delete menu of a restaurant with menutype.	localhost:3001/restaurants/:restaurant_id/menu/drinks	200
/restaurants/:restaurant_id/menu/:menutype/menuitem	POST	Add new menu item to the menu.	localhost:3001/restaurants/:restaurant_id/menu/drinks/menuitem	201
/restaurants/:restaurant_id/menu/:menutype/menuitem	GET	Get all menu items of the menu	localhost:3001/restaurants/:restaurant_id/menu/drinks/menuitem	200
/restaurants/:restaurant_id/menu/:menutype/menuitem/:menuitem	DELETE	Delete menuitem with menuid	localhost:3001/restaurants/:restaurant_id/menu/drinks/menuitem/5a797f026d887752a071a6ae	200

REST API Requests

1. Create restaurant

```
POST http://localhost:3001/restaurants
201 Created
{
    "name" : "McDonald's",
    "location": "San Jose",
    "cuisines": ["Snacks", "Burgers"],
        "ratings": "4",
        "phone_number": "+1-408-429-0627",
        "avg_cost": "20",
        "menus":[
            {
                "menu_type": "sandwiches",
                "menu_items": [
                    {
                        "item_name": "Veggie Burger",
                        "item_price": "9",
                        "desc": "Veg Sandwich"
                    },
                    {
                        "item_name": "Cheeseburger",
                        "item_price": "12",
                        "desc": "Sandwich"
                    }

                ]
            },
            {
                "menu_type": "drinks",
                "menu_items": [
                    {
                        "item_name": "Coke",
                        "item_price": "3",
                        "desc": "Soft Drink"
                    },
                    {
                        "item_name": "Sprite",
                        "item_price": "3",
                        "desc": "Soft Drink"
                    }

                ]
            }
        ]
    }
```

2.Get restaurant details

```
GET http://localhost:3001/restaurants/<restaurant-id>

GET http://localhost:3001/restaurants/5a7d201056489f10b4f5c174
200 OK
[
{
    "_id": "5a7d201056489f10b4f5c174",
    "name": "McDonald's",
    "location": "San Jose",
    "cuisines": [
        "Snacks",
        "Burgers"
    ],
    "ratings": "4",
    "phone_number": "+1-408-429-0627",
    "avg_cost": "20",
    "menus": [
        {
            "menu_type": "sandwiches",
            "menu_items": [
                {
                    "item_name": "Veggie Burger",
                    "item_price": "9",
                    "desc": "Veg Sandwich"
                },
                {
                    "item_name": "Cheeseburger",
                    "item_price": "12",
                    "desc": "Sandwich"
                }
            ]
        },
        {
            "menu_type": "drinks",
            "menu_items": [
                {
                    "item_name": "Coke",
                    "item_price": "3",
                    "desc": "Soft Drink"
                    },
                    {
                        "item_name": "Sprite",
                        "item_price": "3",
                        "desc": "Soft Drink"
                    }
                ]
            }
        ]
    }
]
```

3.Delete restaurant

```
    DELETE http://localhost:3001/restaurants/<restaurant-id>

    DELETE http://localhost:3001/restaurants/5a7d201056489f10b4f5c174
    200 No Content
```

4.Add Menu

```
    POST http://localhost:3001/<restaurant-id>/menu

    POST http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu
    201 Created
    {
        "menus":{
            "menu_type": "lunch",
            "menu_items": [
                {
                    "item_name": "demo1",
                    "item_price": "20",
                    "desc": "demo"
                },
                {
                    "item_name": "demo2",
                    "item_price": "30",
                    "desc": "demo2"
                }
        ]
    }
}
```

5.Get Menu

```
GET http://localhost:3001/restaurants/<restaurant-id>/menu/<menu_type>

GET http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu/drinks
200 OK
{
    "_id": "5a7d201056489f10b4f5c174",
    "menus": [
        {
            "menu_type": "drinks",
            "menu_items": [
                {
                    "item_name": "Coke",
                    "item_price": "3",
                    "desc": "Soft Drink"
                },
                {
                    "item_name": "Sprite",
                    "item_price": "3",
                    "desc": "Soft Drink"
                }
            ]
        }
    ]
}
```

6.Delete Menu

```
DELETE http://localhost:3001/restaurants/<restaurant-id>/menu/<menu_type>

DELETE http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu/drinks
200 No Content
```

7.Add MenuItem

```
POST http://localhost:3001/restaurants/<restaurant-id>/menu/<menu_type>/menuitem
```

```
    POST http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu/drinks/menuitem
    201 Created
    {
        "item_name": "demo4",
        "item_price": "80",
        "desc": "Test"
    }
```

8.Get MenuItem

```
    GET http://localhost:3001/restaurants/<restaurant-id>/menu/<menu_type>/menuitem


    GET http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu/drinks/menuitem
    200 OK
    {
        "menu_items": [
            {
                "item_name": "Coke",
                "item_price": "3",
                "desc": "Soft Drink"
            },
            {
                "item_name": "Sprite",
                "item_price": "3",
                "desc": "Soft Drink"
            }
        ]
    }
```

9.Delete MenuItem

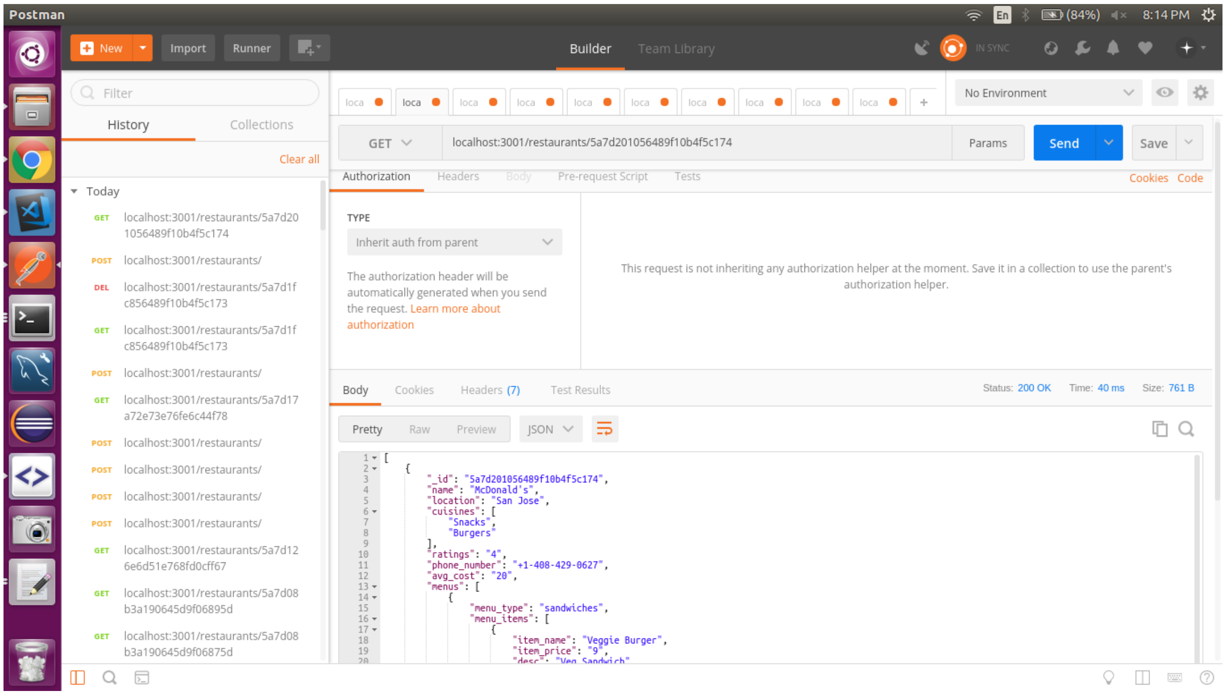
```
    DELETE http://localhost:3001/restaurants/<restaurant-
id>/menu/<menu_type>/menuitem/<item_name>

    DELETE
http://localhost:3001/restaurants/5a7d201056489f10b4f5c174/menu/drinks/menuitem/coke
    200 No Content
```

Redis for Caching

•I have attached some screenshots displaying the response time for both the scenario.

1.MongoDB Access : 40ms
2.Redis Cache : 11ms

•Fetching from mongoDB



GET localhost:3001/restaurants/5a7d201056489f10b4f5c174
POST localhost:3001/restaurants/
DEL localhost:3001/restaurants/5a7d1fc856489f10b4f5c173
GET localhost:3001/restaurants/5a7d1fc856489f10b4f5c173
POST localhost:3001/restaurants/
GET localhost:3001/restaurants/5a7d17a72e73e76fe6c44f78
POST localhost:3001/restaurants/
POST localhost:3001/restaurants/
POST localhost:3001/restaurants/
POST localhost:3001/restaurants/
GET localhost:3001/restaurants/5a7d126e6d51e768fd0cff67
GET localhost:3001/restaurants/5a7d08b3a190645d9f06895d
GET localhost:3001/restaurants/5a7d08b3a190645d9f06875d

GET localhost:3001/restaurants/5a7d201056489f10b4f5c174

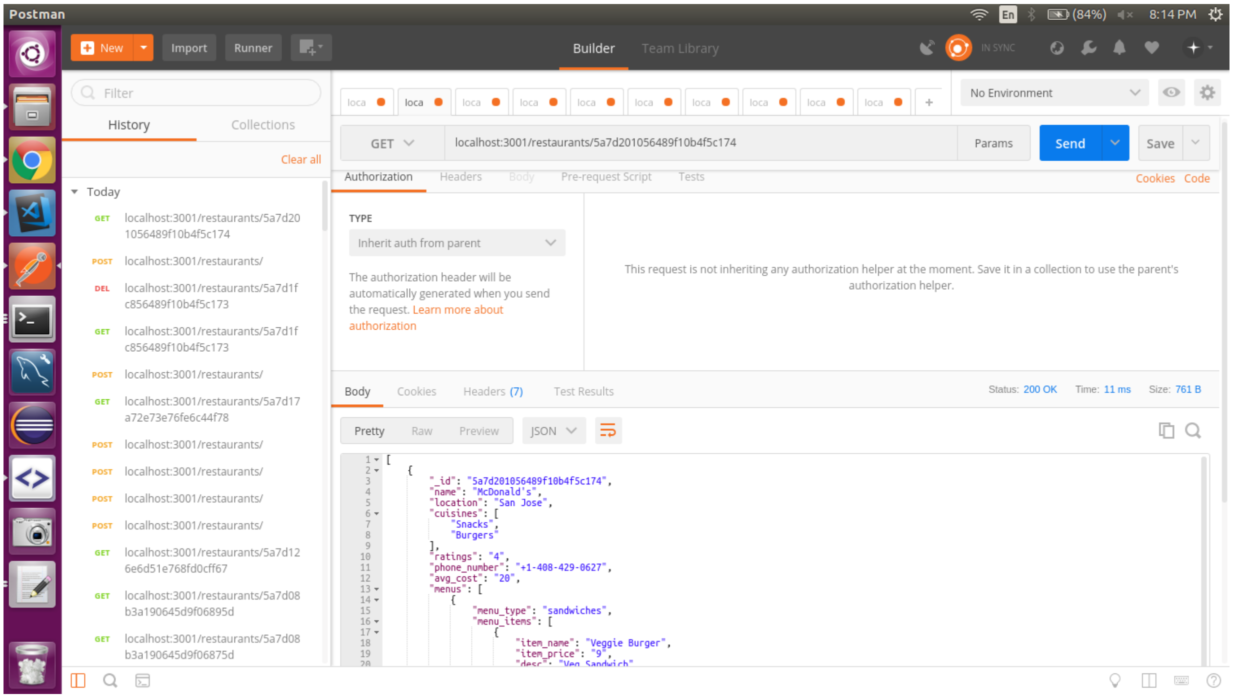
The authorization header will be automatically generated when you send the request. Learn more about authorization

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

Status: 200 OK Time: 40 ms Size: 761 B

```
[
    {
        "_id": "5a7d201056489f10b4f5c174",
        "name": "McDonald's",
        "location": "San Jose",
        "cuisines": [
            "Snacks",
            "Burgers"
        ],
        "ratings": "4",
        "phone_number": "+1-408-429-0627",
        "avg_cost": "20",
        "menus": [
            {
                "menu_type": "sandwiches",
                "menu_items": [
                    {
                        "item_name": "Veggie Burger",
                        "item_price": "9",
                        "desc": "Veg Sandwich"
```

•Fetching from Redis



GET localhost:3001/restaurants/5a7d201056489f10b4f5c174

History:

- GET localhost:3001/restaurants/5a7d201056489f10b4f5c174
- POST localhost:3001/restaurants/
- DEL localhost:3001/restaurants/5a7d1fc856489f10b4f5c173
- GET localhost:3001/restaurants/5a7d1fc856489f10b4f5c173
- POST localhost:3001/restaurants/
- GET localhost:3001/restaurants/5a7d17a72e73e76fe6c44f78
- POST localhost:3001/restaurants/
- POST localhost:3001/restaurants/
- POST localhost:3001/restaurants/
- POST localhost:3001/restaurants/
- GET localhost:3001/restaurants/5a7d126e6d51e768fd0cff67
- GET localhost:3001/restaurants/5a7d08b3a190645d9f06895d
- GET localhost:3001/restaurants/5a7d08b3a190645d9f06875d

The authorization header will be automatically generated when you send the request. Learn more about authorization

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

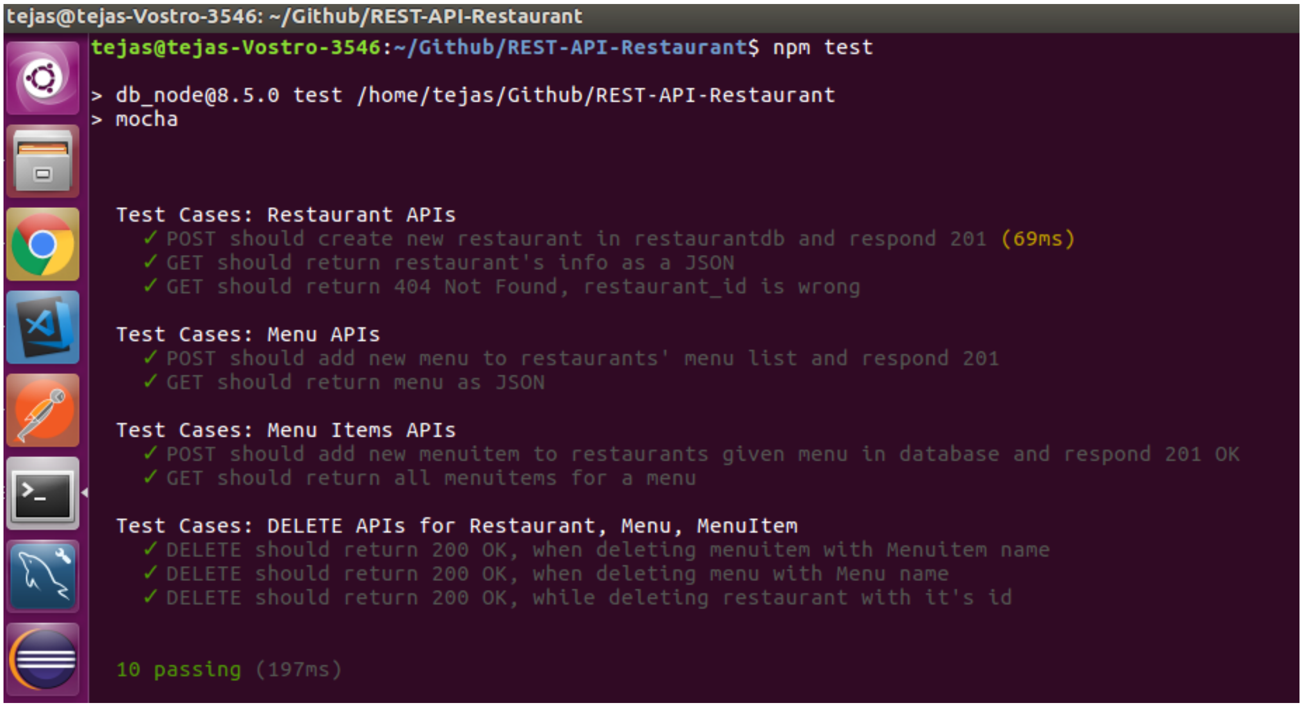
Status: 200 OK Time: 11 ms Size: 761 B

```
[
    {
        "_id": "5a7d201056489f10b4f5c174",
        "name": "McDonald's",
        "location": "San Jose",
        "cuisines": [
            "Snacks",
            "Burgers"
        ],
        "ratings": "4",
        "phone_number": "+1-408-429-0627",
        "avg_cost": "20",
        "menus": [
            {
                "menu_type": "sandwiches",
                "menu_items": [
                    {
                        "item_name": "Veggie Burger",
                        "item_price": "9",
                        "desc": "Veg Sandwich"
```

Unit Tests

Run following command to run unit tests

```
npm test
```



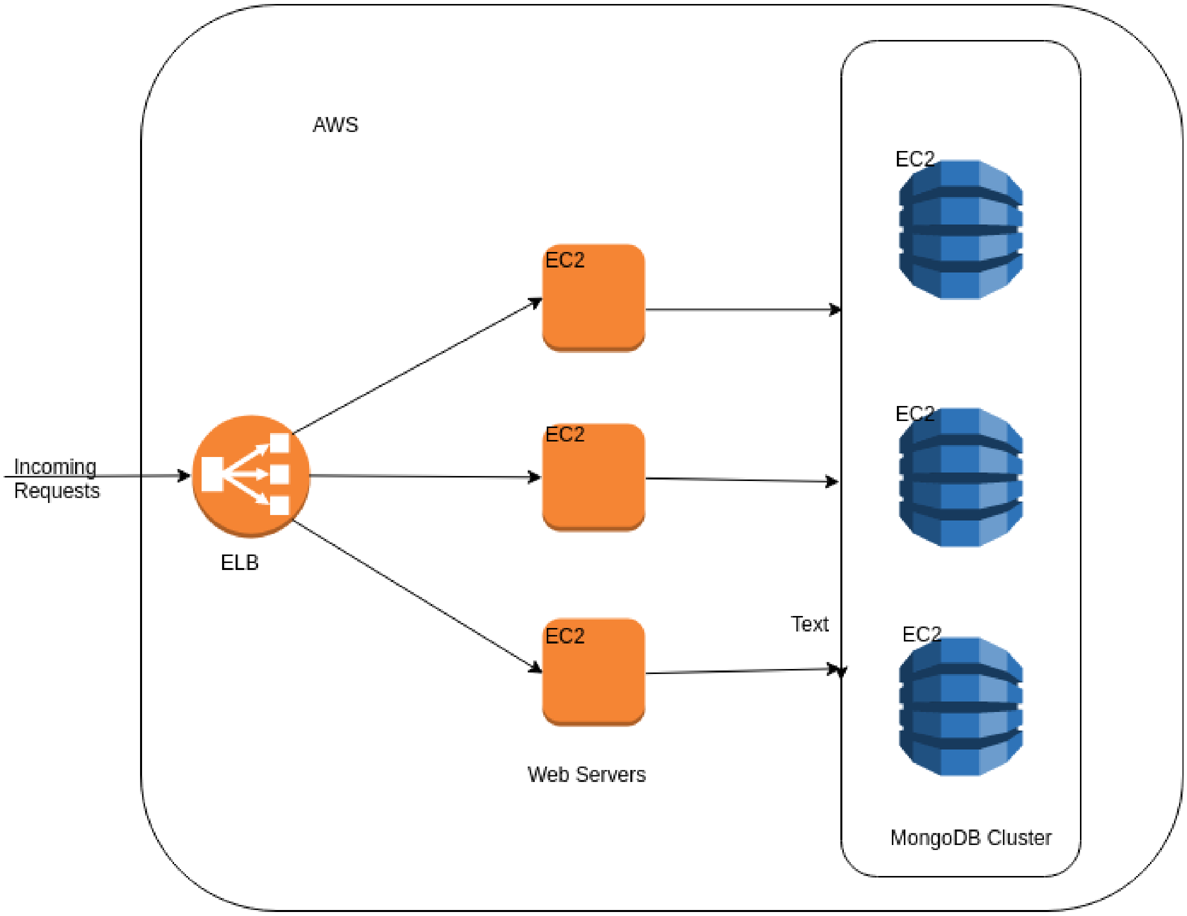
tejas@tejas-Vostro-3546: ~/Github/REST-API-Restaurant

```
tejas@tejas-Vostro-3546:~/Github/REST-API-Restaurant$ npm test

> db_node@8.5.0 test /home/tejas/Github/REST-API-Restaurant
> mocha



  Test Cases: Restaurant APIs
    ✓ POST should create new restaurant in restaurantdb and respond 201 (69ms)
    ✓ GET should return restaurant's info as a JSON
    ✓ GET should return 404 Not Found, restaurant_id is wrong

  Test Cases: Menu APIs
    ✓ POST should add new menu to restaurants' menu list and respond 201
    ✓ GET should return menu as JSON

  Test Cases: Menu Items APIs
    ✓ POST should add new menuitem to restaurants given menu in database and respond 201 OK
    ✓ GET should return all menuitems for a menu

  Test Cases: DELETE APIs for Restaurant, Menu, MenuItem
    ✓ DELETE should return 200 OK, when deleting menuitem with Menuitem name
    ✓ DELETE should return 200 OK, when deleting menu with Menu name
    ✓ DELETE should return 200 OK, while deleting restaurant with it's id


  10 passing (197ms)
```

Handling large data of restaurants all over USA

To handle millions of requests simultaneously and to scale the accordingly, we can host the server on AWS ec2 instances with load balancing. Each instances serves all the API requests and the AWS ELB balances load on all the instances.

To overcome data consistency problem, we can host mongodb database on a separate instance serving the requests being attanded to on different instances.


AWS
Incoming Requests
ELB
EC2
EC2
EC2
Web Servers
Text
EC2
EC2
EC2
MongoDB Cluster