NODE JS WITH MONGOOSE CONNECTIVITY:-

Using Mongoose (Simpler & More Common)

mkdir my-node-server

cd my-node-server

npm install express

Install Mongoose:-

npm install mongoose

db.js file Code :-

```
const mongoose = require("mongoose");

async function connectDB() {
  try {
    await mongoose.connect("mongodb://localhost:27017/myrecord", {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    });
    console.log("✅Connected to MongoDB using Mongoose!");
  } catch (err) {
    console.error("❌ MongoDB connection error:", err);
  }
}

module.exports = connectDB;
```

Index.js file :-

```
const express = require("express");
const connectDB = require("./db");
const mongoose = require("mongoose");

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

connectDB();
```

```
// Example Schema
const UserSchema = new mongoose.Schema({
  name: String,
  email: String,
});

const User = mongoose.model("User", UserSchema);

app.get("/user-data", async (req, res) => {
  const users = await User.find();
  res.json(users);
});


app.post("/user-insert", async (req, res) => {
  try {
    const user = await User.create(req.body);
    res.status(201).json(user);
  } catch (err) {
    res.status(400).json({ error: err.message });
  }
});


app.listen(3000, () => console.log("🚀 Server running on port 3000"));
```

And in mongodb shell insert some

Records:-

```
> use myrecord
< switched to db myrecord
> db.users.updateOne(
    { name: "Alice" },
    { $set: { email: "alice@newdomain.com" } }
  )
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0
  }
> db.users.insertMany([
    { name: "Bob", email: "bob@example.com" },
    { name: "Charlie", email: "charlie@example.com" },
    { name: "Dana", email: "dana@example.com" }
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('69030a24986db08595799851'),
      '1': ObjectId('69030a24986db08595799852'),
      '2': ObjectId('69030a24986db08595799853')
    }
  }
myrecord>
```

Example Collection

Let's say your database is `mydatabase` and your collection is `users`.

To switch to that database:

```
use mydatabase
```

Insert One Document

```
db.users.insertOne({
  name: "Alice",
  email: "alice@example.com"
})
```

Insert Multiple Documents

```
db.users.insertMany([
  { name: "Bob", email: "bob@example.com" },
  { name: "Charlie", email: "charlie@example.com" },
  { name: "Dana", email: "dana@example.com" }
])
```

View the Inserted Documents

```
db.users.find().pretty()
```

Bonus: Update and Delete Examples

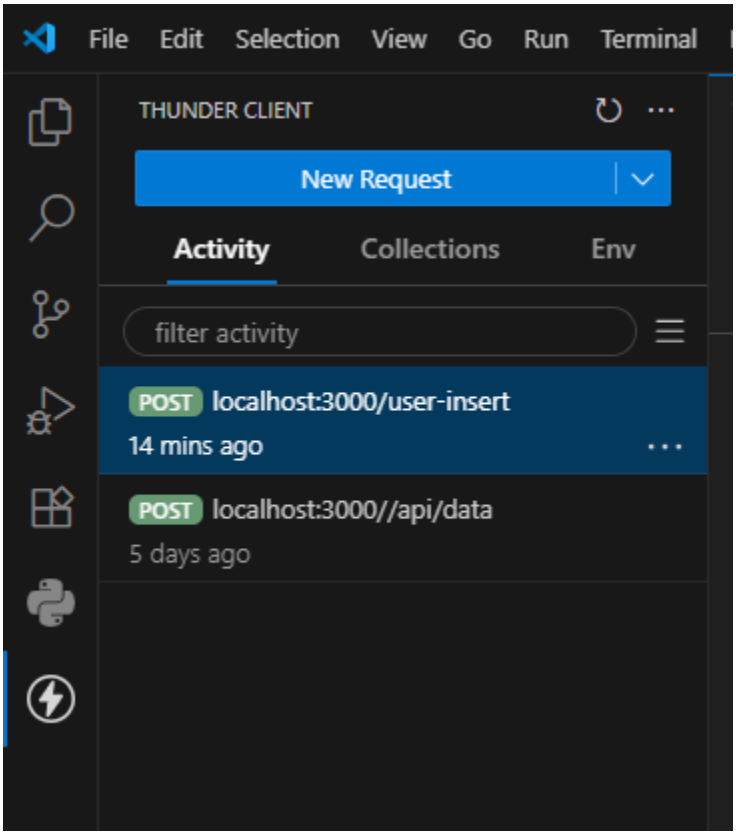
Update one:

```
db.users.updateOne(
  { name: "Alice" },
  { $set: { email: "alice@newdomain.com" } }
)
```

Delete one:

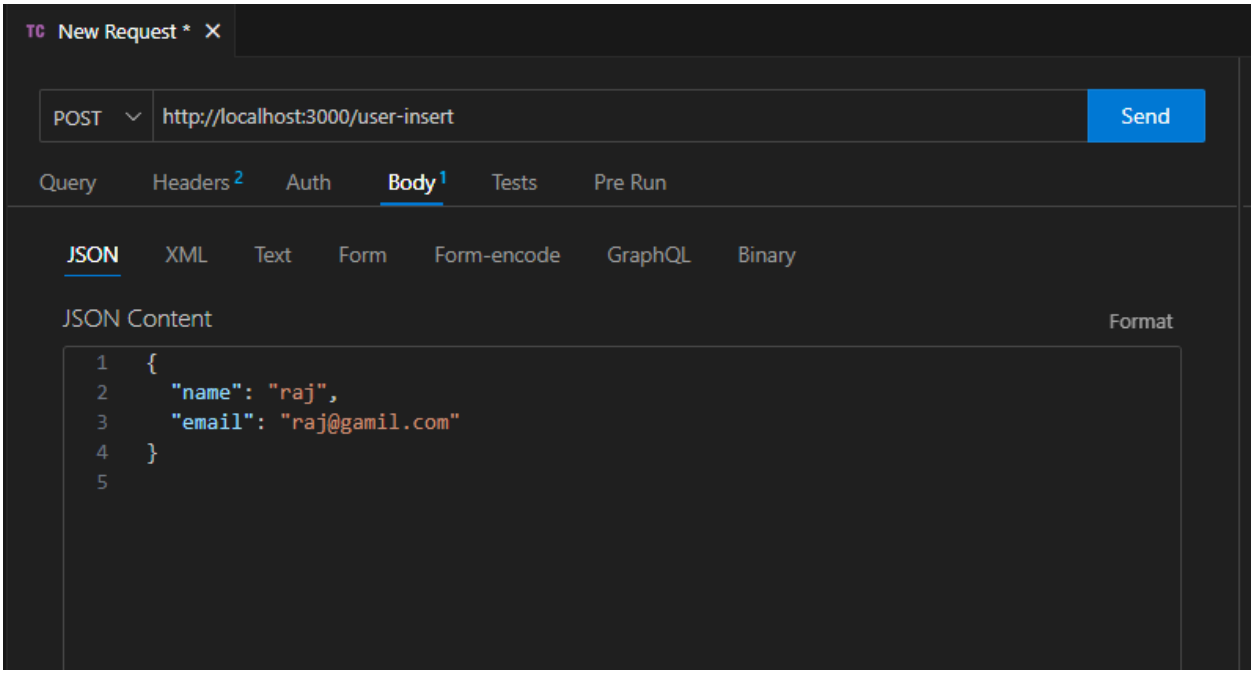
```
db.users.deleteOne({ name: "Bob" })
```

Now in vs code install Thunder Client and click on new request as shown below:-

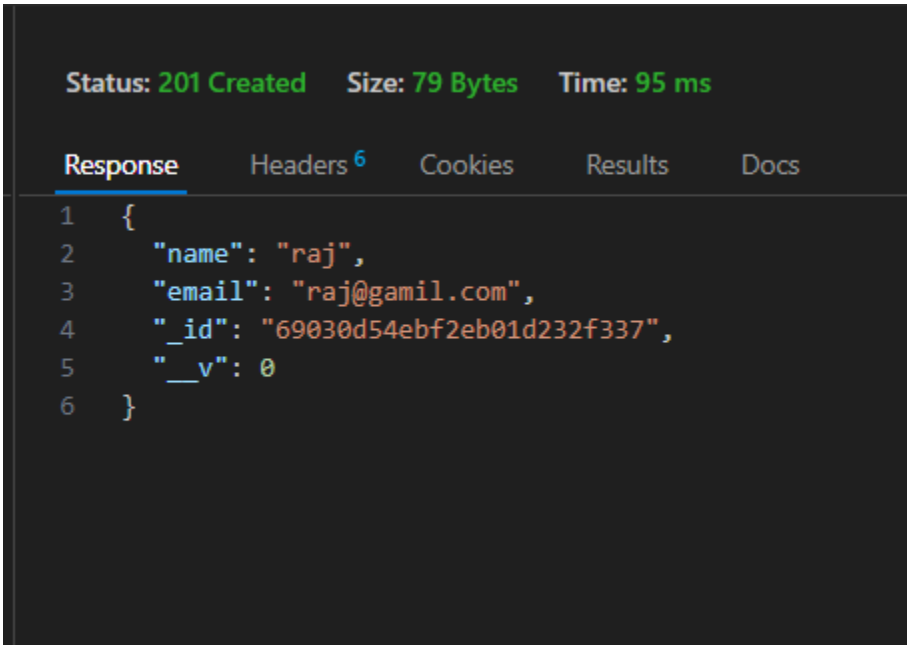


Then Paste

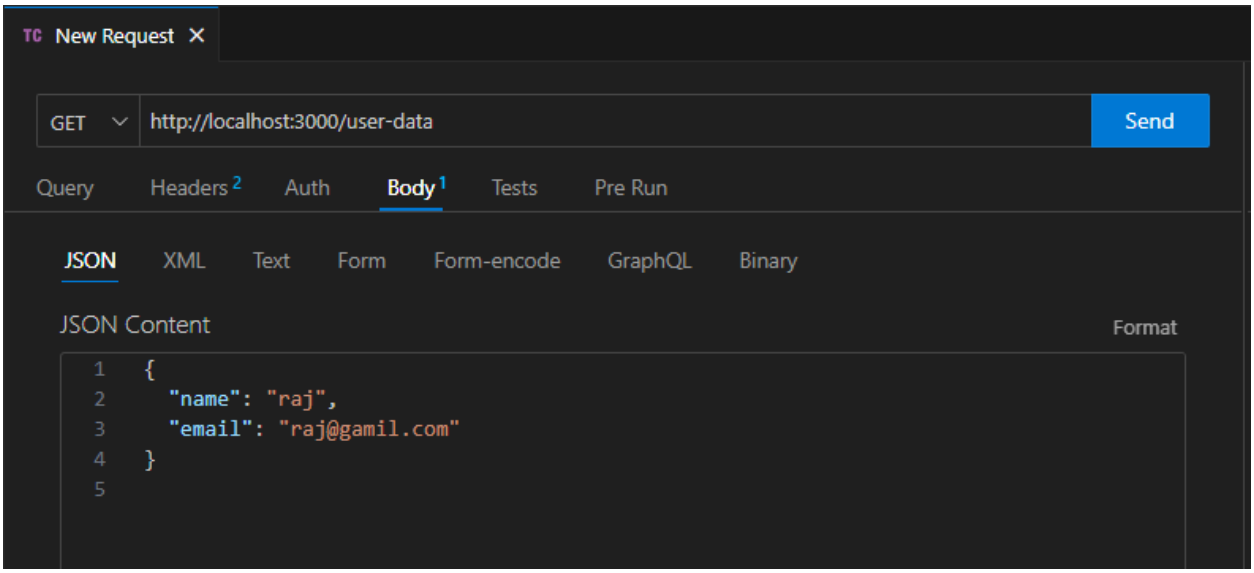
http://localhost:3000/user-insert



Click on send:-



http://localhost:3000/user-data


New Request
GET
http://localhost:3000/user-data
Send
Query
Headers 2
Auth
Body 1
Tests
Pre Run
JSON
XML
Text
Form
Form-encode
GraphQL
Binary
JSON Content
Format
{
"name": "raj",
"email": "raj@gamil.com"
}

Response:-


Status: 200 OK Size: 313 Bytes Time: 140 ms
Response Headers 6 Cookies Results Docs
[
{
"_id": "69030a24986db08595799851",
"name": "Bob",
"email": "bob@example.com"
},
{
"_id": "69030a24986db08595799852",
"name": "Charlie",
"email": "charlie@example.com"
},
{
"_id": "69030a24986db08595799853",
"name": "Dana",
"email": "dana@example.com"
},
{
"_id": "69030d54ebf2eb01d232f337",
"name": "raj",
"email": "raj@gamil.com",
"__v": 0
}
]