

## Mongodb.js

```
const {MongoClient}=require('mongodb');

const url="mongodb://localhost:27017"

const database='student';

const client=new MongoClient(url);


const dbConnect=async()=>{

    const result=await client.connect();

    const db=await result.db(database);

    return db.collection('profile');

}

module.exports=dbConnect;
```

## index.js

```
const dbConnect=require('./mongodb')

const express=require('express');

const { response } = require('express');

const app=express(); app.use(express.json())


//get API

app.get('/getData',async(req,res)=>{
```

```
    let result=await dbConnect();

    result=await result.find().toArray();

    res.send(result);

  })

  //post API

  app.post('/insertData',async(req,res)=>{

    let result=await dbConnect();

    result=await result.insertOne(req.body);

    res.send("Data Inserted Successfully")

  })

  // Put API

  app.put('/updateData/:name',async(req,res)=>{

    let result=await dbConnect();

    result=await result.updateOne({name:req.params.name},{ $set:req.body});

    res.send("Data Updated Successfully")

  })

  //Delete API

  app.delete('/deleteData/:name',async(req,res)=>{

    let result=await dbConnect();

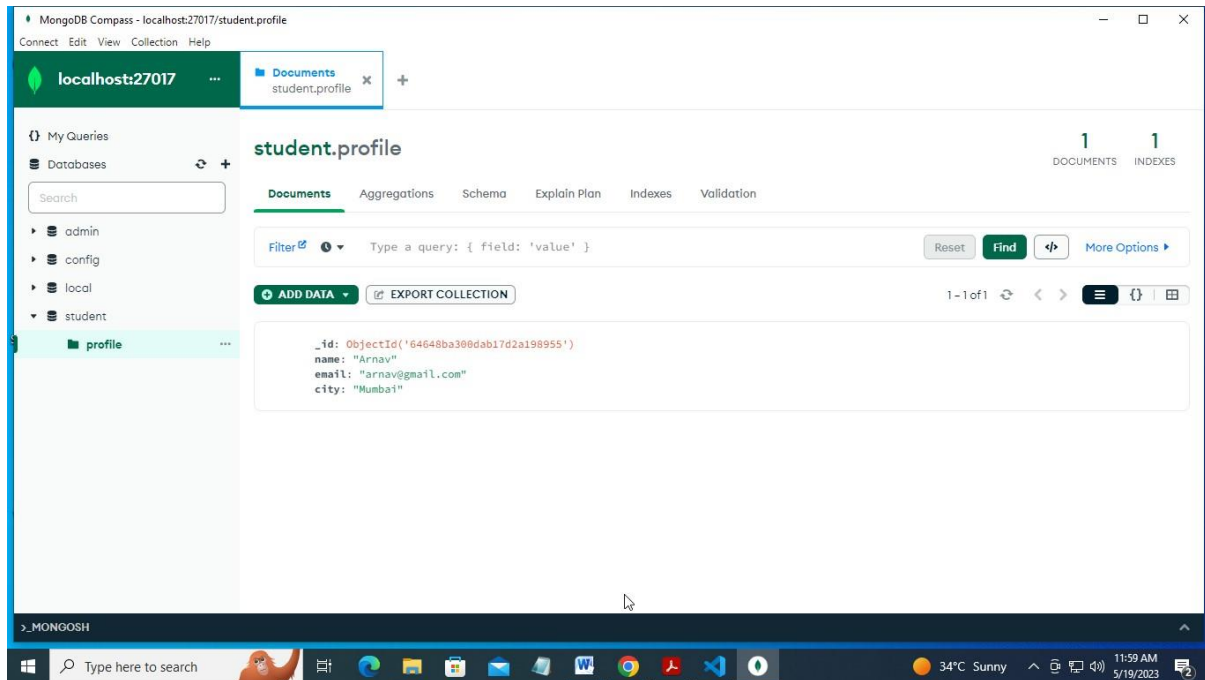
    result=await result.deleteOne({name:req.params.name})

    res.send("Data Deleted Successfully");

  })

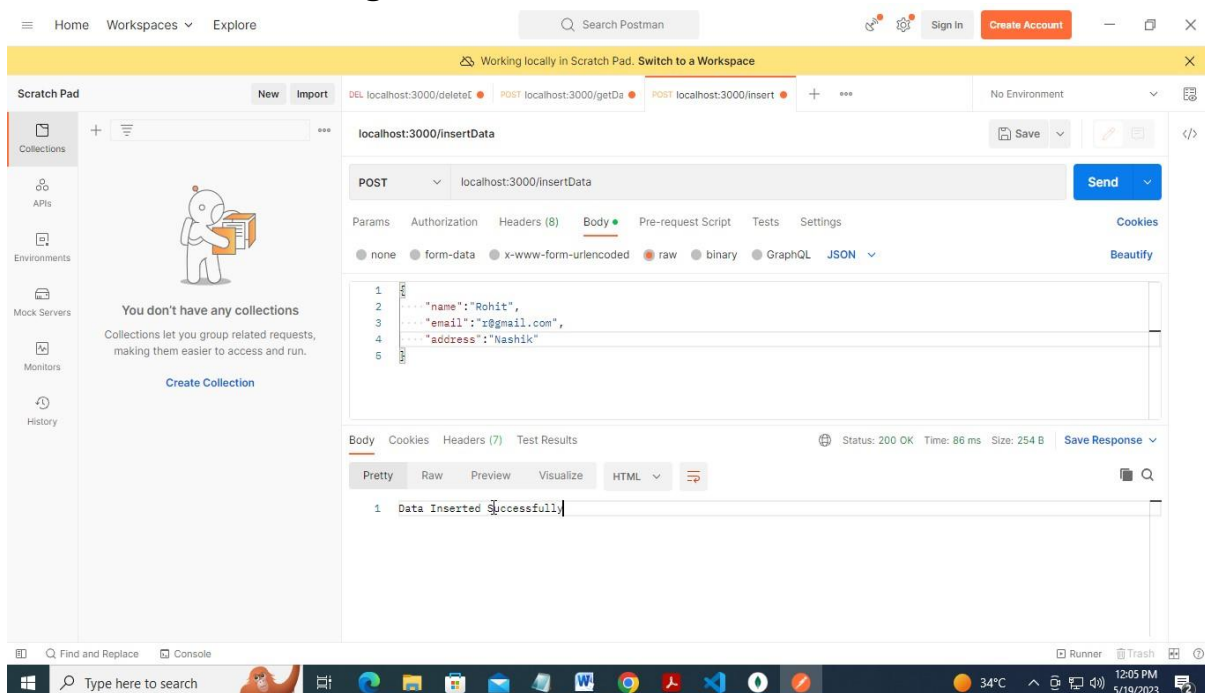
  app.listen(3000);
```

# MongoDB Compass



// Output :

## Create a new Todo using POST /insertData API



## Retrieving all Todos using GET /getData API

The screenshot shows the Postman application interface. The top bar includes navigation links (Home, Workspaces, Explore), a search bar, and user options (Sign In, Create Account). The main workspace is titled "Working locally in Scratch Pad. Switch to a Workspace". On the left, the "Scratch Pad" sidebar shows a "Collections" section with a message: "You don't have any collections. Collections let you group related requests, making them easier to access and run. Create Collection". The main panel displays a GET request to "localhost:3000/getData/". The request is configured with "Headers (6)" and "Body" tabs. The "Body" tab is selected, showing a JSON response in "Pretty" format:

```
1 {
2   "id": "64648ba308dab17d2a198956",
3   "name": "Arnav",
4   "email": "arnav@gmail.com",
5   "city": "Mumbai"
6 }
7
8 {
9   "id": "6467196d837806a703913aeb",
10  "name": "Rohit",
11  "email": "r@gmail.com",
12  "address": "Nashik"
13 }
14 }
```

The status bar at the bottom indicates "Status: 200 OK", "Time: 9 ms", and "Size: 420 B". The Windows taskbar at the very bottom shows the system clock as 12:09 PM on 5/19/2023.

## Updating a Todo using PUT localhost:3000/updateData/Rohit API

The screenshot shows the Postman application interface for a PUT request. The top bar is identical to the previous screenshot. The main workspace is titled "Working locally in Scratch Pad. Switch to a Workspace". The "Scratch Pad" sidebar is the same. The main panel displays a PUT request to "localhost:3000/updateData/Rohit". The request is configured with "Headers (8)" and "Body" tabs. The "Body" tab is selected, showing a JSON body in "Pretty" format:

```
1 {
2   "name": "Rohit",
3   "email": "r@gmail.com",
4   "address": "Pune"
5 }
```

The status bar at the bottom indicates "Status: 200 OK", "Time: 87 ms", and "Size: 253 B". Below the response body, the text "Data Updated Successfully" is visible. The Windows taskbar at the very bottom shows the system clock as 12:11 PM on 5/19/2023.

## Deleting a Todo using DELETE localhost:3000/deleteData/Rohit API

The screenshot displays the Postman interface with a DELETE request configured to `localhost:3000/deleteData/Rohit`. The request is successfully executed, returning a 200 OK status. The response body, shown in the 'Body' tab, contains the text `1 Data Deleted Successfully`.

**Request Details:**

- Method: DELETE
- URL: localhost:3000/deleteData/Rohit
- Headers (8): [Not visible]
- Body: [Not visible]
- Pre-request Script: [Not visible]
- Tests: [Not visible]
- Settings: [Not visible]

**Response Details:**

- Status: 200 OK
- Time: 16 ms
- Size: 253 B
- Save Response: [Not visible]

**Response Body (Pretty):**

```
1 Data Deleted Successfully
```