

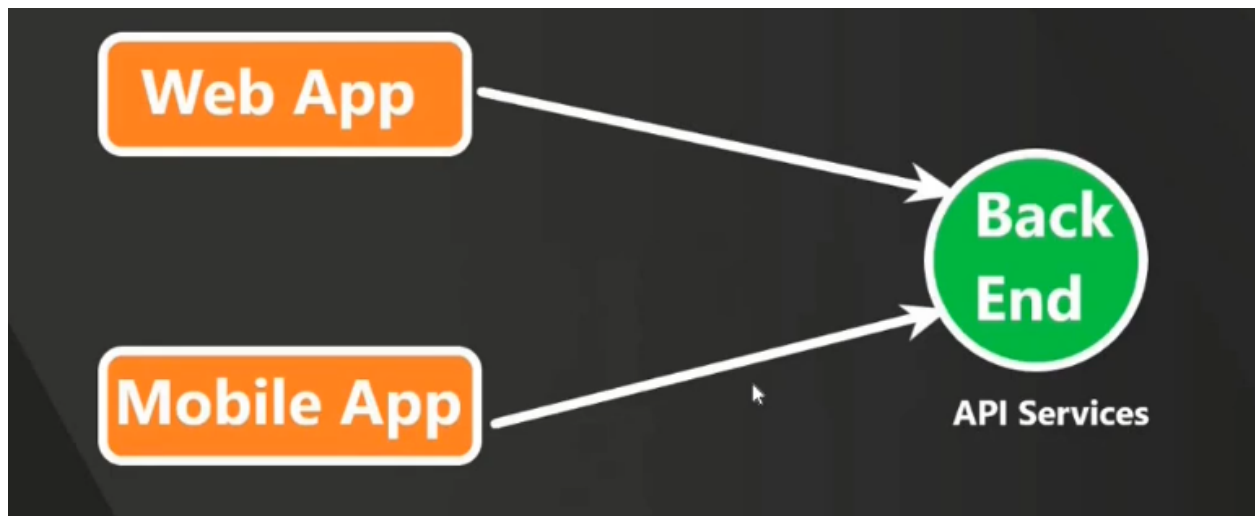
## NodeJs

Node is **NOT** a  
Programming Language

Node is **NOT** a  
Framework

It's an runtime environment  
used for executing  
**JavaScript** code.

We often use node to  
build **back-end** services  
like **API's**



### Features of Node JS

**Highly scalable, Data-intensive  
and real-time apps**

## Special Features in Node JS

**Node is easy to get started**

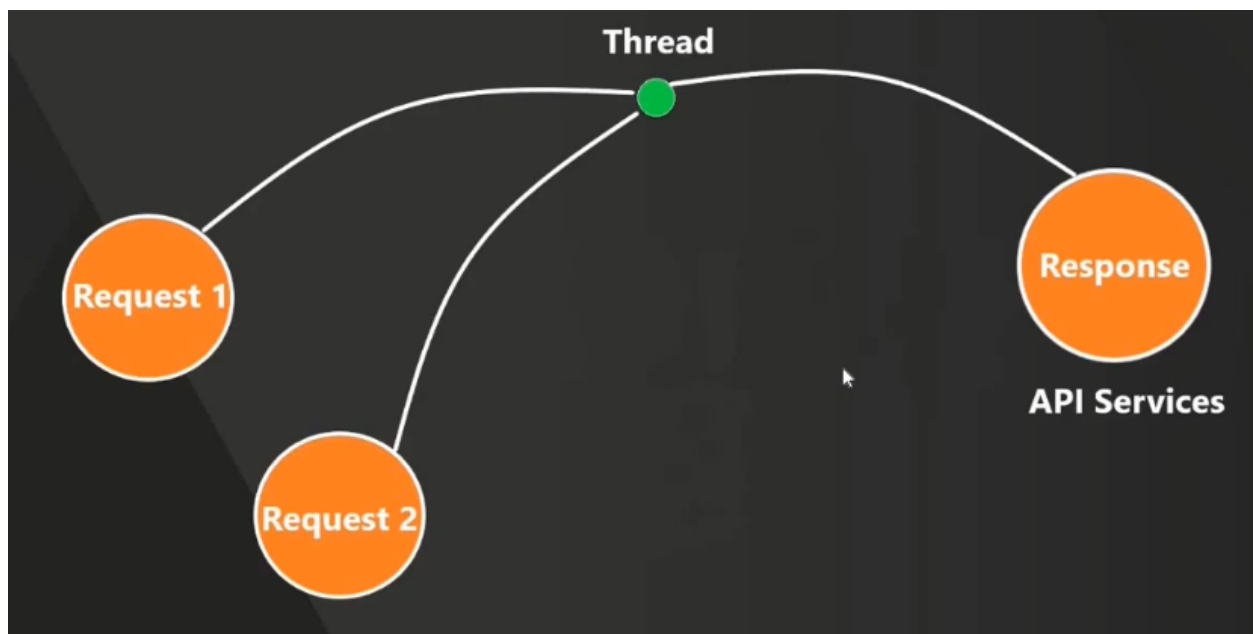
**Superfast & highly scalable services**

**Large ecosystem of open source libs**

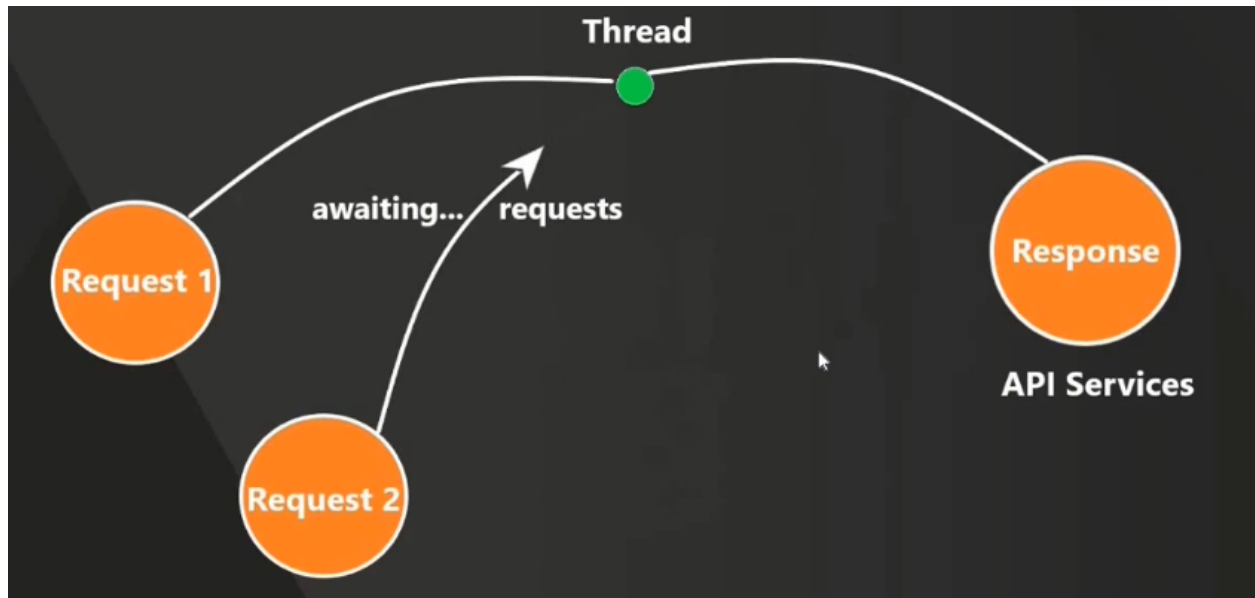
**JavaScript Everywhere**

## NodeJS WorkFlow

→ **Asynchronous / Non-Blocking**



→ **Synchronous / Blocking**



→ NodeJs is non-Blocking / Asynchronous

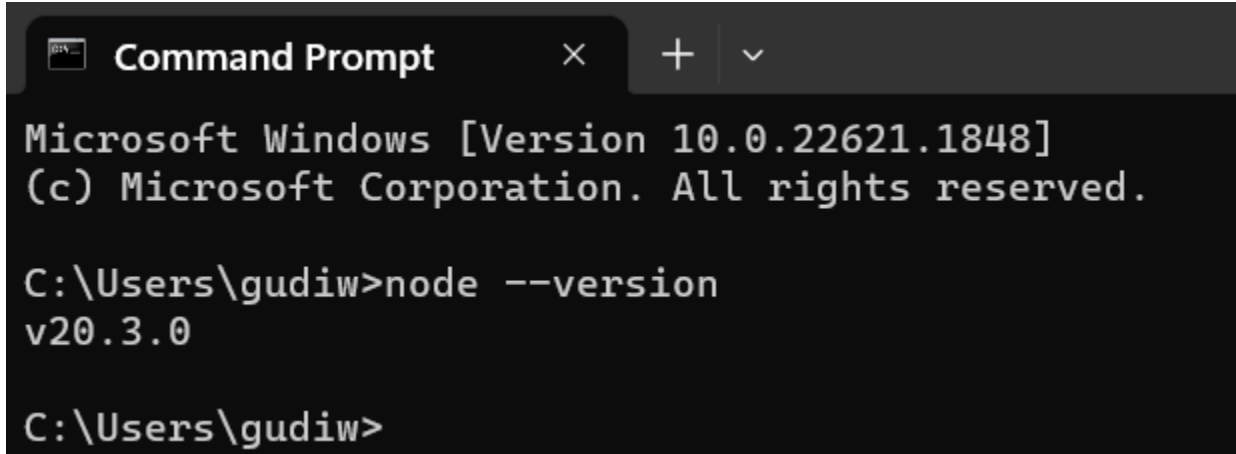


→ <https://nodejs.org/en/download> (Download NodeJS)

Always use LTS Recommended For Most Users

**For Checking NodeJs is installed or not?**

**Go to Cmd (Command Prompt) :**

A screenshot of a Windows Command Prompt window. The title bar says "Command Prompt". The text inside shows the Windows version and copyright information, followed by the command "node --version" and its output "v20.3.0".

```
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gudiw>node --version
v20.3.0

C:\Users\gudiw>
```

```
C:\Users\gudiw>node --version
v20.3.0
```

```
C:\Users\gudiw>cd Documents
```

```
C:\Users\gudiw\Documents> cd tuts
```

```
C:\Users\gudiw\Documents\tuts>mkdir NodeJS-TSH
```

```
C:\Users\gudiw\Documents\tuts>cd NodeJS-TSH
```

```
C:\Users\gudiw\Documents\tuts\NodeJS-TSH>code .   ##(Open VS Code)
```

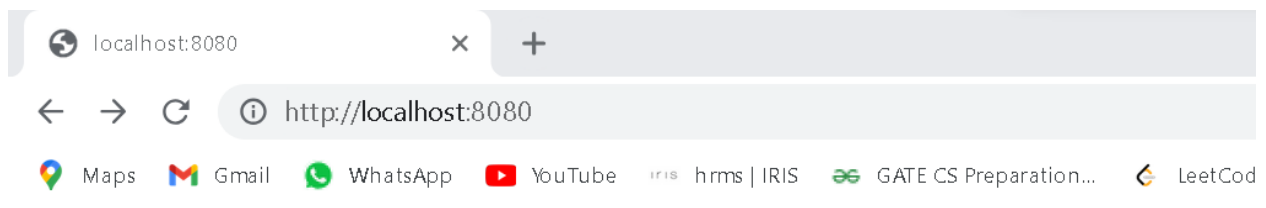
## HTTP Module

server.js

```
var http = require('http');

http.createServer((req, res) => {
  res.write('Hello World!!');
  res.end();
  console.log("Sever Running ....");
}).listen(8080)
```

O/P:



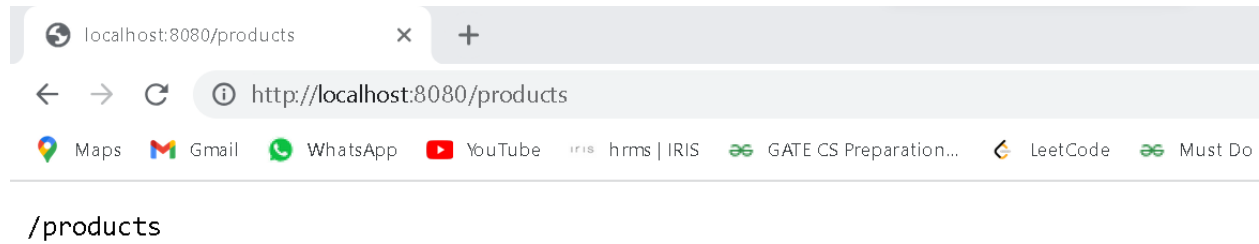
Hello World!!

server.js

```
var http = require('http');

http.createServer((req, res) => {
  //res.write('Hello World!!');
  res.write(req.url);
  res.end();
  console.log("Sever Running ....");
}).listen(8080)
```

**O/P:**



## File System Module

### File Read & Write

**server.js**

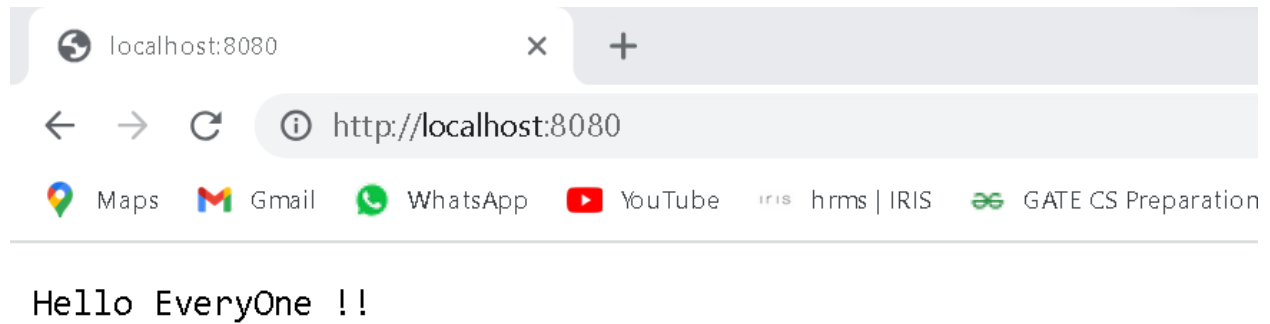
```
var http = require('http');
var fs = require('fs'); //import modules fs ->file System

http.createServer((req, res) => {
    fs.readFile('test.txt', (err, data) =>{
        res.write(data);
        res.end();
    })
}).listen(8080)
```

**test.txt**

**Hello Everyone !!**

**O/P:**



→**Modified “test.txt” file**

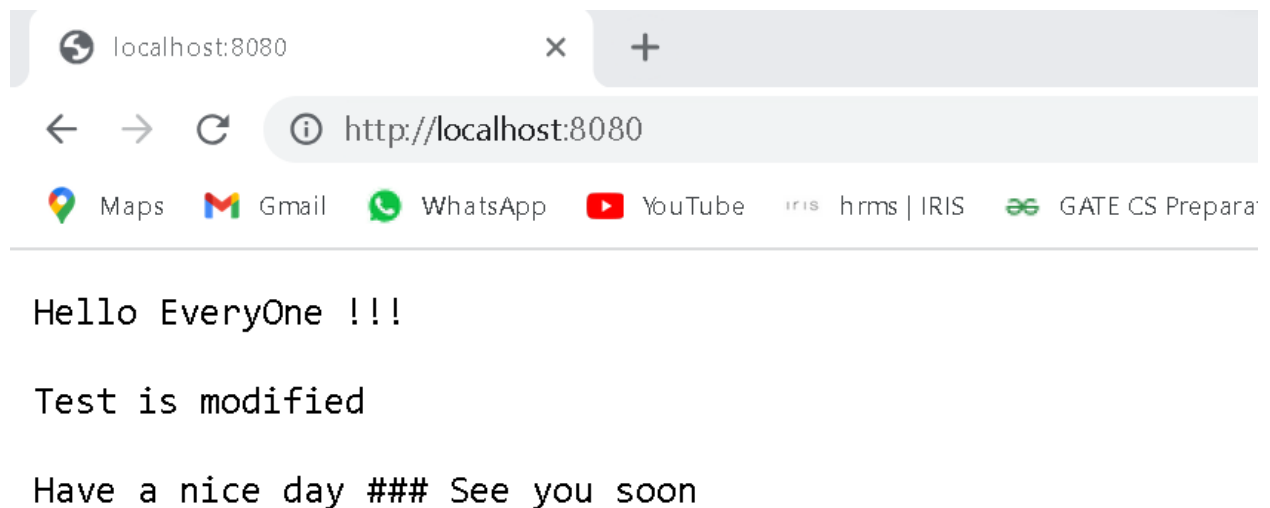
**test.txt**

**Hello Everyone !!!**

**Test is modified**

**Have a nice day ### See you soon**

**O/P:**





→We need add data to the existing File

server.js

```
var http = require('http');
var fs = require('fs'); //import modules fs ->file System

http.createServer((req, res) => {
    fs.appendFile('test.txt', 'Thank You', (err, data) =>{
        res.write(data);
        res.end();
    })
}).listen(8080)
```

O/P:

test.txt is appended with “Thank You”

Hello Everyone !!!

Test is modified

Have a nice day ### See you soon

Thank You

→Overriding the message

server.js

```
var http = require('http');
var fs = require('fs'); //import modules fs ->file System

http.createServer((req, res) => {
    fs.writeFile('test.txt', 'overridden the current data',
    (err, data) =>{
```

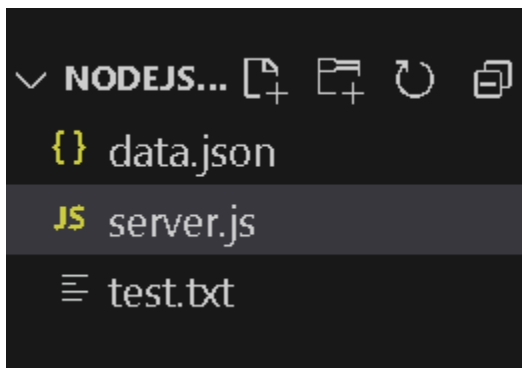
```
    res.write(data);  
    res.end();  
  })  
}).listen(8080)
```

O/P:

→Entire test is replaced new test  
**overridden the current data**

→We can create file using fs.writeFile (Suppose if the given file is not Present)

→Initial

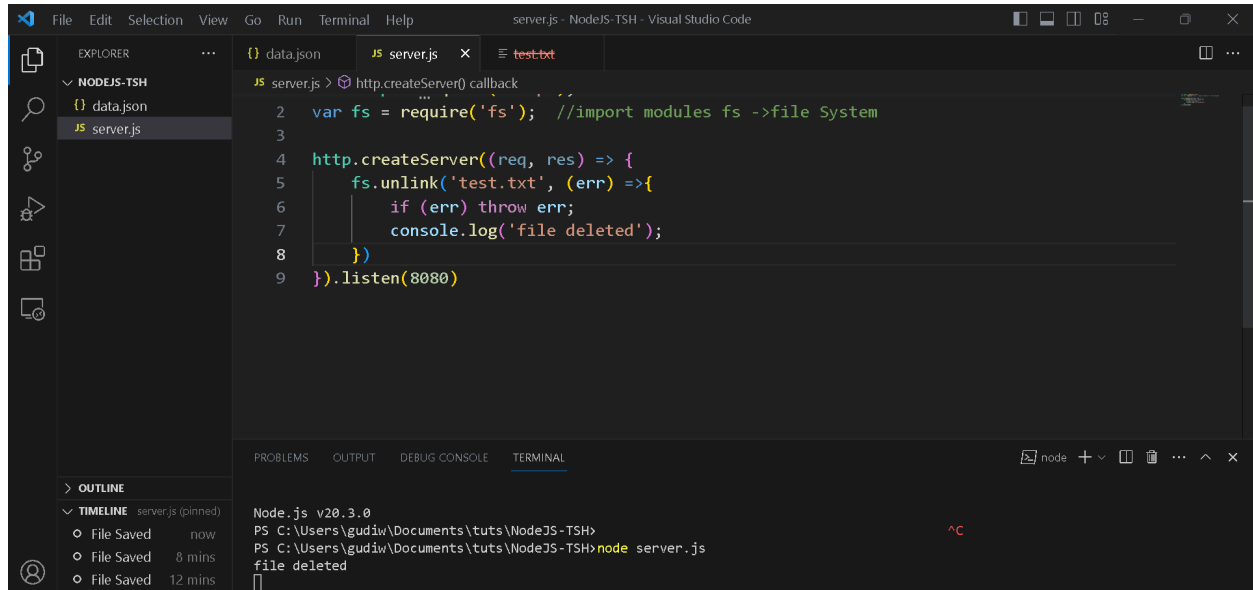


server.js

```
var http = require('http');  
var fs = require('fs'); //import modules fs ->file System  
  
http.createServer((req, res) => {  
  fs.unlink('test.txt', (err) =>{  
    if (err) throw err;  
    console.log('file deleted');  
  })  
})
```

```
}).listen(8080)
```

O/P:



The screenshot shows the Visual Studio Code interface with a project named 'server.js - NodeJS-TSH'. The Explorer sidebar on the left shows a file named 'server.js' selected. The main editor displays the following JavaScript code:

```
1 server.js > http.createServer() callback
2 var fs = require('fs'); //import modules fs ->file System
3
4 http.createServer((req, res) => {
5   fs.unlink('test.txt', (err) =>{
6     if (err) throw err;
7     console.log('file deleted');
8   })
9 }).listen(8080)
```

At the bottom, the TERMINAL panel shows the output of the command 'node server.js':

```
Node.js v20.3.0
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> node server.js
file deleted
```

test.txt is deleted

## URL Module

→Query Params

server.js

```
var url = require('url');

//Query Params
var addr =
'https://www.google.com/search?q=react.js&oq=reac&gs_lcrp=EgZjaHJvbWUqBggEEEUYOzIGCAAQRRg8MgYIARBFgDsyBggCEEUYOzIGCAMQRRg5Mg'
```

```
YIBBBFGDsyBggFEEUYPTIGCAYQRRg8MgYIBxBFGDzSAQg0NjczajBqN6gCALAC
AA&sourceid=chrome&ie=UTF-8'

var q = url.parse(address, true);

console.log(q.host);
```

O/P:

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**node server.js**

**www.google.com**

server.js

```
var url = require('url');

//Query Params
var addr =
'https://www.google.com/search?q=react.js&oq=reac&gs_lcrp=EgZjaHJvbWUqBggEEEUYOzIGCAAQRRg8MgYIARBFGDsyBggCEEUYOzIGCAMQRRg5MgYIBBBFGDsyBggFEEUYPTIGCAYQRRg8MgYIBxBFGDzSAQg0NjczajBqN6gCALAC
AA&sourceid=chrome&ie=UTF-8'

var q = url.parse(addr, true);

console.log(q.pathname);
```

O/P:

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**node server.js**

**/search**

server.js

```
var url = require('url');

//Query Params
var addr =
'https://www.google.com/search?q=react.js&oq=reac&gs_lcrp=EgZjaHJvbWUqBggEEEUYOzIGCAAQRRg8MgYIARBFgDsyBggCEEUYOzIGCAMQRRg5MgYIBBBFGDsyBggFEEUYPTIGCAYQRRg8MgYIBxBFgDzSAQg0NjczaJBqN6gCALACAA&sourceid=chrome&ie=UTF-8'

var q = url.parse(addr, true);

console.log(q.search);
```

O/P”

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**node server.js**

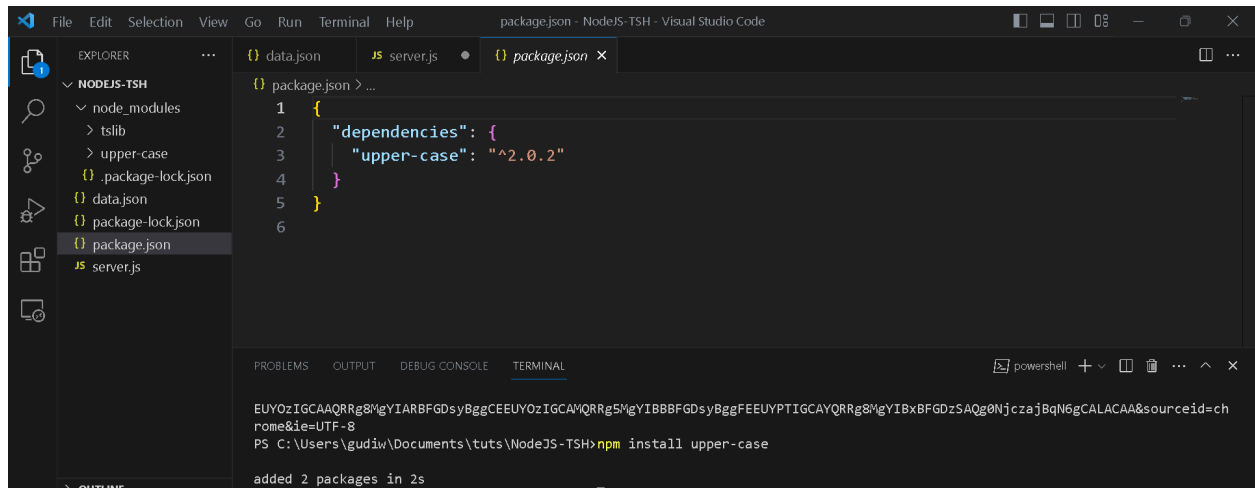
**?q=react.js&oq=reac&gs\_lcrp=EgZjaHJvbWUqBggEEEUYOzIGCAAQRRg8MgYIARBFgDsyBggCEEUYOzIGCAMQRRg5MgYIBBBFGDsyBggFEEUYPTIGCAYQRRg8MgYIBxBFgDzSAQg0NjczaJBqN6gCALACAA&sourceid=chrome&ie=UTF-8**

## Node JS NPM

Using npm we need to install your packages.

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**npm install upper-case**

**added 2 packages in 2s**

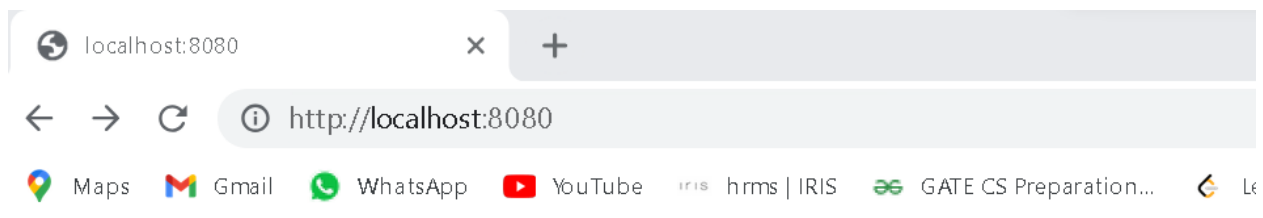


## server.js

```
var http = require('http');
var uc = require('upper-case');

http.createServer((req, res) => {
  res.write(uc.upperCase('Hello, World!'));
  res.end();
}).listen(8080)
```

O/P:



HELLO, WORLD!

## Node JS Email

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**npm install nodemailer**

added 1 package, and audited 4 packages in 3s

found 0 vulnerabilities

**server.js**

```
var nodemailer = require('nodemailer');
var transporter = nodemailer.createTransport({
  service: 'gmail',
  auth: {
    user: 'gudiwadaraghava999@gmail.com',
    pass: 'rag' //Please Give the Correct Answer
  }
});

var options = {
  from : 'gudiwadaraghava999@gmail.com',
  to : 'raghava2k21@gmail.com',
  subject : 'Testing Mail',
  text : 'Thats easy to send mail to others'
}

transporter.sendMail(options, (err, info) =>{
  if(err){
    console.log(err);
  }
  else{
    console.log("Email is Sent" + info.response)
  }
})
```

```
})
```

**O/P:**

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>node server.js
```

```
Error: connect ECONNREFUSED ::1:587
```

```
  at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1574:16) {  
    errno: -4078,  
    code: 'ESOCKET',  
    syscall: 'connect',  
    address: '::1',  
    port: 587,  
    command: 'CONN'  
  }
```

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> node server.js
```

```
Error: Invalid login: 535-5.7.8 Username and Password not accepted. Learn more at  
535 5.7.8 https://support.google.com/mail/?p=BadCredentials
```

```
x18-20020aa784d2000000b0068288aaf23esm4841931pfn.100 - gsmtplib
```

```
  at SMTPConnection._formatError  
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-  
connection\index.js:790:19)  
  at SMTPConnection._actionAUTHComplete  
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-  
connection\index.js:1564:34)  
  at SMTPConnection.<anonymous>  
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-  
connection\index.js:546:26)  
  at SMTPConnection._processResponse  
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-  
connection\index.js:969:20)
```



```

    at SMTPConnection._onData
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-connection\index.js:755:14)
    at SMTPConnection._onSocketData
(C:\Users\gudiw\Documents\tuts\NodeJS-TSH\node_modules\nodemailer\lib\smtp-connection\index.js:193:44)
    code: 'EAUTH',
    response: '535-5.7.8 Username and Password not accepted. Learn more at\n'
+
    '535 5.7.8 https://support.google.com/mail/?p=BadCredentials
x18-20020aa784d2000000b0068288aaf23esm4841931pfn.100 - gsmtplib',
    responseCode: 535,
    command: 'AUTH PLAIN'
  }
}

```

→ We have to give authentication from Our Mail

## Fetching Data from API

server.js

```

var http = require('http');
var fs = require('fs');

http.createServer((req, res) => {
  fs.readFile('data.json', (err, data) =>{
    res.write(data);
    res.end();
    console.log('API is Running..')
  })
}).listen(8080)

```

data.json

```
[
  {
    "id" : "1",
    "name" : "Hyderabad"
  },
  {
    "id" : "2",
    "name" : "Warangal"
  },
  {
    "id" : "3",
    "name" : "Visakhapatnam"
  },
  {
    "id" : "4",
    "name" : "Vijayawada"
  },
  {
    "id" : "5",
    "name" : "Guntur"
  },
  {
    "id" : "6",
    "name" : "Nellore"
  },
  {
    "id" : "7",
    "name" : "kurnool"
  },
  {
    "id" : "8",
```

```
    "name" : "kadapa"
  },
  {
    "id" : "9",
    "name" : "Tirupati"
  },
  {
    "id" : "10",
    "name" : "Anantapuram"
  },
  {
    "id" : "11",
    "name" : "Ongole"
  },
  {
    "id" : "12",
    "name" : "Chittoor"
  },
  {
    "id" : "13",
    "name" : "Machilipatnam"
  },
  {
    "id" : "14",
    "name" : "Chilakaluripeta"
  },
  {
    "id" : "15",
    "name" : "Srikakulam"
  },
  {
    "id" : "16",
    "name" : "Bhimavaram"
```

```
},
{
  "id" : "17",
  "name" : "Hindupur"
},
{
  "id" : "18",
  "name" : "Khammam"
},
{
  "id" : "19",
  "name" : "Peddapalli"
},
{
  "id" : "20",
  "name" : "Suryapet"
},
{
  "id" : "21",
  "name" : "Karimnagar"
},
{
  "id" : "22",
  "name" : "Adilabad"
},
{
  "id" : "23",
  "name" : "Nagarkurnool"
},
{
  "id" : "24",
  "name" : "Nalgonda"
},
```

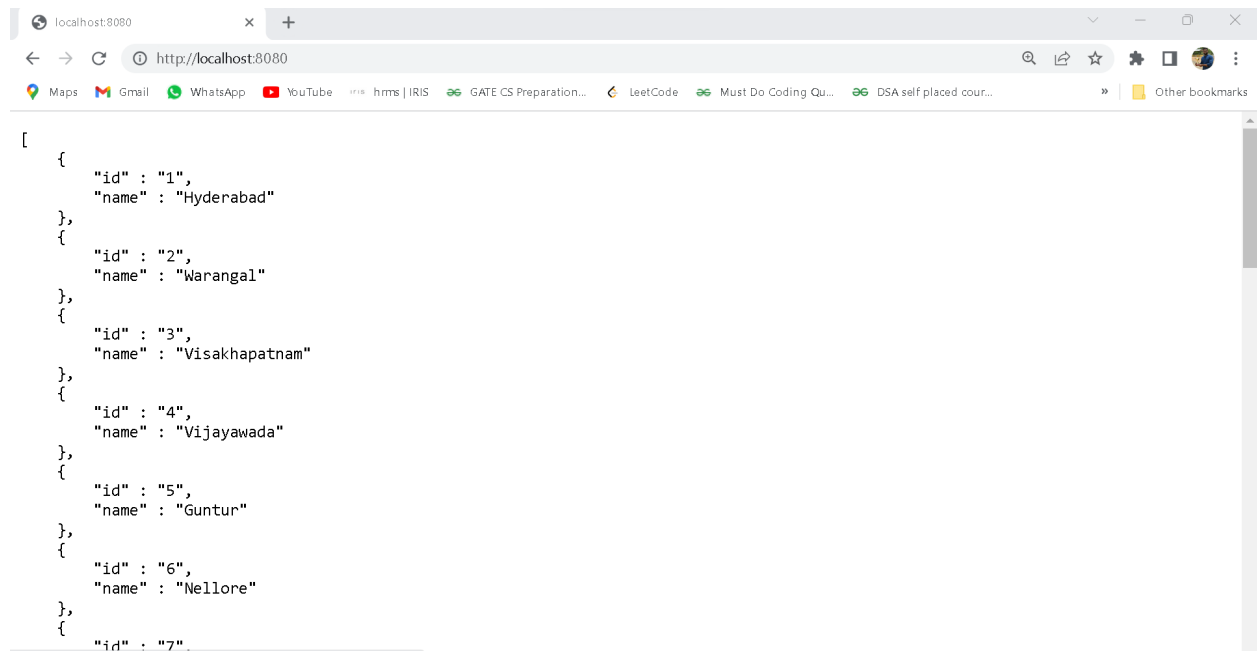
```
{
  "id" : "25",
  "name" : "Rajamma Sircilla"
},
{
  "id" : "26",
  "name" : "Sangareddy"
}
]
```

**O/P:**

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> **node server.js**

API is Running..

API is Running..



A screenshot of a web browser window displaying a JSON array of city data. The browser's address bar shows the URL `http://localhost:8080`. The page content shows a JSON array with seven objects, each representing a city with an 'id' and a 'name'. The cities listed are Hyderabad, Warangal, Visakhapatnam, Vijayawada, Guntur, Nellore, and a partially visible entry for '7'.

```
[
  {
    "id" : "1",
    "name" : "Hyderabad"
  },
  {
    "id" : "2",
    "name" : "Warangal"
  },
  {
    "id" : "3",
    "name" : "Visakhapatnam"
  },
  {
    "id" : "4",
    "name" : "Vijayawada"
  },
  {
    "id" : "5",
    "name" : "Guntur"
  },
  {
    "id" : "6",
    "name" : "Nellore"
  },
  {
    "id" : "7"
  }
]
```



## How to Create API with Node.js Express MongoDB

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> npm install mongoose
```

added 24 packages, and audited 86 packages in 23s

9 packages are looking for funding  
run `npm fund` for details

found 0 vulnerabilities

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> npm install express
```

added 58 packages, and audited 62 packages in 6s

8 packages are looking for funding  
run `npm fund` for details

found 0 vulnerabilities

→**nodemon is for running our server Continuously**

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> npm install -D nodemon
```

added 33 packages, and audited 119 packages in 7s

12 packages are looking for funding  
run `npm fund` for details

found 0 vulnerabilities

```
PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH> npm init
```

This utility will walk you through creating a package.json file.  
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields  
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and  
save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (nodejs-tsh)

version: (1.0.0)

description: **testing api**

entry point: **(server.js)**

git repository:

keywords:

author: **Raghava**

license: (ISC) MIT

About to write to C:\Users\gudiw\Documents\tuts\NodeJS-TSH\package.json:

```
{
  "dependencies": {
    "express": "^4.18.2",
    "mongoose": "^7.3.4",
    "nodemailer": "^6.9.3",
    "upper-case": "^2.0.2"
  },
  "devDependencies": {
    "nodemon": "^3.0.1"
  },
  "scripts": {
    "start": "node server.js"
  }
}
```



```
},  
  "name": "nodejs-tsh",  
  "version": "1.0.0",  
  "description": "testing api",  
  "main": "server.js",  
  "author": "Raghava",  
  "license": "MIT"  
}
```

Is this OK? (yes)

```
{ } package.json ×
{ } package.json > ...
4      mongoose : "7.5.4",
5      "nodemailer": "^6.9.3",
6      "upper-case": "^2.0.2"
7    },
8    "devDependencies": {
9      "nodemon": "^3.0.1"
10   },
11   "scripts": {
12     "start": "node server.js",
13     "server": "nodemon server.js"
14   },
15   "name": "nodejs-tsh",
16   "version": "1.0.0",
17   "description": "testing api",
```

PS C:\Users\gudiw\Documents\tuts\NodeJS-TSH>**npm run server**

> nodejs-tsh@1.0.0 server

> nodemon server.js

[nodemon] 3.0.1

[nodemon] to restart at any time, enter `rs`

[nodemon] watching path(s): \*.\*

[nodemon] watching extensions: js,mjs,cjs,json

[nodemon] starting `node server.js`

Server Running ....

## MangoDB Setup & Atlas

<https://www.mongodb.com/atlas/database>

Go to the Website signUP

1. Create cluster (given cluster name → nodetuts)
2. Create database

### Create Database ×

Database name ?

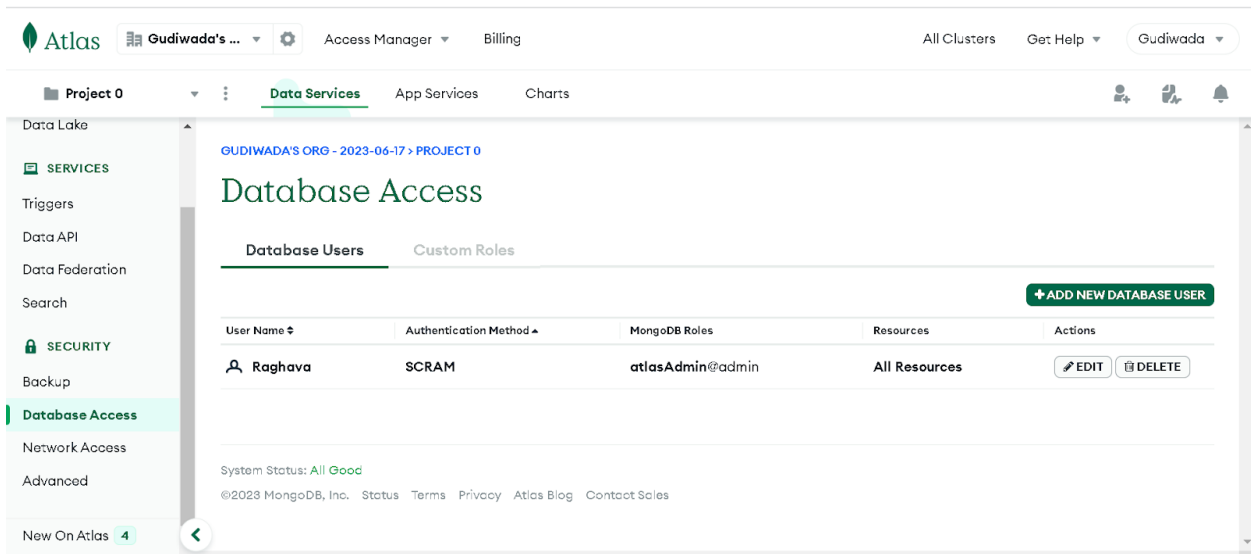
Collection name ?

Additional Preferences

Cancel

Create

# Go to Database Access



## Add new database user

### Add New Database User

Create a database user to grant an application or user, access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding [Access Manager](#)

#### Authentication Method

Password

Certificate

AWS IAM  
(MongoDB 4.4 and up)

MongoDB uses [SCRAM](#) as its default authentication method.


#### Password Authentication

HIDE


Autogenerate Secure Password


Copy


→while adding database user give access to the privileges

**Built-in Role** 

Select one [built-in role](#) for this user.

**Custom Roles** 

Select your [pre-defined custom role\(s\)](#). Create a custom role in the [Custom Roles](#)  tab.

**Specific Privileges** 1 SELECTED 

Select multiple privileges and what database and collection they are associated with.  
Leaving collection blank will grant this role for all collections in the database.

readWrite


@

netninja

Collection\*

Add Specific Privilege

We can also mention whatever the database we want

**Specific Privileges** 1 SELECTED 


Select multiple privileges and what database and collection they are associated with.  
Leaving collection blank will grant this role for all collections in the database.

readWriteAnyDatabase

@

admin

Collection\*



Add Specific Privilege







→After creation of Database user look like this

## Database Access

Database Users

Custom Roles

+ ADD NEW DATABASE USER



User Name ↕	Authentication Method ▲	MongoDB Roles	Resources	Actions
 netninja	SCRAM	readWrite@netninja	All Resources	<div> EDIT</div> <div> DELETE</div>
 Raghava	SCRAM	atlasAdmin@admin	All Resources	<div> EDIT</div> <div> DELETE</div>

# Database Access


Database Users

Custom Roles

+ ADD NEW DATABASE USER

User Name ↕	Authentication Method ▲	MongoDB Roles	Resources	Actions
 netninja	SCRAM	readWriteAnyDatabase@admin	All Resources	<div><div>EDIT</div><div>DELETE</div></div>
 Raghava	SCRAM	atlasAdmin@admin	All Resources	<div><div>EDIT</div><div>DELETE</div></div>

—>Go To Connect

 Gudiwada's ... Access Manager Billing All Clusters Get Help Gudiwada

Project 0 Data Services App Services Charts

DEPLOYMENT

Database

Data Lake

SERVICES

Triggers

Data API

Data Federation

Search

SECURITY

Backup

Database Access

Overview

RealTime

Metrics

Collections

Search

Profiler

Performance Advisor

Online Archive

SANDBOX

NODES

REPLICA SET

Connect

Configuration

...

Sample dataset successfully loaded. Access it in Collections or by connecting with the MongoDB Shell.

Use tags to efficiently label and categorize your clusters. Any tags you apply will display here. [Learn more about tagging.](#)

ADD TAG

REGION Mumbai (ap-south-1)

ac-... shard-00-00.1xi... 

SECONDARY

ac-... shard-00-01.1xi... 

PRIMARY

ac-... shard-00-02.1xi... 

SECONDARY

This is a Shared Tier Cluster

If you need a database that's better for high-performance production applications, upgrade to a dedicated cluster.

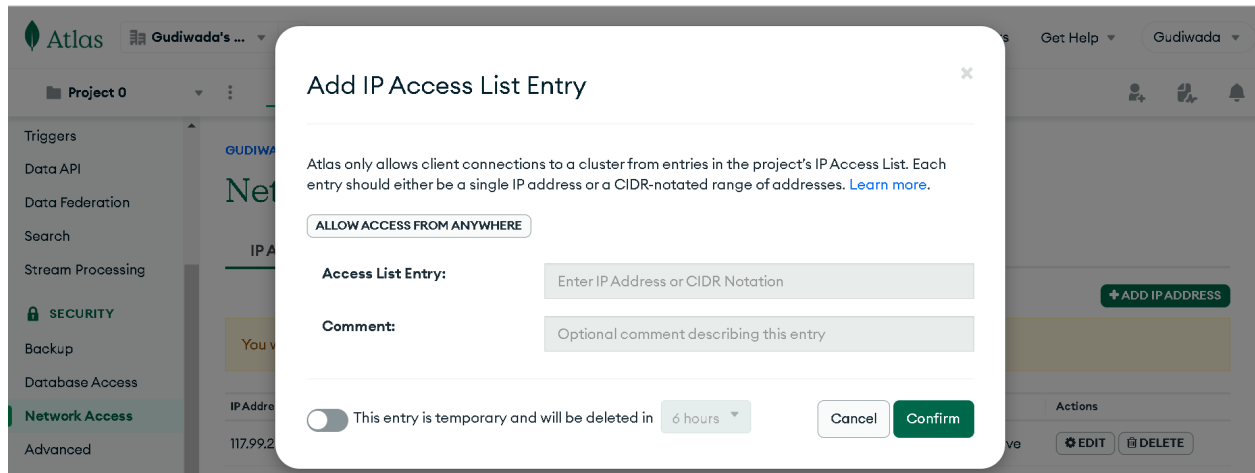
Upgrade

Operations R: 0.01 W: 0 0.02/s

Last 6 Hours

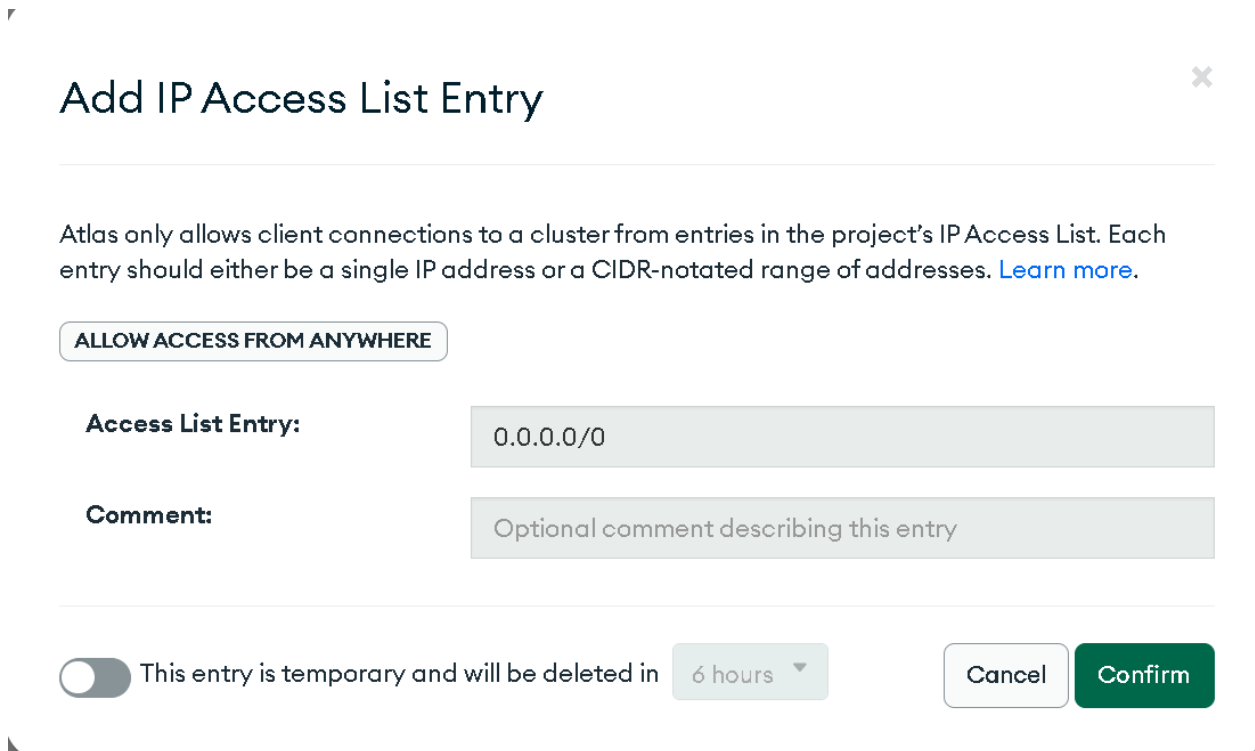
### 3. Add IP Address

→Click on “Allow Access From ANYWHERE”



The screenshot shows the Atlas web interface with a modal dialog titled "Add IP Access List Entry". The dialog contains a close button (X) in the top right corner. Below the title, there is a descriptive text: "Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more.](#)". Below this text is a button labeled "ALLOW ACCESS FROM ANYWHERE". Underneath, there are two input fields: "Access List Entry:" with a placeholder "Enter IP Address or CIDR Notation" and "Comment:" with a placeholder "Optional comment describing this entry". At the bottom, there is a toggle switch labeled "This entry is temporary and will be deleted in" followed by a dropdown menu showing "6 hours". To the right of the toggle are "Cancel" and "Confirm" buttons.

→Click on Confirm



This block provides a detailed view of the "Add IP Access List Entry" dialog box. It features a close button (X) in the top right corner. The title "Add IP Access List Entry" is at the top. Below the title is the same descriptive text as in the screenshot: "Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more.](#)". Below this is the "ALLOW ACCESS FROM ANYWHERE" button. The "Access List Entry:" field now contains the value "0.0.0.0/0". The "Comment:" field remains empty with its placeholder text. At the bottom, the toggle switch is now turned on (indicated by a filled circle), and the text "This entry is temporary and will be deleted in" is followed by the "6 hours" dropdown. The "Cancel" and "Confirm" buttons are at the bottom right.

## →Active

IP Address	Comment	Status	Actions
117.99.207.64/32		● Active	<button>⚙ EDIT</button> <button>🗑 DELETE</button>

✓ Current IP Address (49.205.106.101/32) added!  
Visit [Network Access](#) to modify your settings.

## →Drivers (select go to your Application)

mongodb+srv://<username>:<password>@nodetuts.1xigocw.mongodb.net  
/test?retryWrites=true&w=majority

### Example:

mongodb+srv://netninja:test1234@[nodetuts.1xigocw.mongodb.net/node-tuts?retryWrites=true&w=majority](https://nodetuts.1xigocw.mongodb.net/node-tuts?retryWrites=true&w=majority)

<username> → netninja (add new database user)

<password> → test1234

test → node-tuts (database name)

## server.js

```
const express = require('express');
const mongoose = require('mongoose');
const BrandName = require('./model');

const app = express();

app.use(express.json()) //middleware
```



```
mongoose.connect('mongodb+srv://netninja:test1234@nodetuts.1xi
gocw.mongodb.net/node-tuts?retryWrites=true&w=majority')
    .then(() => console.log('DB Connected ...'))
    .catch(err => console.log(err));

app.get('/getallbrands', async (req, res) =>{
    try{
        const allData = await BrandName.find();
        return res.json(allData);
    }
    catch(err){
        console.log(err.message);
    }
})

app.post('/addbrands', async (req, res) => {
    const {brandname} = req.body;

    try{
        const newData = new BrandName({brandname});
        await newData.save();
        return res.json(await BrandName.find())
    }
    catch(err){
        console.log(err.message);
    }
})

app.get('/getallbrands/:id', async (req, res) => {
    try{
        const Data = await BrandName.findById(req.params.id)
        return res.json(Data);
    }
})
```

```

        catch(err) {
            console.log(err.message);
        }
    })

app.delete('/deletebrand/:id', async (req, res) =>{
    try{
        await BrandName.findByIdAndDelete(req.params.id);
        return res.json(await BrandName.find())
    }
    catch(err) {
        console.log(err.message);
    }
})

app.listen(3000, () => console.log('Server Running ....'))

```

## model.js

```

const mongoose = require('mongoose');

const BrandName = mongoose.Schema({
    brandname : {
        type : String,
        required : true,
    },
    date : {
        type : Date,
        default : Date.now
    }
})

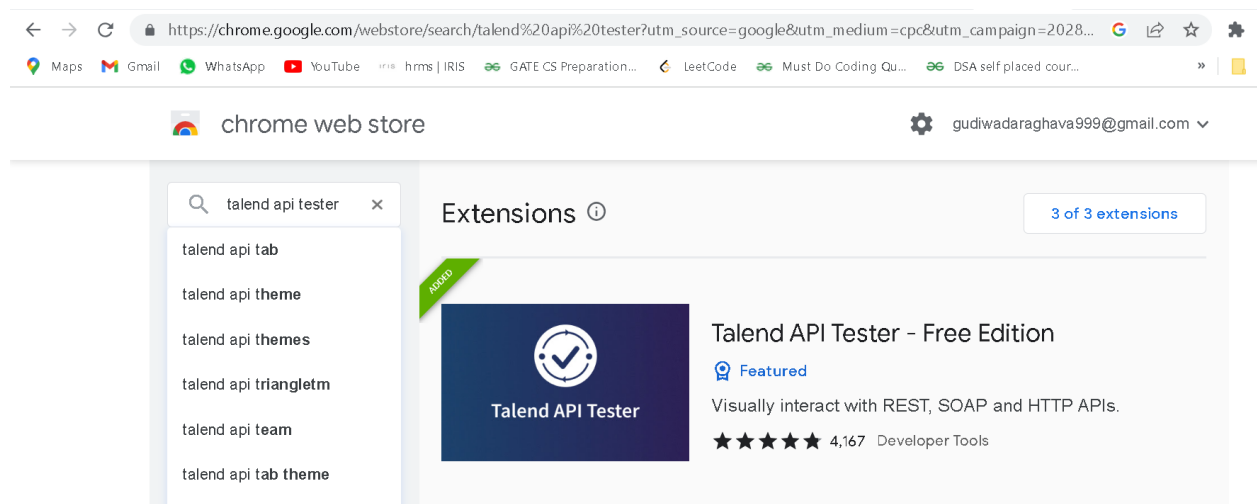
```

```
module.exports = mongoose.model('brandname', BrandName)
```

→post request we need to add from browser

→For that Chrome Extension “**Talend API Tester**”

[https://chrome.google.com/webstore/search/talend%20api%20tester?utm\\_source=google&utm\\_medium=cpc&utm\\_campaign=20282050520&utm\\_content=151103784475&utm\\_term=chrome%20web%20store&gclid=CjwKCAjwwb6lBhBJEiwAbuVUSswWzE5jdUPUv\\_rM9oe0LHNwQWKtGbfCkWnoYkVZvIMwoEzWvOcRIRoCYdgQAvD\\_BwE](https://chrome.google.com/webstore/search/talend%20api%20tester?utm_source=google&utm_medium=cpc&utm_campaign=20282050520&utm_content=151103784475&utm_term=chrome%20web%20store&gclid=CjwKCAjwwb6lBhBJEiwAbuVUSswWzE5jdUPUv_rM9oe0LHNwQWKtGbfCkWnoYkVZvIMwoEzWvOcRIRoCYdgQAvD_BwE)

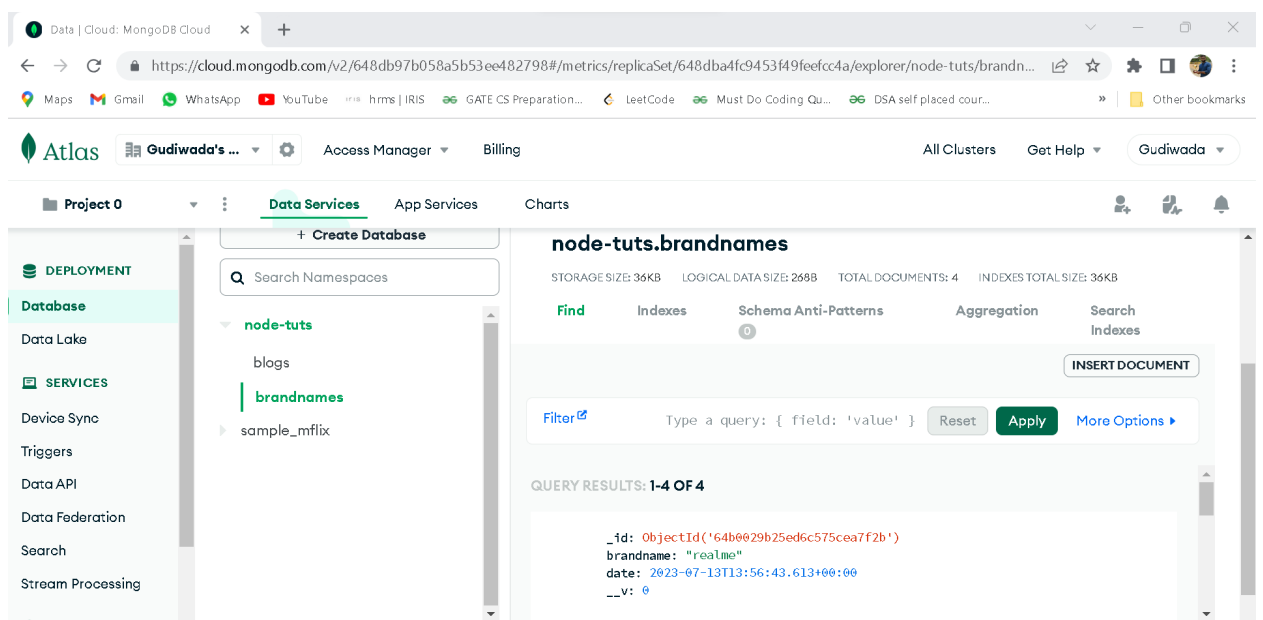


O/P:

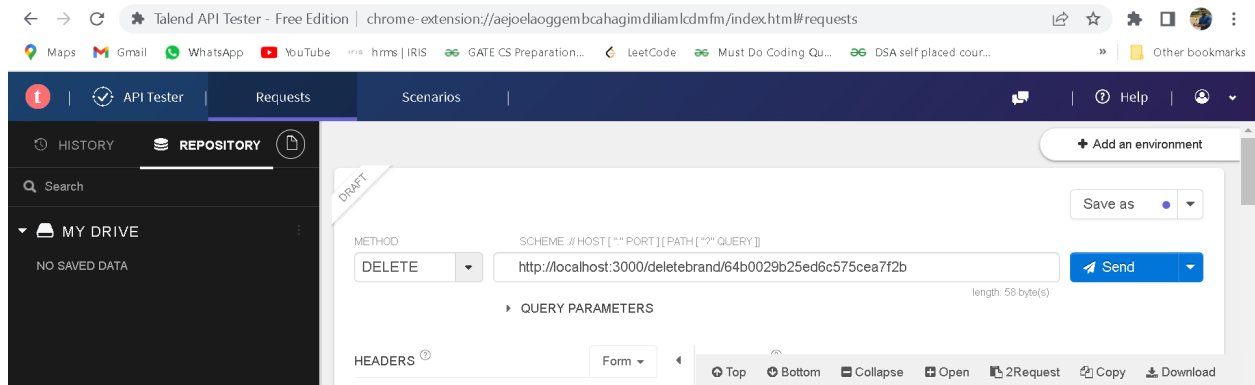
```
[{"_id": "64b0029b25ed6c575cea7f2b", "brandname": "realme", "date": "2023-07-13T13:56:43.613Z", "__v": 0}, {"_id": "64b002ed25ed6c575cea7f2e", "brandname": "samsung", "date": "2023-07-13T13:58:05.014Z", "__v": 0}, {"_id": "64b0030925ed6c575cea7f31", "brandname": "Nokia", "date": "2023-07-13T13:58:33.218Z", "__v": 0}, {"_id": "64b0034a25ed6c575cea7f36", "brandname": "iPhone", "date": "2023-07-13T13:59:38.231Z", "__v": 0}]
```

```
{"_id": "64b0029b25ed6c575cea7f2b", "brandname": "realme", "date": "2023-07-13T13:56:43.613Z", "__v": 0}
```

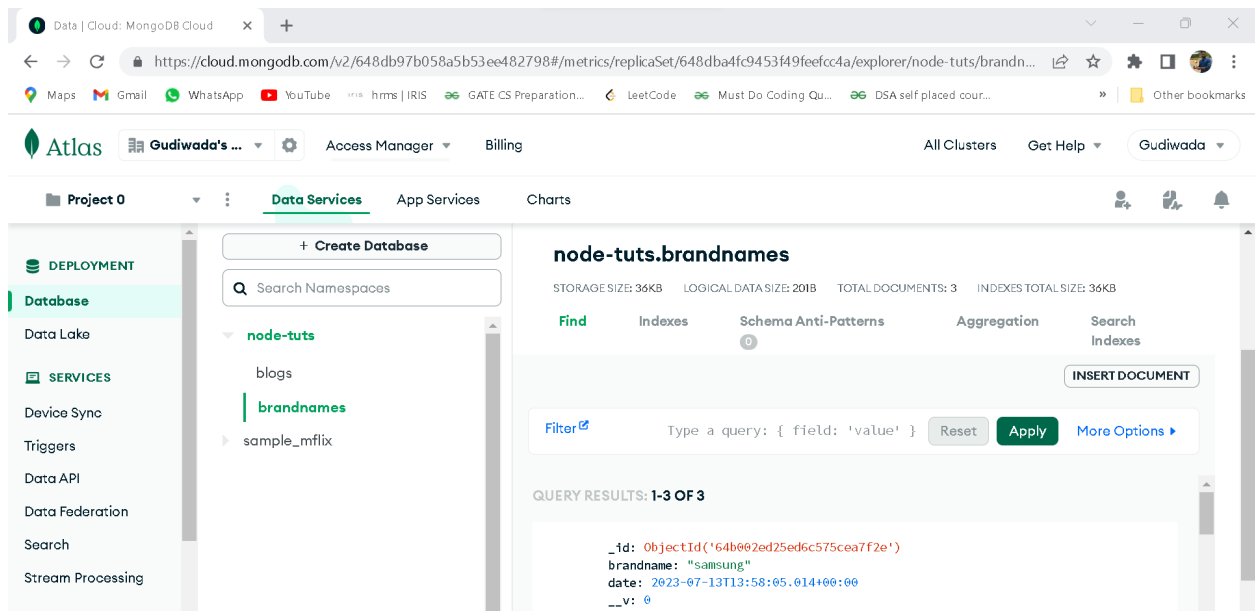
→All Post requests are merged in Database(MongoDB)



## →Deleted “realme”



## →After Deletion we have 3 in Database



→ Post <http://localhost:3000/addbrands>

HEADERS <sup>?</sup> pretty ▾

X-Powered-By: Express  
Content-Type: application/json; charset=utf-8  
Content-Leng... 99 bytes  
ETag: W/"63-u6jZZ+82XdFavFmVLdxTbH0DdC  
c"  
Date: Thu, 13 Jul 2023 13:56:43 GMT -1  
s  
Connection: keep-alive  
Keep-Alive: timeout=5

BODY <sup>?</sup> pretty ▾

```
[
  {
    _id: "64b0029b25ed6c575cea7f2b",
    brandname: "realme",
    date: "2023-07-13T13:56:43.613Z",
    __v: 0
  }
]
```

Top Bottom Collapse Open 2Request Copy Download

Length: 99 bytes

Node.js - Google Docs x Data | Cloud: MongoDB Cloud x localhost:3000/getallbrands/64afdef96e27a7d913328b48 x +

https://cloud.mongodb.com/v2/648db97b058a5b53ee482798#/metrics/replicaSet/648dba4fc9453f49feefcc4a/explorer/node-tuts/brandn... ☆ ⚙️ 📄 🌐

Maps Gmail WhatsApp YouTube hms | IRIS GATE CS Preparation... LeetCode Must Do Coding Qu... DSA self placed cour... » Other bookmarks

Atlas Gudiwada's ... Access Manager Billing All Clusters Get Help Gudiwada ▾

Project 0 Data Services App Services Charts

DEPLOYMENT Database Data Lake SERVICES Device Sync Triggers Data API Data Federation Search Stream Processing SECURITY

+ Create Database

Search Namespaces

node-tuts

- blogs
- brandnames
- sample\_mflix

node-tuts.brandnames

STORAGE SIZE: 20KB LOGICAL DATA SIZE: 45B TOTAL DOCUMENTS: 1 INDEXES TOTAL SIZE: 20KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

INSERT DOCUMENT

Filter Type a query: { field: 'value' } Reset Apply More Options ▸

QUERY RESULTS: 1-3 OF 3

```
{
  "_id": ObjectId('64afdef96e27a7d913328b48'),
  "brandname": "Samsung"
}
```

- **ExpressJs is needed to handle with API's**
- **Store the data in database we need MangoDB**

What is ExpressJS?

It's an back end web  
application **framework**  
for Node.js

It is designed for  
building **web**  
**applications** and **APIs**

<https://expressjs.com/> (Official Website of Express Js)

<https://www.npmjs.com/> (username :raghava45 pass: r.....\$123)

<https://www.npmjs.com/package/express>



→Command to install

>>>npm install express

req.params.id is string →We are getting data from url is string

item.id →is number/integer

item.id.toString →String

server.js

```
const express = require('express');
const app = express();

app.use(express.json())

app.get('/', function (req, res) {
    res.send('<h1>Hello World<h1>')
})

const products = [
    {
        id: 1,
        name: "iphone"
    },
    {
        id: 2,
        name: "mi"
    },
    {
        id: 3,
        name: "realme"
    }
]
```

```

app.get('/products', (req, res) =>{
    res.json(products);
})

app.get('/products/:id', (req, res) =>{
    const newData = products.filter(item => item.id.toString()
=== req.params.id )
    return res.json(newData);
})

//Post & put both similar
//put is updating data
app.post('/addproducts', (req, res) =>{
    const {id, name} = req.body;
    console.log(id, name);
    return res.send("data Stored !!")
})

app.listen(5000, () => console.log("server running..."))

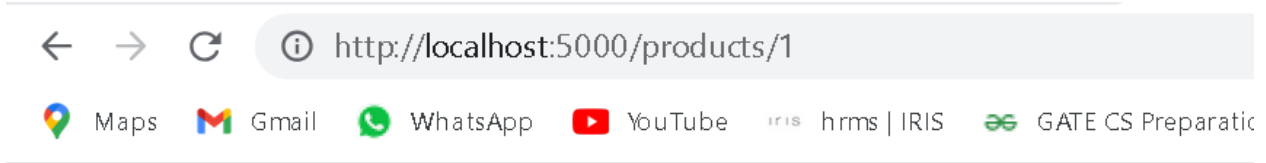
```

**O/P:**



The screenshot shows a web browser window with the address bar displaying `http://localhost:5000/products`. Below the address bar, there is a row of icons for various services: Maps, Gmail, WhatsApp, YouTube, IRIS, hrms | IRIS, GATE CS Preparation..., LeetCode, and Must Do Coding. The main content area of the browser displays a JSON array of three objects, each representing a product with an id and a name.

```
[{"id":1,"name":"iphone"}, {"id":2,"name":"mi"}, {"id":3,"name":"realme"}]
```



```
[{"id":1,"name":"iphone"}]
```

→By Using Post Method Sending the data

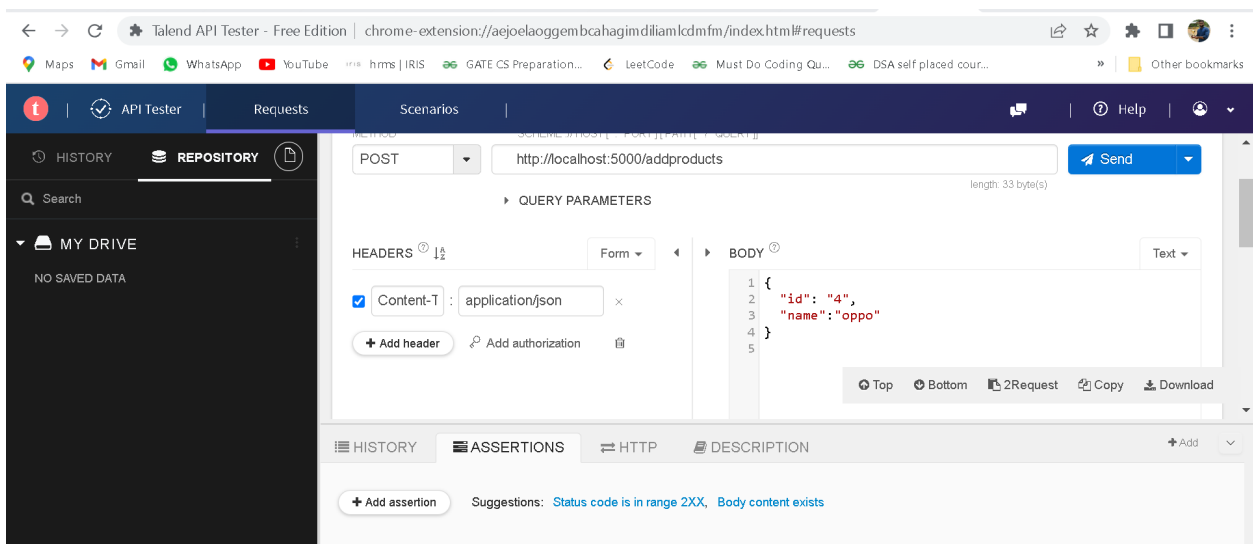
[nodemon] starting `node server.js`

server running...

4 oppo

5 Nokia

6 BlackBerry



Talend API Tester - Free Editionchrome-extension://aejoelaoggembcahagimdiliamlcdmfm/index.htm#requests

MapsGmailWhatsAppYouTubehms | IRISGATE CS Preparation...LeetCodeMust Do Coding Qu...DSA self placed cour...Other bookmarks

API Tester

Requests

Scenarios

Help

HISTORYREPOSITORY

Search

MY DRIVE

NO SAVED DATA

200 OK

HEADERS

pretty

X-Powered-By: Express  
Content-Type: text/html; charset=utf-8  
Content-Leng... 14 bytes  
ETag: W/"e-u1UM0oyvEWB7ZkxsATbKndDU8b8"  
Date: Thu, 13 Jul 2023 13:33:41 GMT  
Connection: keep-alive  
Keep-Alive: timeout=5

BODY

pretty

data Stored !!

length: 14 bytes

copy

TopBottom2RequestCopyDownload

HISTORYASSERTIONSHTTPDESCRIPTION

Add