FACULTY OF APPLIED SCIENCES AND TECHNOLOGY

NODE/EXPRESS WEB API

ITE5315 - Project

Group Member name
Umang Chudasma (N01472620)
Raveena Katariya (N01452464)

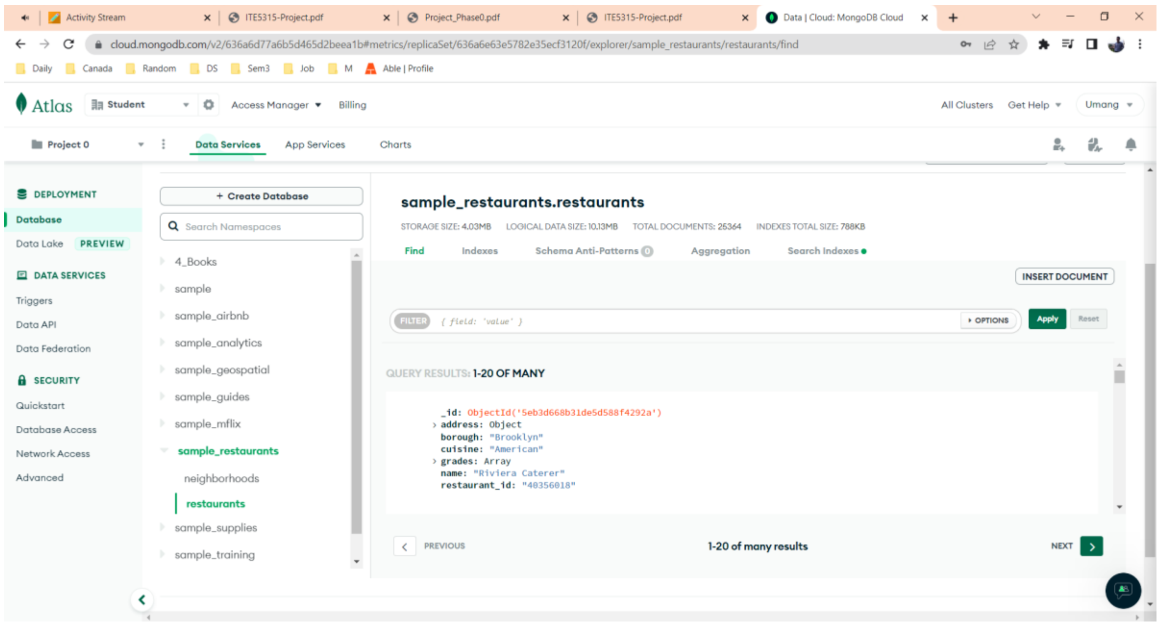
Submission Date
12/5/2022

This document explains how to build Node/Express Web API ………………………….

Table of Contents

Question 1:

(Describe the major steps for implementing the MongoDB database in Atlas)



cloud.mongodb.com/v2/636a6d77a6b5d465d2beea1b#metrics/replicaSet/636a6e63e5782e35ecf3120f/explorer/sample_restaurants/restaurants/find

sample_restaurants.restaurants

STORAGE SIZE: 4.03MB LOGICAL DATA SIZE: 10.13MB TOTAL DOCUMENTS: 25364 INDEXES TOTAL SIZE: 788KB

{ field: 'value' }

QUERY RESULTS: 1-20 OF MANY

```
  _id: ObjectId('5eb3d668b31de5d588f4292a')
> address: Object
  borough: "Brooklyn"
  cuisine: "American"
> grades: Array
  name: "Riviera Caterer"
  restaurant_id: "40356018"
```

1-20 of many results

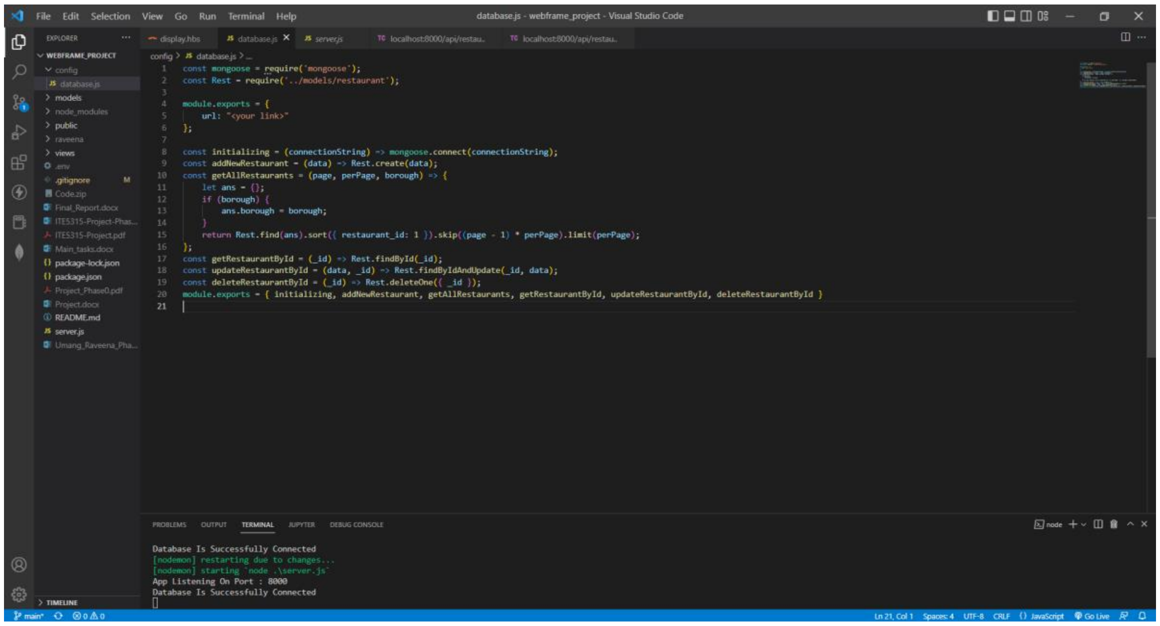
We have loaded the restaurant database in the Atlas.

Question 2:

(Describe the major steps for implementing Routes in the API, how you test this program, add some screenshots of the output)

All the routes were easy, only page, perPage and borough was a little challenging. We use an arrow function that skips the number of documents mentioned and sort the restaurant by its id. Also, we are using the inbuilt limit function that helps us display limit fields per page.

Database.js

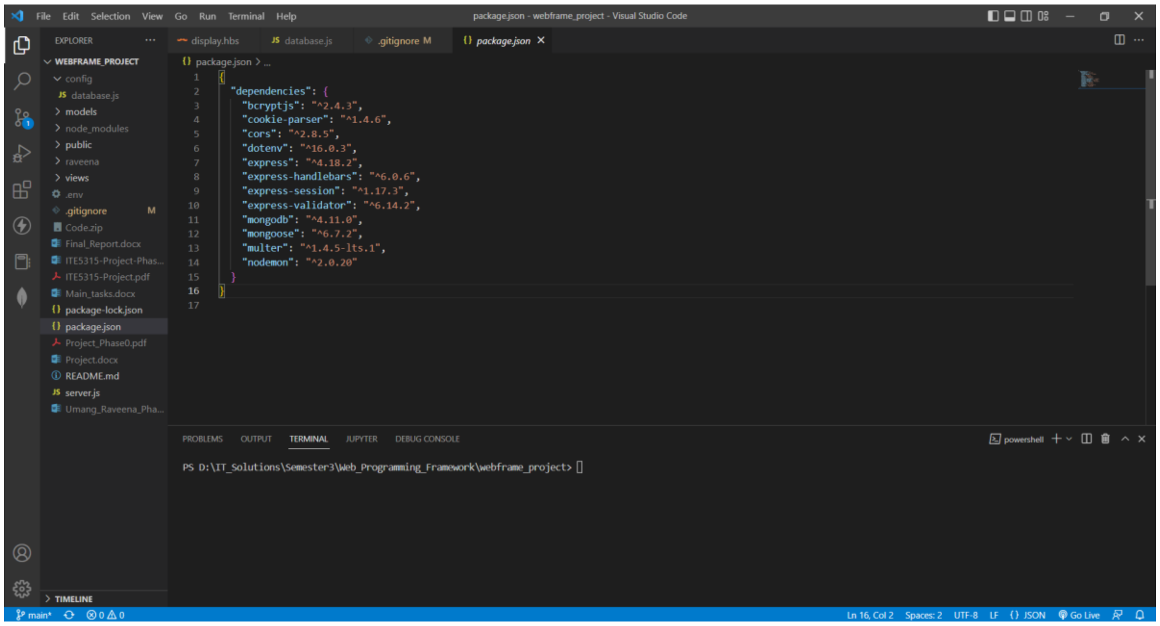


```javascript
const mongoose = require('mongoose');
const Rest = require('../models/restaurant');

module.exports = {
    url: "<your link>"
};

const initializing = (connectionString) => mongoose.connect(connectionString);
const addNewRestaurant = (data) => Rest.create(data);
const getAllRestaurants = (page, perPage, borough) => {
    let ans = {};
    if (borough) {
        ans.borough = borough;
    }
    return Rest.find(ans).sort({ restaurant_id: 1 }).skip((page - 1) * perPage).limit(perPage);
};
const getRestaurantById = (_id) => Rest.findById(_id);
const updateRestaurantById = (data, _id) => Rest.findByIdAndUpdate(_id, data);
const deleteRestaurantById = (_id) => Rest.deleteOne({ _id });
module.exports = { initializing, addNewRestaurant, getAllRestaurants, getRestaurantById, updateRestaurantById, deleteRestaurantById }

```

```
Database Is Successfully Connected
[nodemon] restarting due to changes...
[nodemon] starting `node .\server.js`
App Listening On Port : 8000
Database Is Successfully Connected
```

Server.js

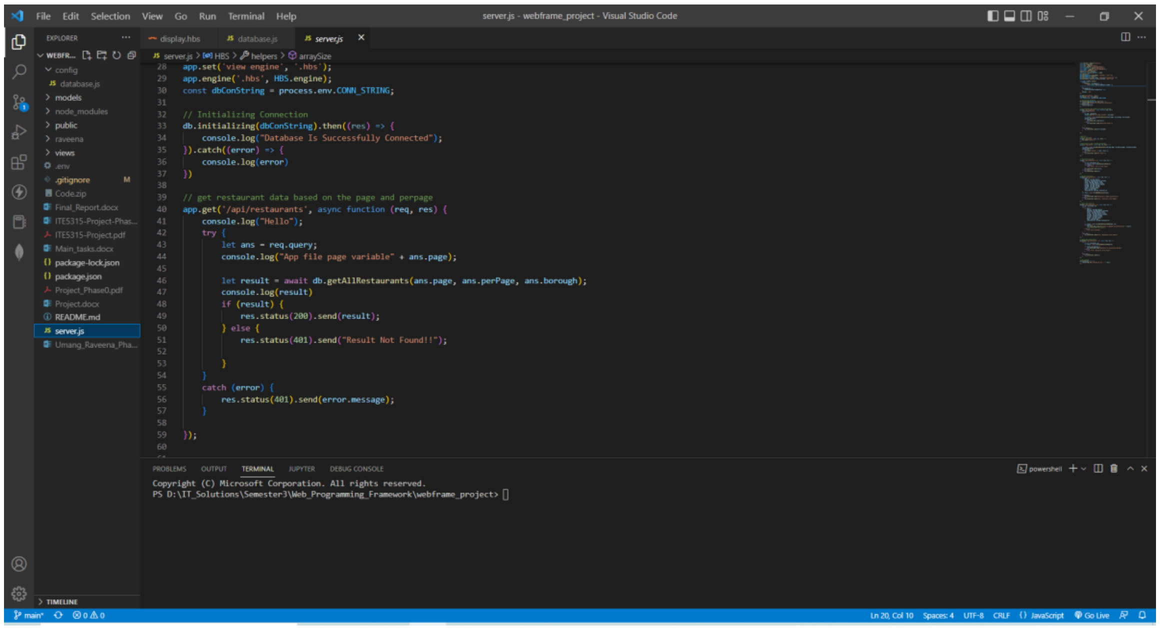
Important dependencies



```
{
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "cookie-parser": "^1.4.6",
    "cors": "^2.8.5",
    "dotenv": "^16.0.3",
    "express": "^4.18.2",
    "express-handlebars": "^6.0.6",
    "express-session": "^1.17.3",
    "express-validator": "^6.14.2",
    "mongodb": "^4.11.0",
    "mongoose": "^6.7.2",
    "multer": "^1.4.5-lts.1",
    "nodemon": "^2.0.20"
  }
}
```

```
PS D:\IT_Solutions\Semester3\Web_Programming_Framework\webframe_project> 
```

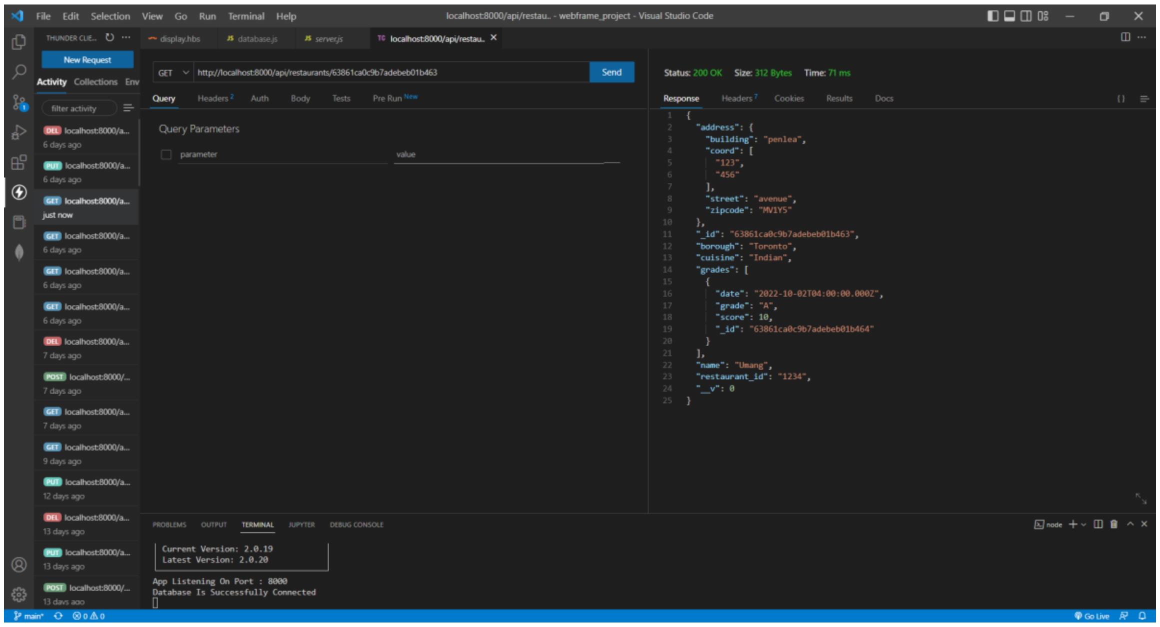
Initializing Code



```
app.set('view engine', '.hbs');
app.engine('.hbs', HBS.engine);
const dbConString = process.env.CONN_STRING;

// Initializing Connection
db.initializing(dbConString).then((res) => {
    console.log("Database Is Successfully Connected");
}).catch((error) => {
    console.log(error)
})

// get restaurant data based on the page and perpage
app.get('/api/restaurants', async function (req, res) {
    console.log("Hello");
    try {
        let ans = req.query;
        console.log("App file page variable" + ans.page);

        let result = await db.getAllRestaurants(ans.page, ans.perPage, ans.borough);
        console.log(result)
        if (result) {
            res.status(200).send(result);
        } else {
            res.status(401).send("Result Not Found!!");

        }
    }
    catch (error) {
        res.status(401).send(error.message);
    }

});
```

```
Copyright (C) Microsoft Corporation. All rights reserved.
PS D:\IT_Solutions\Semester3\Web_Programming_Framework\webframe_project> 
```

GET Method



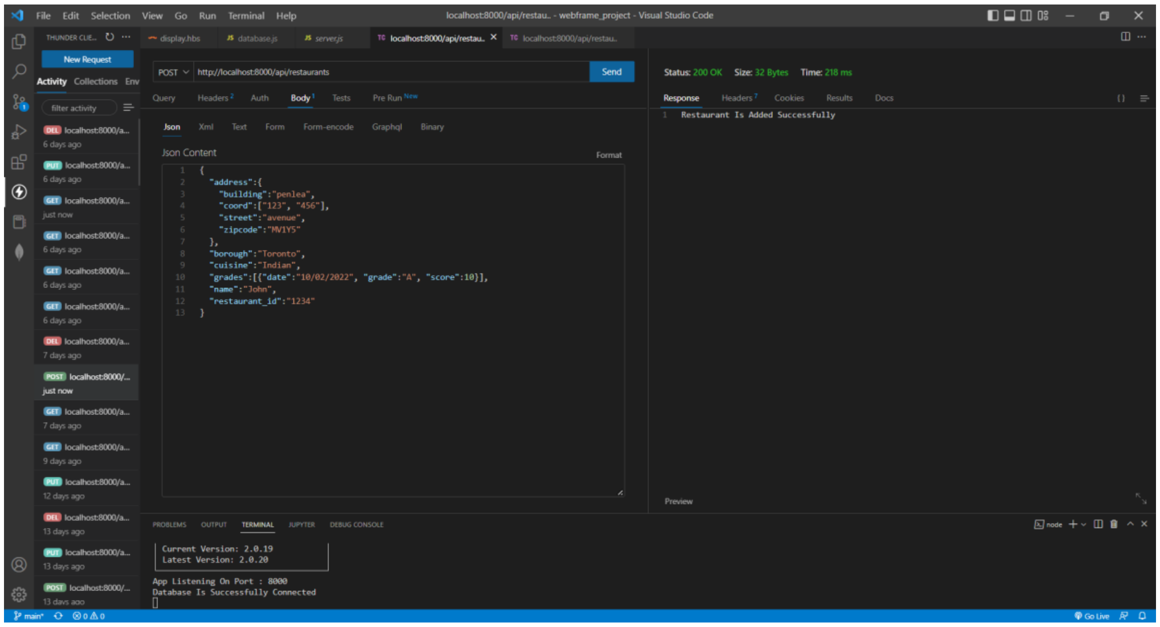
GET http://localhost:8000/api/restaurants/63861ca0c9b7adebeb01b463

Status: 200 OK Size: 312 Bytes Time: 71 ms

```
{
  "address": {
    "building": "penlea",
    "coord": [
      "123",
      "456"
    ],
    "street": "avenue",
    "zipcode": "MV1Y5"
  },
  "_id": "63861ca0c9b7adebeb01b463",
  "borough": "Toronto",
  "cuisine": "Indian",
  "grades": [
    {
      "date": "2022-10-02T04:00:00.000Z",
      "grade": "A",
      "score": 10,
      "_id": "63861ca0c9b7adebeb01b464"
    }
  ],
  "name": "Umang",
  "restaurant_id": "1234",
  "__v": 0
}
```

```
Current Version: 2.0.19
Latest Version: 2.0.20

App Listening On Port : 8000
Database Is Successfully Connected
```

POST Method



POST http://localhost:8000/api/restaurants

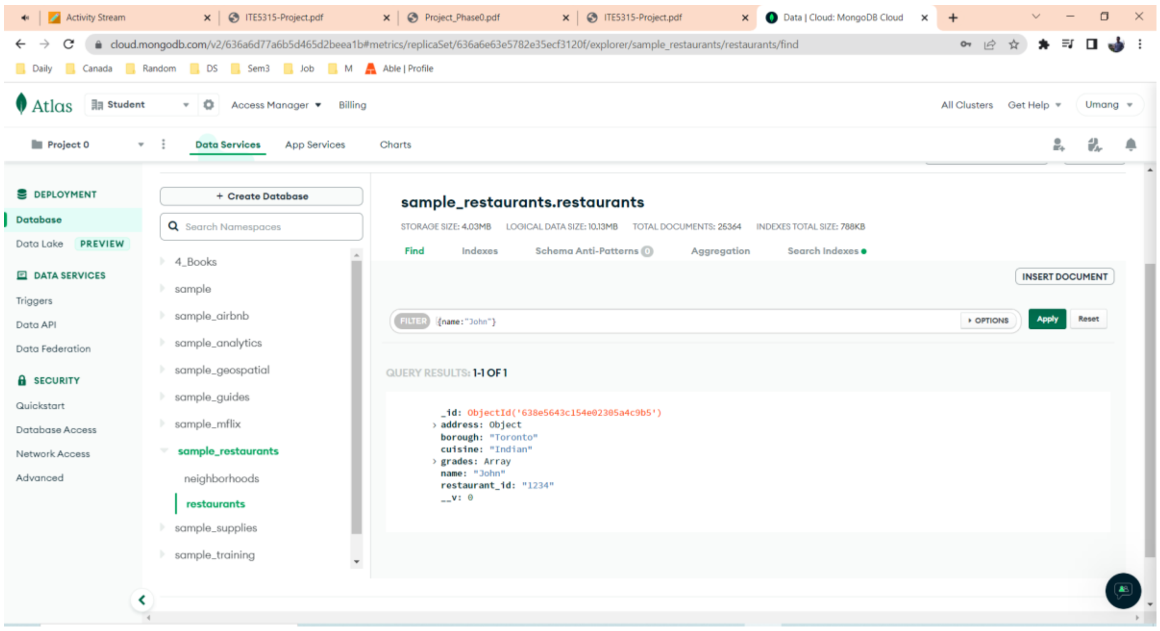
Json Content

```
{
  "address":{
    "building":"penlea",
    "coord":["123", "456"],
    "street":"avenue",
    "zipcode":"MV1Y5"
  },
  "borough":"Toronto",
  "cuisine":"Indian",
  "grades":[{"date":"10/02/2022", "grade":"A", "score":10}],
  "name":"John",
  "restaurant_id":"1234"
}
```

Status: 200 OK Size: 32 Bytes Time: 218 ms

```
Restaurant Is Added Successfully
```

```
Current Version: 2.0.19
Latest Version: 2.0.20

App Listening On Port : 8000
Database Is Successfully Connected
```



sample_restaurants.restaurants

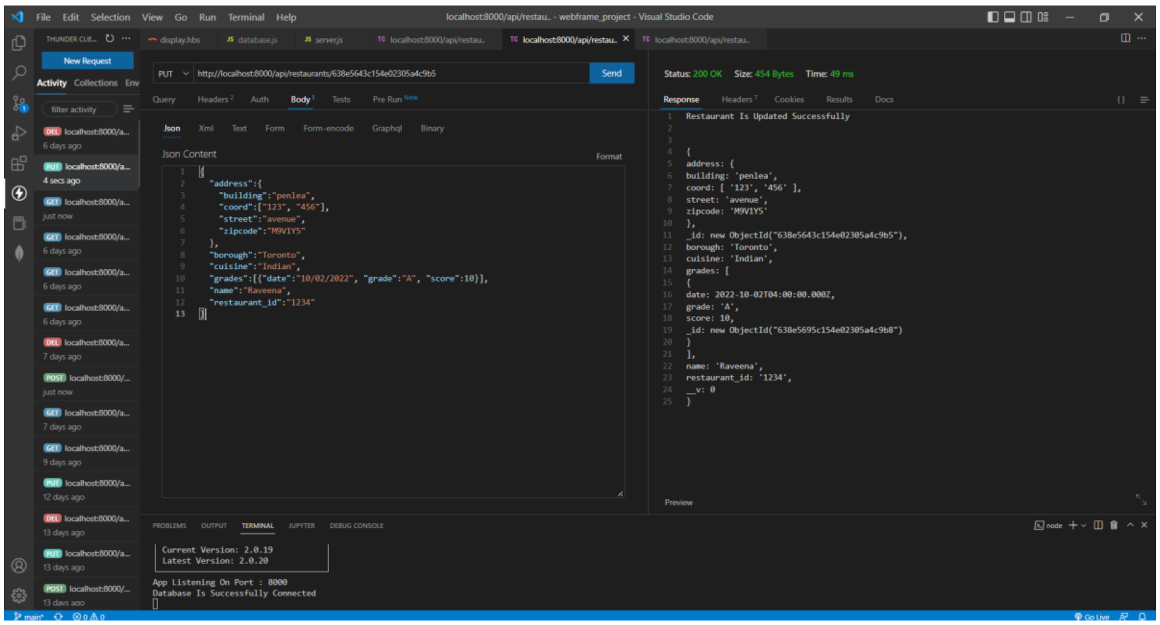
STORAGE SIZE: 4.03MB LOGICAL DATA SIZE: 10.13MB TOTAL DOCUMENTS: 25364 INDEXES TOTAL SIZE: 788KB

FILTER `{name:"John"}`

QUERY RESULTS: 1-1 OF 1

```
  _id: ObjectId('638e5643c154e02305a4c9b5')
> address: Object
  borough: "Toronto"
  cuisine: "Indian"
> grades: Array
  name: "John"
  restaurant_id: "1234"
  __v: 0
```

PUT Method



PUT `http://localhost:8000/api/restaurants/638e5643c154e02305a4c9b5`

Json Content

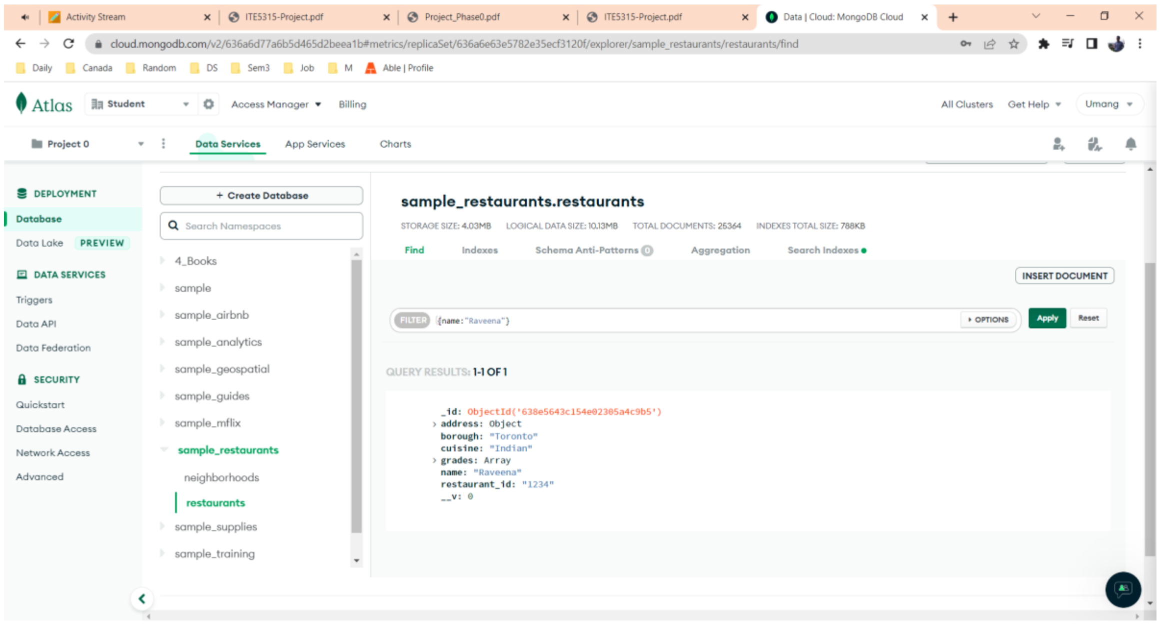
```
{
  "address":{
    "building":"penlea",
    "coord":["123", "456"],
    "street":"avenue",
    "zipcode":"M9V1Y5"
  },
  "borough":"Toronto",
  "cuisine":"Indian",
  "grades":[{"date":"10/02/2022", "grade":"A", "score":10}],
  "name":"Raveena",
  "restaurant_id":"1234"
}
```

Status: 200 OK Size: 454 Bytes Time: 49 ms

Response

```
Restaurant Is Updated Successfully


{
address: {
building: 'penlea',
coord: [ '123', '456' ],
street: 'avenue',
zipcode: 'M9V1Y5'
},
_id: new ObjectId("638e5643c154e02305a4c9b5"),
borough: 'Toronto',
cuisine: 'Indian',
grades: [
{
date: 2022-10-02T04:00:00.000Z,
grade: 'A',
score: 10,
_id: new ObjectId("638e5695c154e02305a4c9b8")
}
],
name: 'Raveena',
restaurant_id: '1234',
__v: 0
}
```

```
Current Version: 2.0.19
Latest Version: 2.0.20

App Listening On Port : 8000
Database Is Successfully Connected
```



sample_restaurants.restaurants

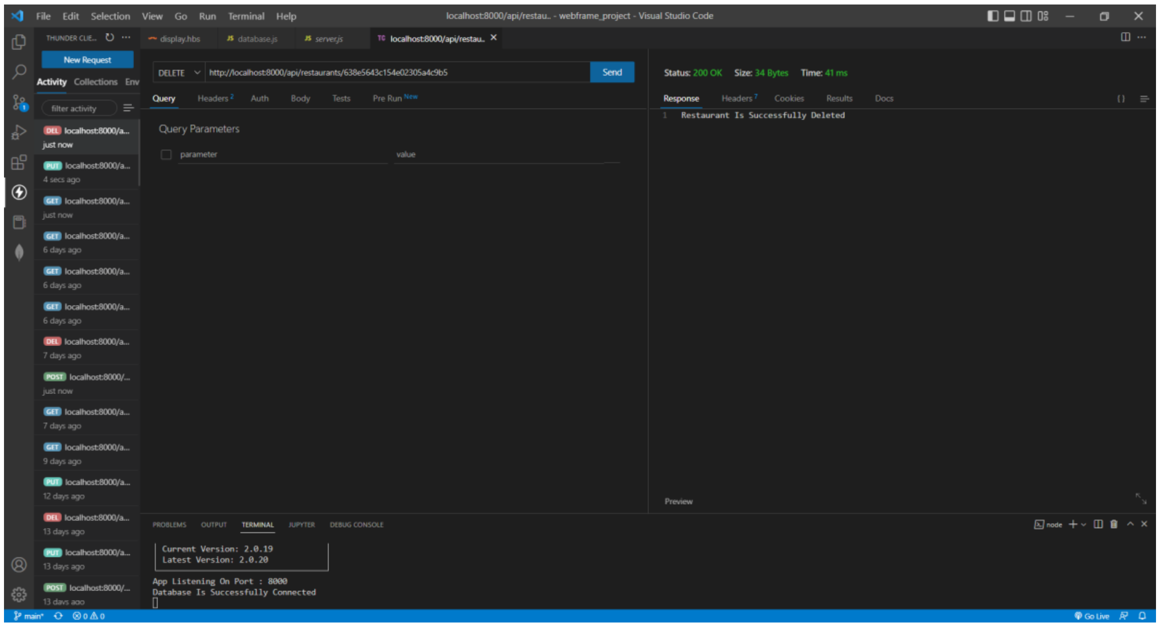
STORAGE SIZE: 4.03MB LOGICAL DATA SIZE: 10.13MB TOTAL DOCUMENTS: 25364 INDEXES TOTAL SIZE: 788KB

FILTER {name:"Raveena"}

QUERY RESULTS: 1-1 OF 1

```
_id: ObjectId('638e5643c154e02305a4c9b5')
> address: Object
borough: "Toronto"
cuisine: "Indian"
> grades: Array
name: "Raveena"
restaurant_id: "1234"
__v: 0
```

DELETE Method



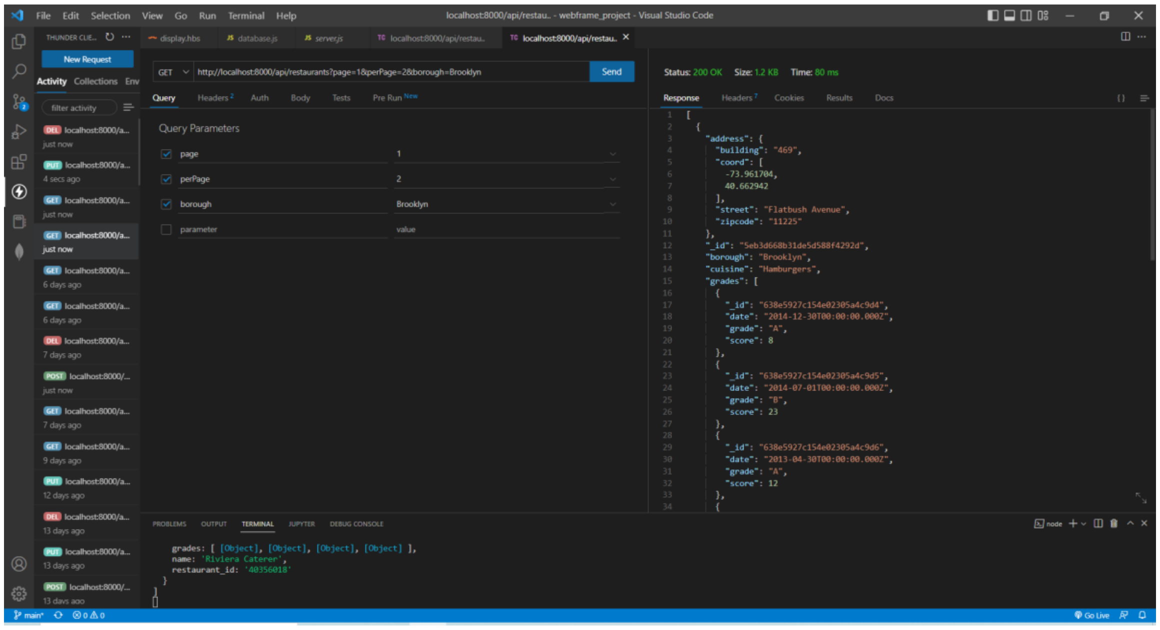
DELETE http://localhost:8000/api/restaurants/638e5643c154e02305a4c9b5

Status: 200 OK Size: 34 Bytes Time: 41 ms

```
Restaurant Is Successfully Deleted
```

```
Current Version: 2.0.19
Latest Version: 2.0.20

App Listening On Port : 8000
Database Is Successfully Connected
```

GET method using page, perPage and borough as query parameters.



GET http://localhost:8000/api/restaurants?page=1&perPage=2&borough=Brooklyn

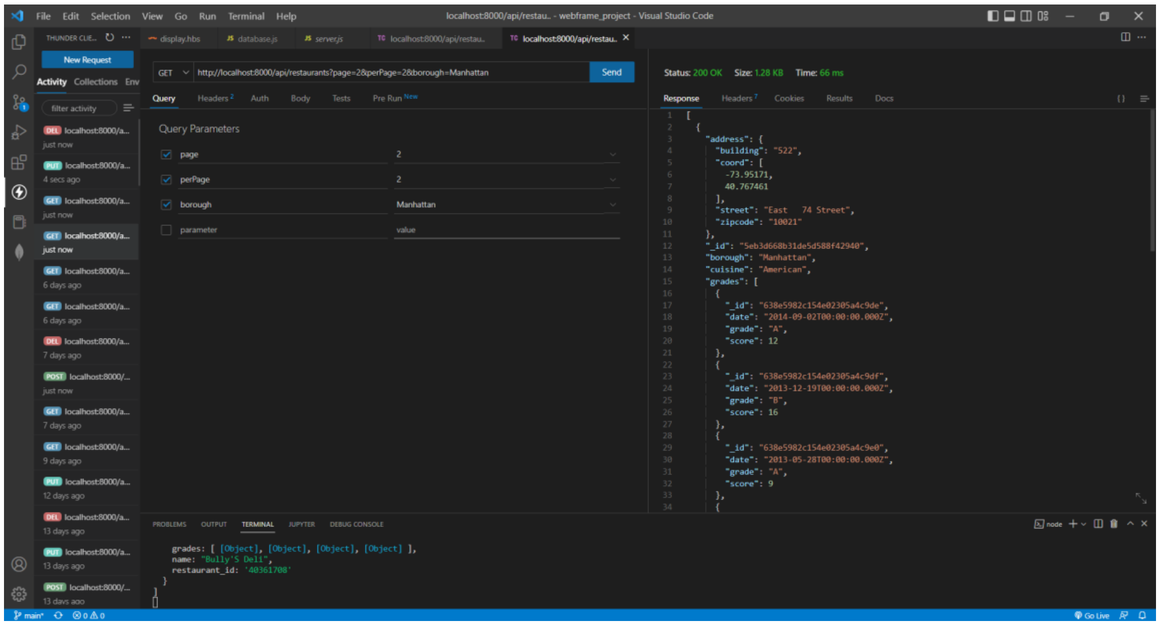
Query Parameters

✓ page	1
✓ perPage	2
✓ borough	Brooklyn
parameter	value

Status: 200 OK Size: 1.2 KB Time: 80 ms

```
[
  {
    "address": {
      "building": "469",
      "coord": [
        -73.961704,
        40.662942
      ],
      "street": "Flatbush Avenue",
      "zipcode": "11225"
    },
    "_id": "5eb3d668b31de5d588f4292d",
    "borough": "Brooklyn",
    "cuisine": "Hamburgers",
    "grades": [
      {
        "_id": "638e5927c154e02305a4c9d4",
        "date": "2014-12-30T00:00:00.000Z",
        "grade": "A",
        "score": 8
      },
      {
        "_id": "638e5927c154e02305a4c9d5",
        "date": "2014-07-01T00:00:00.000Z",
        "grade": "B",
        "score": 23
      },
      {
        "_id": "638e5927c154e02305a4c9d6",
        "date": "2013-04-30T00:00:00.000Z",
        "grade": "A",
        "score": 12
      },
      {
```

```
    grades: [ [Object], [Object], [Object], [Object] ],
    name: 'Riviera Caterer',
    restaurant_id: '40356018'
  }
]
```



GET http://localhost:8000/api/restaurants?page=2&perPage=2&borough=Manhattan

Query Parameters

✓ page	2
✓ perPage	2
✓ borough	Manhattan
parameter	value

Status: 200 OK Size: 1.28 KB Time: 66 ms

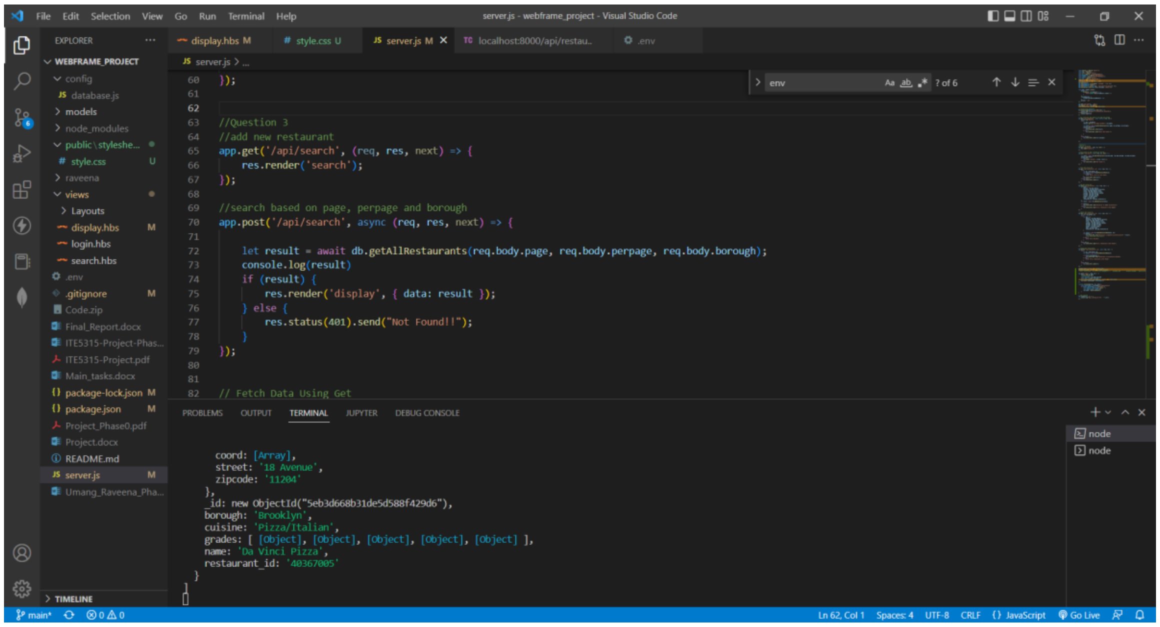
```
[
  {
    "address": {
      "building": "522",
      "coord": [
        -73.95171,
        40.767461
      ],
      "street": "East   74 Street",
      "zipcode": "10021"
    },
    "_id": "5eb3d668b31de5d588f42940",
    "borough": "Manhattan",
    "cuisine": "American",
    "grades": [
      {
        "_id": "638e5982c154e02305a4c9de",
        "date": "2014-09-02T00:00:00.000Z",
        "grade": "A",
        "score": 12
      },
      {
        "_id": "638e5982c154e02305a4c9df",
        "date": "2013-12-19T00:00:00.000Z",
        "grade": "B",
        "score": 16
      },
      {
        "_id": "638e5982c154e02305a4c9e0",
        "date": "2013-05-28T00:00:00.000Z",
        "grade": "A",
        "score": 9
      },
      {
```

```
    grades: [ [Object], [Object], [Object], [Object] ],
    name: "Bully'S Deli",
    restaurant_id: '40361708'
  }
]
```

Question 3:

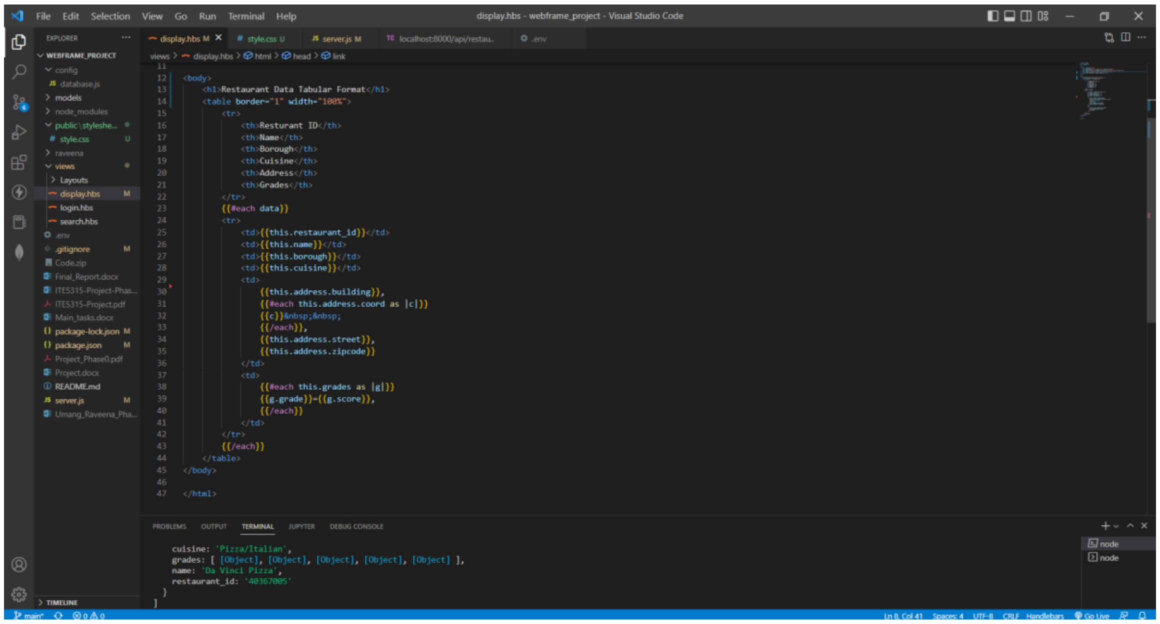
(Describe the major steps for designing the FORM/UI, how you test this program, add some screenshots of the output)

Code Logic:



```
});


//Question 3
//add new restaurant
app.get('/api/search', (req, res, next) => {
    res.render('search');
});

//search based on page, perpage and borough
app.post('/api/search', async (req, res, next) => {

    let result = await db.getAllRestaurants(req.body.page, req.body.perpage, req.body.borough);
    console.log(result)
    if (result) {
        res.render('display', { data: result });
    } else {
        res.status(401).send("Not Found!!");
    }
});


// Fetch Data Using Get
```

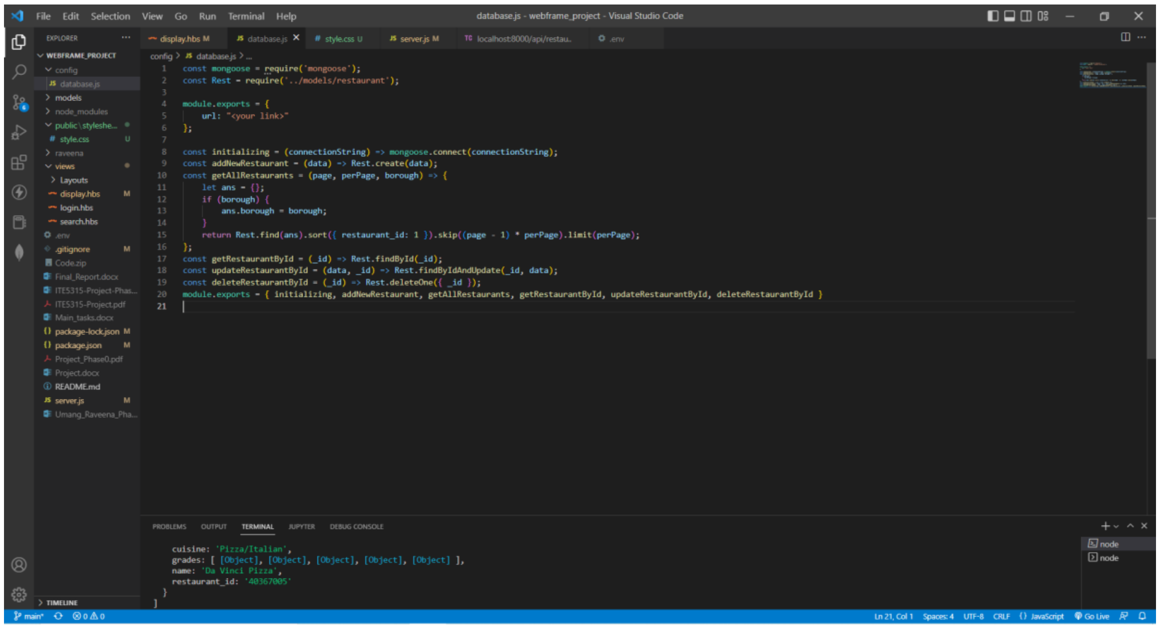
```
      coord: [Array],
      street: '18 Avenue',
      zipcode: '11204'
    },
    _id: new ObjectId("5eb3d668b31de5d588f429d6"),
    borough: 'Brooklyn',
    cuisine: 'Pizza/Italian',
    grades: [ [Object], [Object], [Object], [Object], [Object] ],
    name: 'Da Vinci Pizza',
    restaurant_id: '40367005'
  }
]
```



```
<body>
    <h1>Restaurant Data Tabular Format</h1>
    <table border="1" width="100%">
        <tr>
            <th>Resturant ID</th>
            <th>Name</th>
            <th>Borough</th>
            <th>Cuisine</th>
            <th>Address</th>
            <th>Grades</th>
        </tr>
        {{#each data}}
        <tr>
            <td>{{this.restaurant_id}}</td>
            <td>{{this.name}}</td>
            <td>{{this.borough}}</td>
            <td>{{this.cuisine}}</td>
            <td>
                {{this.address.building}},
                {{#each this.address.coord as |c|}}
                {{c}}&nbsp;&nbsp;
                {{/each}},
                {{this.address.street}},
                {{this.address.zipcode}}
            </td>
            <td>
                {{#each this.grades as |g|}}
                {{g.grade}}={{g.score}},
                {{/each}}
            </td>
        </tr>
        {{/each}}
    </table>
</body>

</html>
```

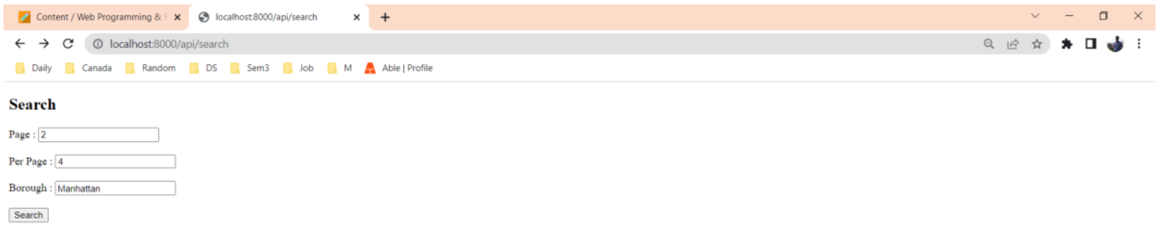
```
    cuisine: 'Pizza/Italian',
    grades: [ [Object], [Object], [Object], [Object], [Object] ],
    name: 'Da Vinci Pizza',
    restaurant_id: '40367005'
  }
]
```

Database.js file



```
const mongoose = require('mongoose');
const Rest = require('../models/restaurant');

module.exports = {
    url: "<your link>"
};

const initializing = (connectionString) => mongoose.connect(connectionString);
const addNewRestaurant = (data) => Rest.create(data);
const getAllRestaurants = (page, perPage, borough) => {
    let ans = {};
    if (borough) {
        ans.borough = borough;
    }
    return Rest.find(ans).sort({ restaurant_id: 1 }).skip((page - 1) * perPage).limit(perPage);
};
const getRestaurantById = (_id) => Rest.findById(_id);
const updateRestaurantById = (data, _id) => Rest.findByIdAndUpdate(_id, data);
const deleteRestaurantById = (_id) => Rest.deleteOne({ _id });
module.exports = { initializing, addNewRestaurant, getAllRestaurants, getRestaurantById, updateRestaurantById, deleteRestaurantById }
```

```
    cuisine: 'Pizza/Italian',
    grades: [ [Object], [Object], [Object], [Object], [Object] ],
    name: 'Da Vinci Pizza',
    restaurant_id: '40367005'
  }
]
```



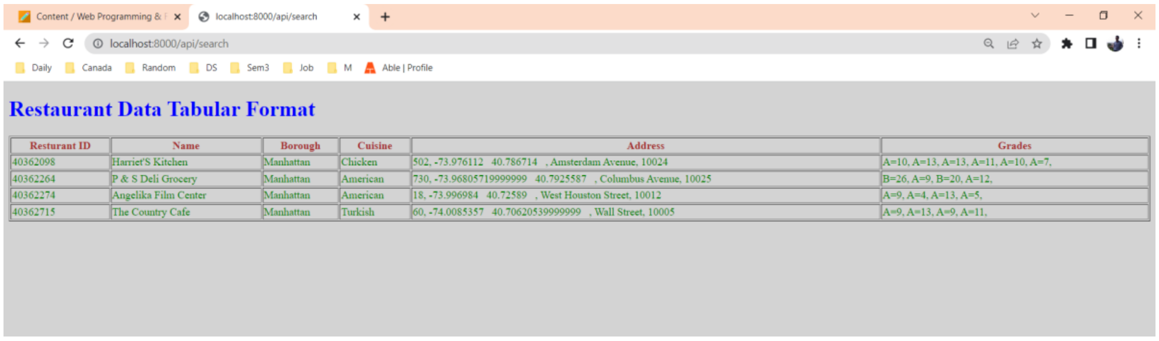
Search

Page : 2

Per Page : 4

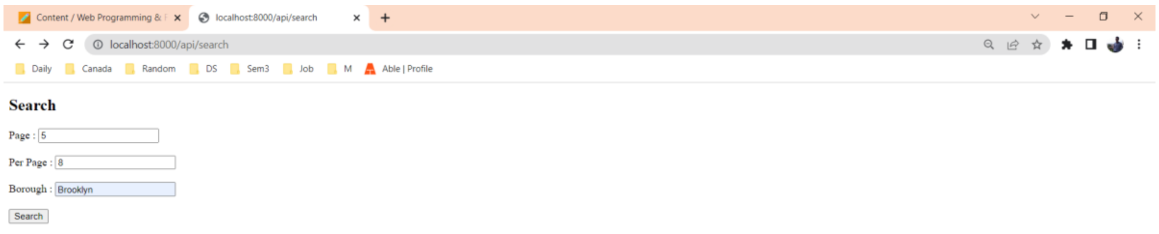
Borough : Manhattan

Search



Restaurant Data Tabular Format

Resturant ID	Name	Borough	Cuisine	Address	Grades
40362098	Harriet'S Kitchen	Manhattan	Chicken	502, -73.976112 40.786714 , Amsterdam Avenue, 10024	A=10, A=13, A=13, A=11, A=10, A=7,
40362264	P & S Deli Grocery	Manhattan	American	730, -73.96805719999999 40.7925587 , Columbus Avenue, 10025	B=26, A=9, B=20, A=12,
40362274	Angelika Film Center	Manhattan	American	18, -73.996984 40.72589 , West Houston Street, 10012	A=9, A=4, A=13, A=5,
40362715	The Country Cafe	Manhattan	Turkish	60, -74.0085357 40.70620539999999 , Wall Street, 10005	A=9, A=13, A=9, A=11,



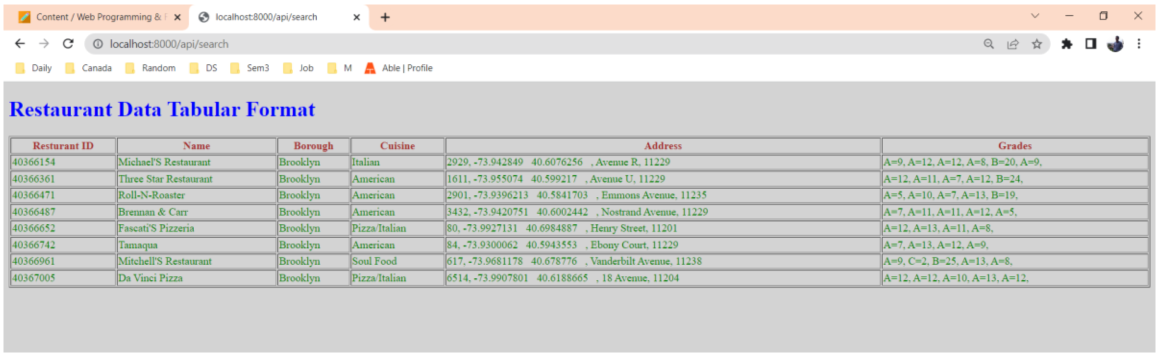
Search

Page : 5

Per Page : 8

Borough : Brooklyn

Search



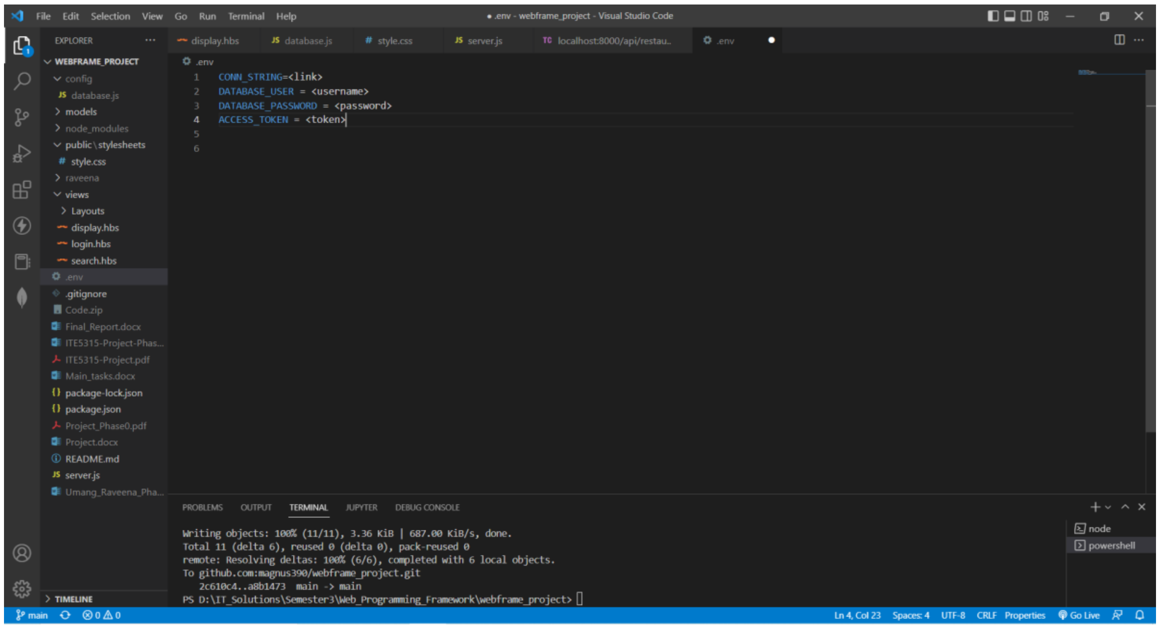
Restaurant Data Tabular Format

Resturant ID	Name	Borough	Cuisine	Address	Grades
40366154	Michael'S Restaurant	Brooklyn	Italian	2929, -73.942849 40.6076256 , Avenue R, 11229	A=9, A=12, A=12, A=8, B=20, A=9,
40366361	Three Star Restaurant	Brooklyn	American	1611, -73.955074 40.599217 , Avenue U, 11229	A=12, A=11, A=7, A=12, B=24,
40366471	Roll-N-Roaster	Brooklyn	American	2901, -73.9396213 40.5841703 , Emmons Avenue, 11235	A=5, A=10, A=7, A=13, B=19,
40366487	Brennan & Carr	Brooklyn	American	3432, -73.9420751 40.6002442 , Nostrand Avenue, 11229	A=7, A=11, A=11, A=12, A=5,
40366652	Fascati'S Pizzeria	Brooklyn	Pizza/Italian	80, -73.9927131 40.6984887 , Henry Street, 11201	A=12, A=13, A=11, A=8,
40366742	Tamaqua	Brooklyn	American	84, -73.9300062 40.5943553 , Ebony Court, 11229	A=7, A=13, A=12, A=9,
40366961	Mitchell'S Restaurant	Brooklyn	Soul Food	617, -73.9681178 40.678776 , Vanderbilt Avenue, 11238	A=9, C=2, B=25, A=13, A=8,
40367005	Da Vinci Pizza	Brooklyn	Pizza/Italian	6514, -73.9907801 40.6188665 , 18 Avenue, 11204	A=12, A=12, A=10, A=13, A=12,

Question 4:

(Describe the major steps for implementing security features, how you test this program, add some screenshots of the output)

The .env file consists of username, password, and connection string. And the same variable I have used in our program.



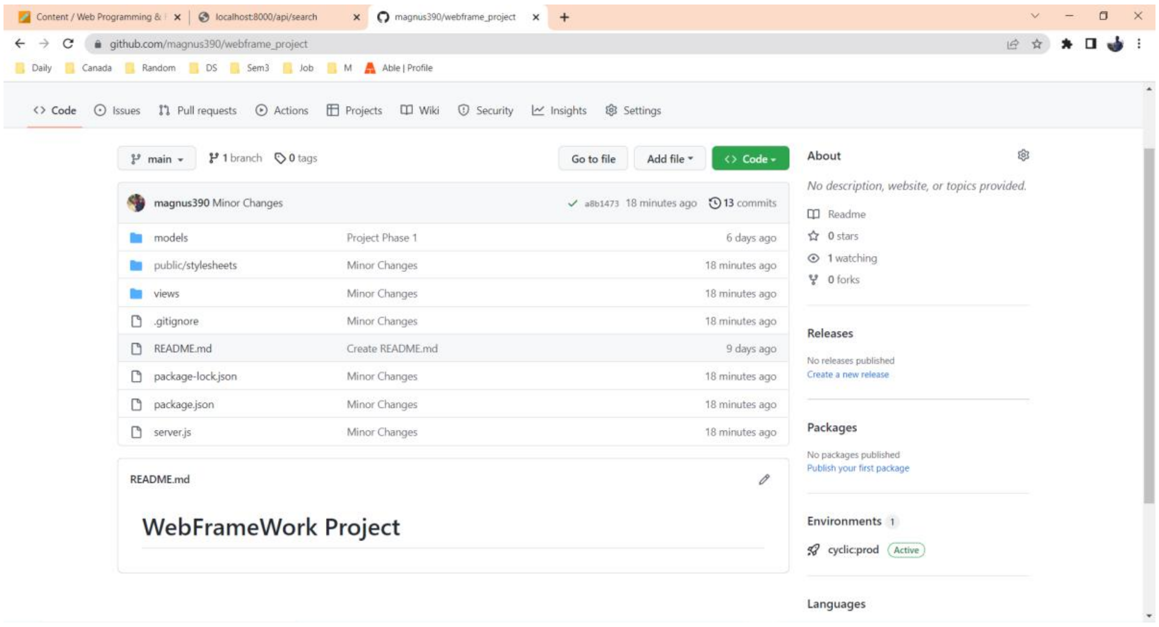
```
CONN_STRING=<link>
DATABASE_USER = <username>
DATABASE_PASSWORD = <password>
ACCESS_TOKEN = <token>
```

```
Writing objects: 100% (11/11), 3.36 KiB | 687.00 KiB/s, done.
Total 11 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), completed with 6 local objects.
To github.com:magnus390/webframe_project.git
   2c610c4..a8b1473  main -> main
PS D:\IT_Solutions\Semester3\Web_Programming_Framework\webframe_project>
```

Question 5:

(Describe the major steps for deployment)

I have created the deployment using cyclic as mentioned and the app has started running



magnus390 Minor Changes ✓ a8b1473 18 minutes ago 13 commits

models	Project Phase 1	6 days ago
public/stylesheets	Minor Changes	18 minutes ago
views	Minor Changes	18 minutes ago
.gitignore	Minor Changes	18 minutes ago
README.md	Create README.md	9 days ago
package-lock.json	Minor Changes	18 minutes ago
package.json	Minor Changes	18 minutes ago
server.js	Minor Changes	18 minutes ago

README.md

WebFrameWork Project

About

No description, website, or topics provided.

Readme
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

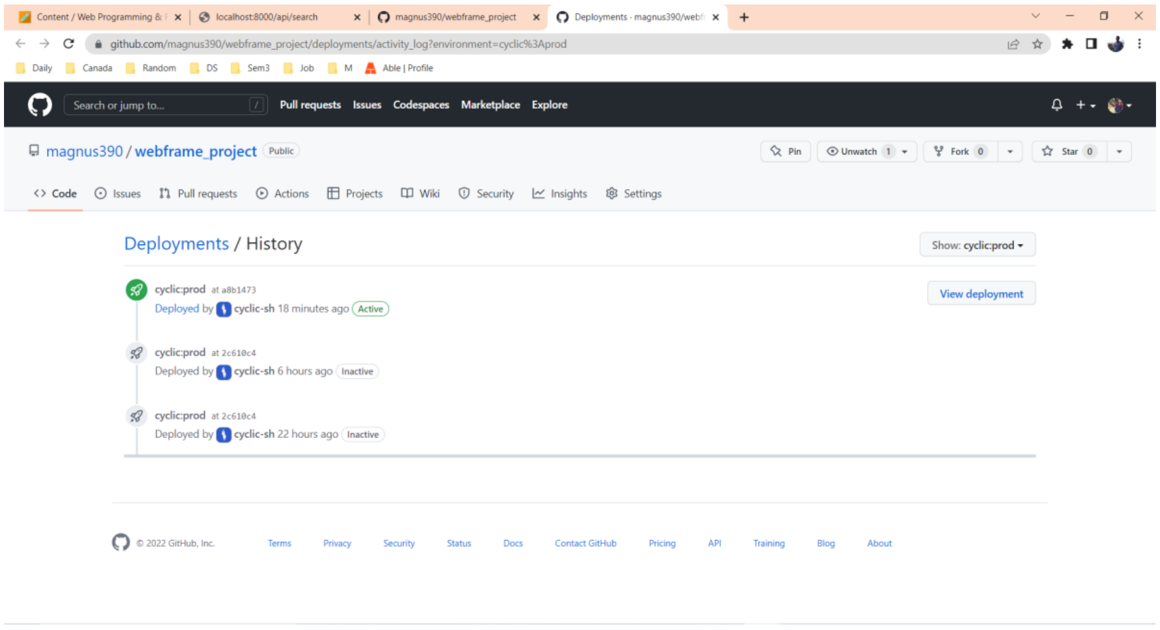
Packages

No packages published
Publish your first package

Environments 1

cyclic:prod Active

Languages


Content / Web Programming &
localhost:8000/api/search
magnus390/webframe_project
Deployments · magnus390/webf
github.com/magnus390/webframe_project/deployments/activity_log?environment=cyclic%3Aprod
Daily
Canada
Random
DS
Sem3
Job
M
Able | Profile
Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
magnus390 / webframe_project
Public
Pin
Unwatch 1
Fork 0
Star 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings

Deployments / History

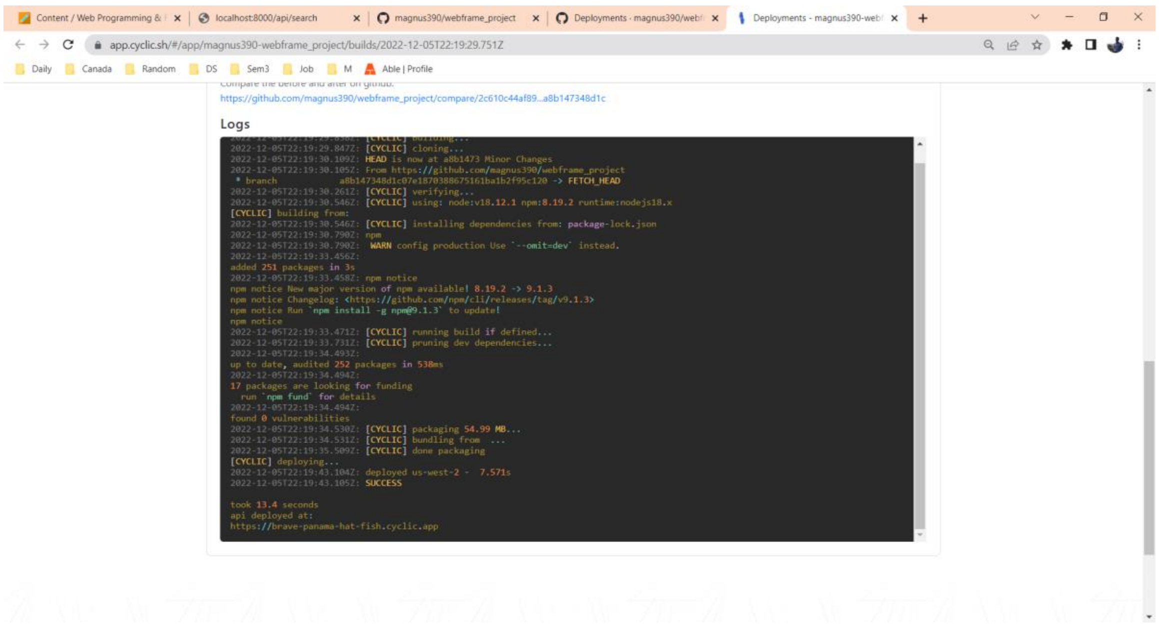
Show: cyclic:prod

cyclic:prod at a8b1473
Deployed by cyclic-sh 18 minutes ago Active

View deployment

cyclic:prod at 2c610c4
Deployed by cyclic-sh 6 hours ago Inactive

cyclic:prod at 2c610c4
Deployed by cyclic-sh 22 hours ago Inactive



https://github.com/magnus390/webframe_project/compare/2c610c44af89...a8b147348d1c

Logs

```
2022-12-05T22:19:29.847Z: [CYCLIC] cloning...
2022-12-05T22:19:30.109Z: HEAD is now at a8b1473 Minor Changes
2022-12-05T22:19:30.105Z: From https://github.com/magnus390/webframe_project
 * branch            a8b147348d1c07e1870388675161ba1b2f95c120 -> FETCH_HEAD
2022-12-05T22:19:30.261Z: [CYCLIC] verifying...
2022-12-05T22:19:30.546Z: [CYCLIC] using: node:v18.12.1 npm:8.19.2 runtime:nodejs18.x
[CYCLIC] building from:
2022-12-05T22:19:30.546Z: [CYCLIC] installing dependencies from: package-lock.json
2022-12-05T22:19:30.790Z: npm
2022-12-05T22:19:30.790Z:  WARN config production Use `--omit=dev` instead.
2022-12-05T22:19:33.456Z:
added 251 packages in 3s
2022-12-05T22:19:33.458Z: npm notice
npm notice New major version of npm available! 8.19.2 -> 9.1.3
npm notice Changelog: <https://github.com/npm/cli/releases/tag/v9.1.3>
npm notice Run `npm install -g npm@9.1.3` to update!
npm notice
2022-12-05T22:19:33.471Z: [CYCLIC] running build if defined...
2022-12-05T22:19:33.731Z: [CYCLIC] pruning dev dependencies...
2022-12-05T22:19:34.493Z:
up to date, audited 252 packages in 538ms
2022-12-05T22:19:34.494Z:
17 packages are looking for funding
  run `npm fund` for details
2022-12-05T22:19:34.494Z:
found 0 vulnerabilities
2022-12-05T22:19:34.530Z: [CYCLIC] packaging 54.99 MB...
2022-12-05T22:19:34.531Z: [CYCLIC] bundling from  ...
2022-12-05T22:19:35.509Z: [CYCLIC] done packaging
[CYCLIC] deploying...
2022-12-05T22:19:43.104Z: deployed us-west-2 -  7.571s
2022-12-05T22:19:43.105Z: SUCCESS

took 13.4 seconds
api deployed at:
https://brave-panama-hat-fish.cyclic.app
```

Question 6:

(Describe the major steps for designing the bonus question)

Not implemented the bonus question

Summary

(Describe how did you divide the work, share your feedback about this project like new points that you learn, challenges, …)

Umang – 1, 2, 4, 6
Raveena – 3, 5, 6

1. We learnt a lot about crud operations using route in node/express.
2. Got a very nice idea on Atlas and interacting with the data.
3. We also got knowledge on how to use JWT token and apply different security features in our app so that only authenticated users can have access to the route.
4. We also did parallel programming on GitHub.
5. Finally, we learnt on how to deploy our application on the cyclic.