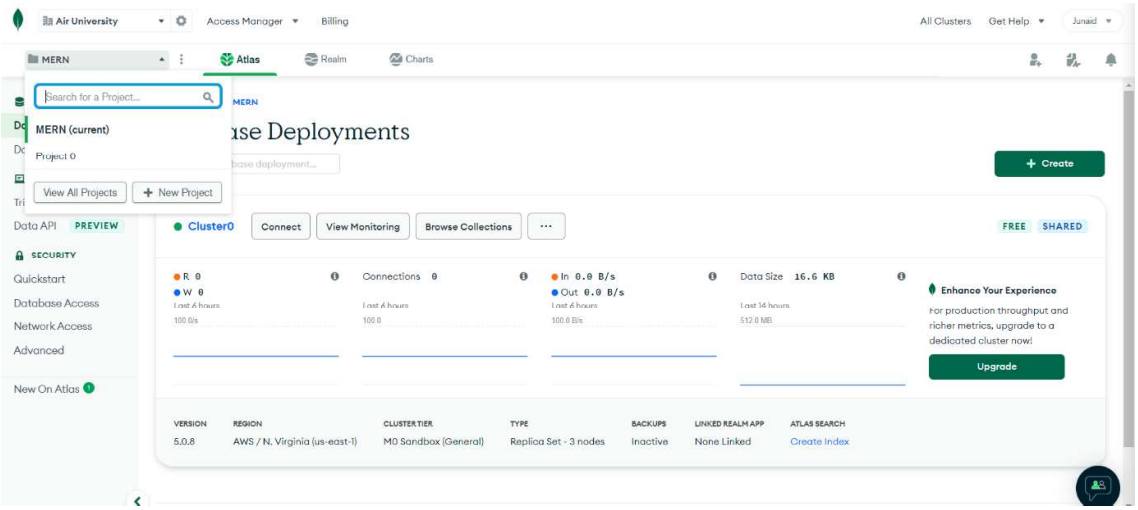
Statement Purpose:

To familiarize the students with

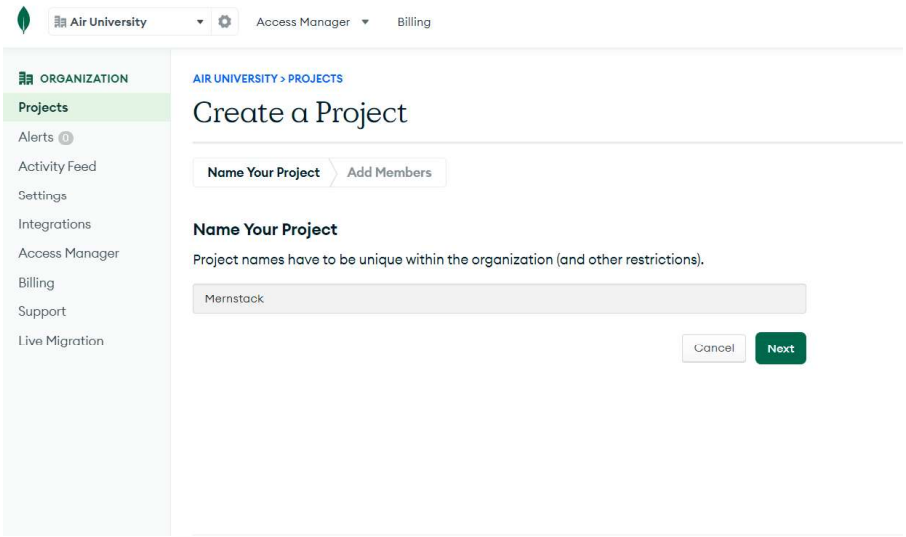
Learning Objectives

a) Connectivity with MongoDB
b) Data Find form MongoDB
c) Data Insert into MongoDB

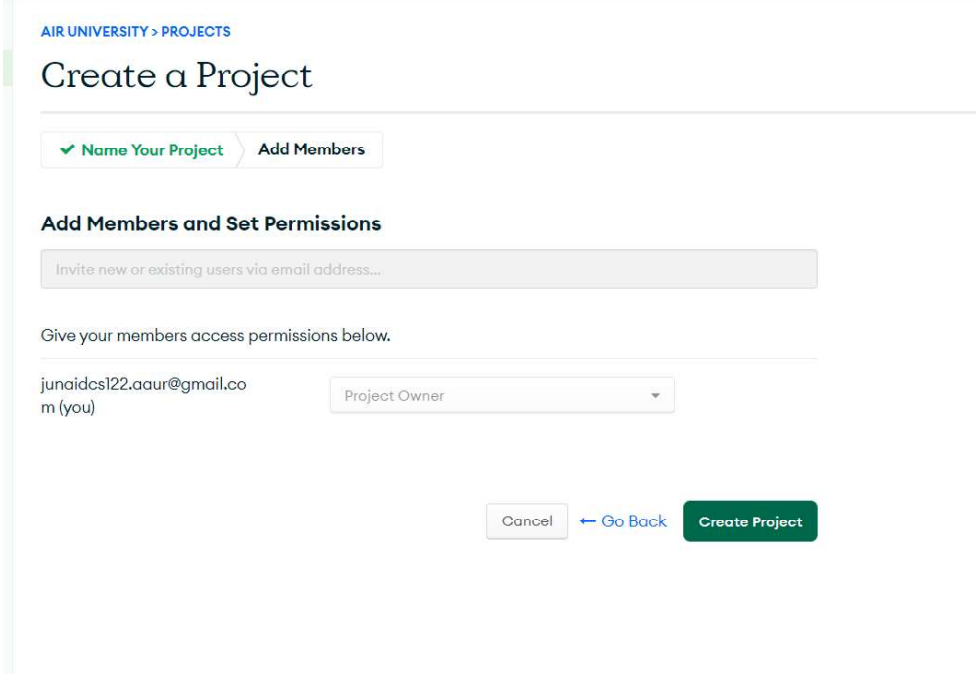
1. Download MongoDB from https://www.mongodb.com/try/download/community
2. Create an atlas account on https://cloud.mongodb.com/
3. After sign-in on new account, create new project



4. Write project name i.e **Mernstack** and click on **Next**



Click on “Create project”



Click on “Build a Database”

AIR UNIVERSITY > MERNSTACK

Database Deployments

Create a database

Choose your cloud provider, region, and specs.

Build a Database

Once your database is up and running, live migrate an existing MongoDB database into Atlas with our Live Migration Service.

Deploy a cloud database

Experience the best of MongoDB on AWS, Azure, and Google Cloud. Choose a deployment option to get started.

PREVIEW

Serverless

For serverless applications that aren't critical with variable traffic. Minimal configuration required.

- Pay only for the operations you run
- Resources scale seamlessly to meet your workload
- Always-on security and backups

Create

Starting at
$0.30/1M reads

ADVANCED

Dedicated

For production applications with sophisticated workload requirements. Advanced configuration controls.

- Network isolation and fine-grained access controls
- On-demand performance advice
- Multi-region and multi-cloud options available

Create

Starting at
$0.08/hr*
*estimated cost $56.94/month

FREE

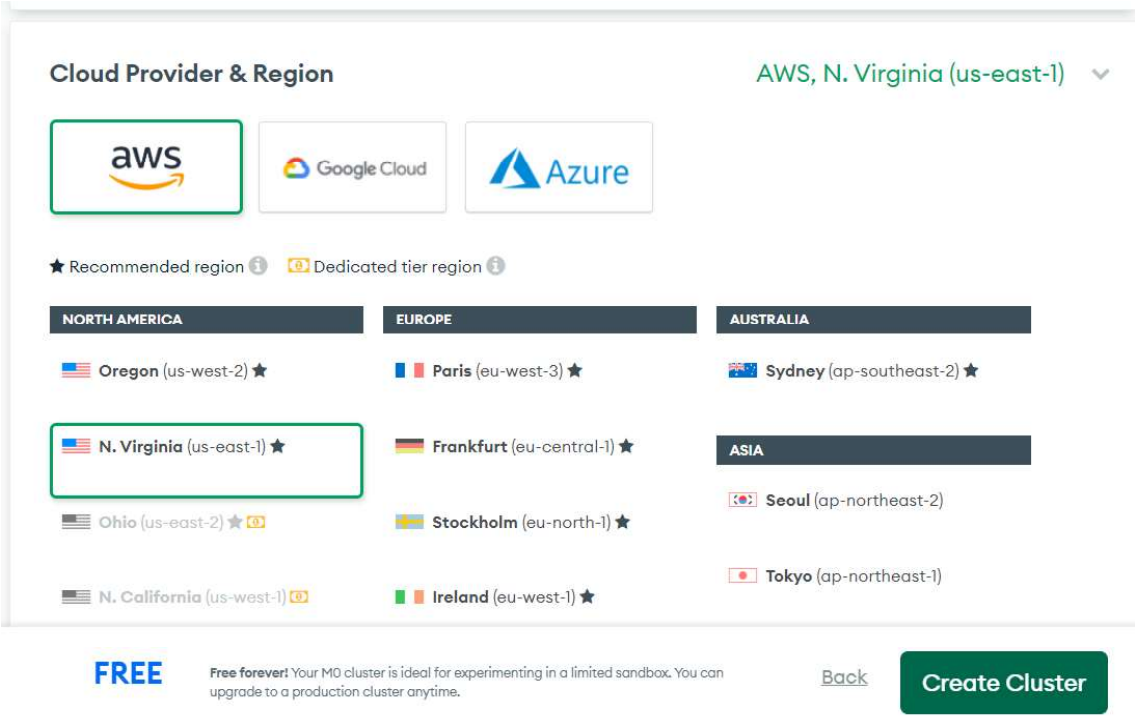
Shared

For learning and exploring MongoDB in a cloud environment. Basic configuration options.

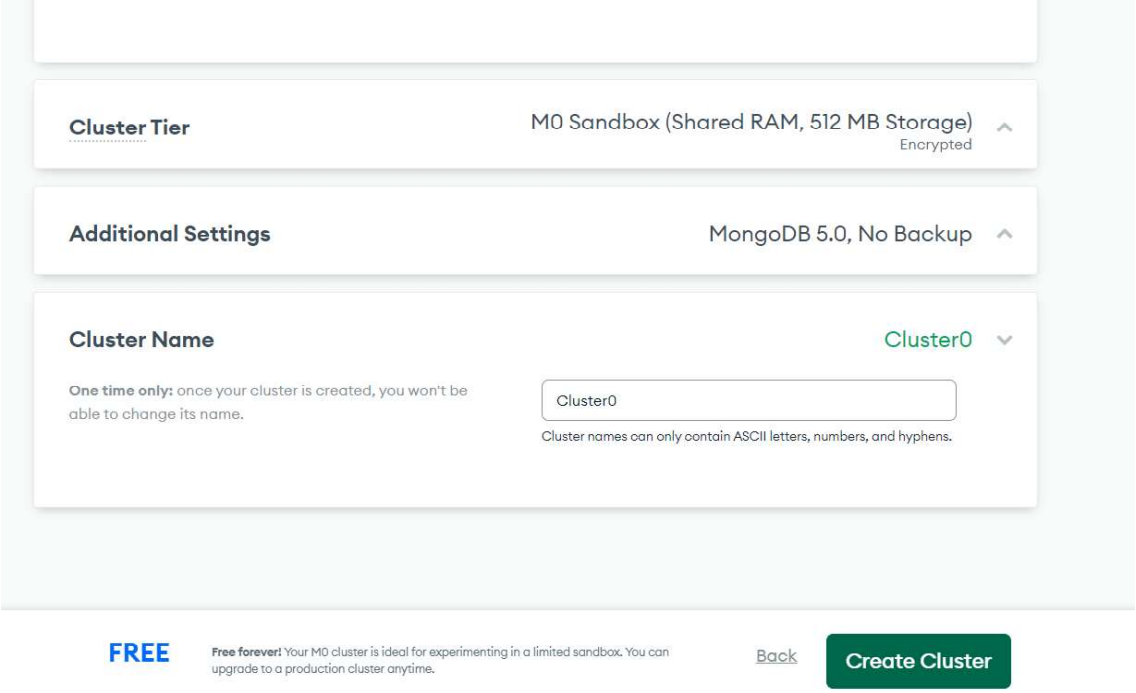
- No credit card required to start
- Explore with sample datasets
- Upgrade to dedicated clusters for full functionality

Create

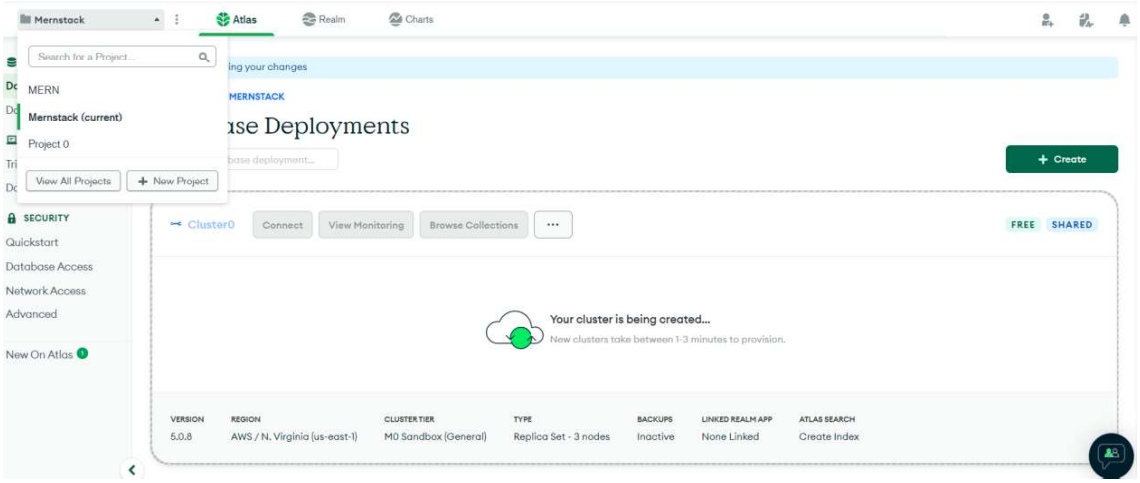
Starting at
FREE

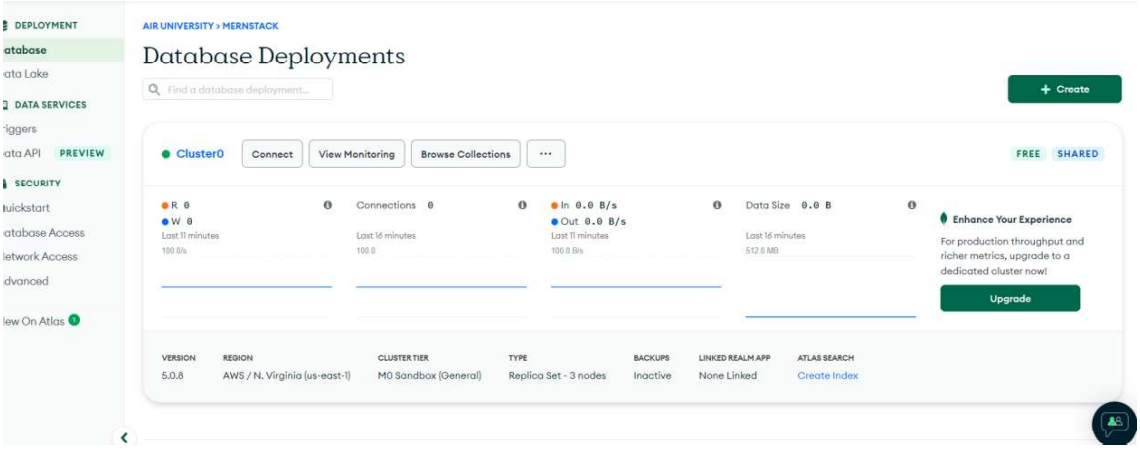


You can also change the cluster name



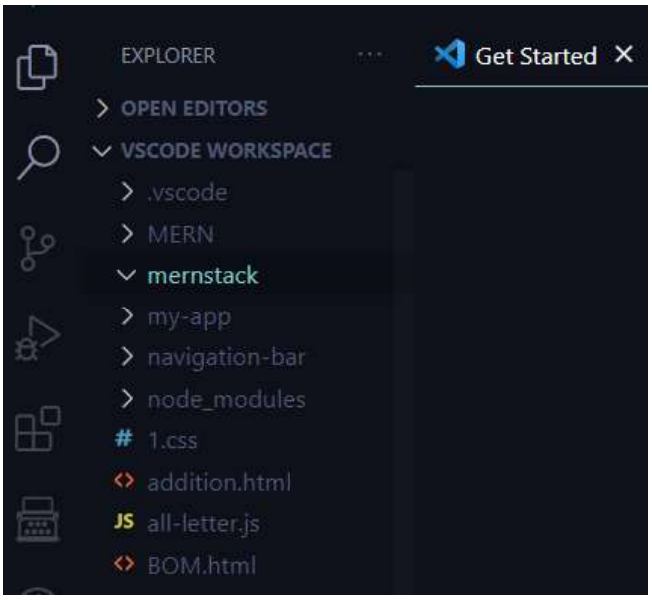
After Creating the cluster click on projects and it will take come minutes to create cluster



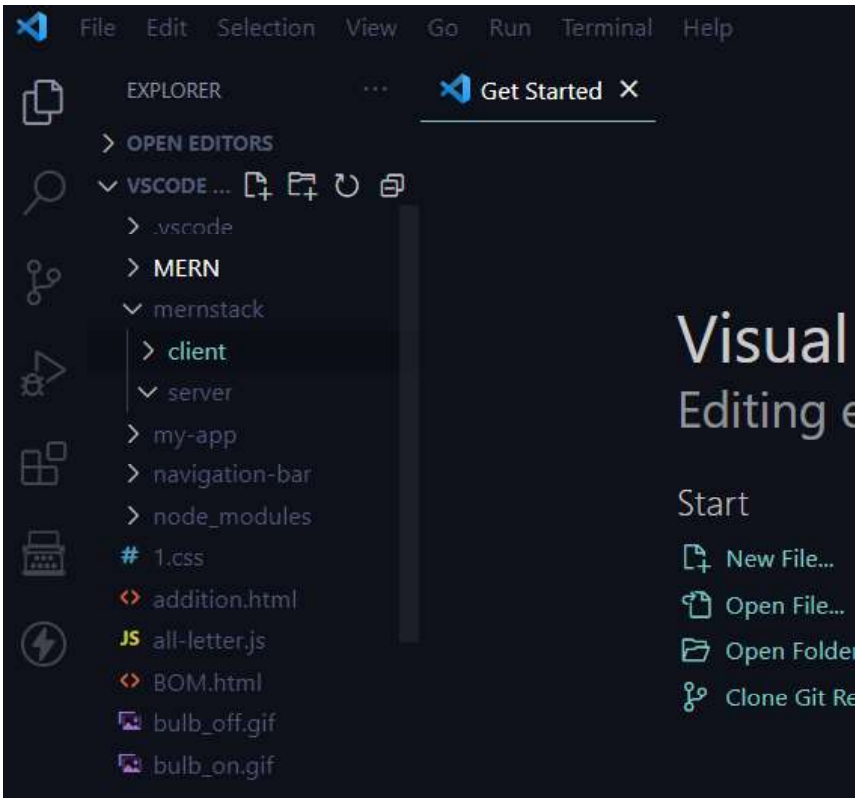


After creating a cluster

Create a folder in your workspace



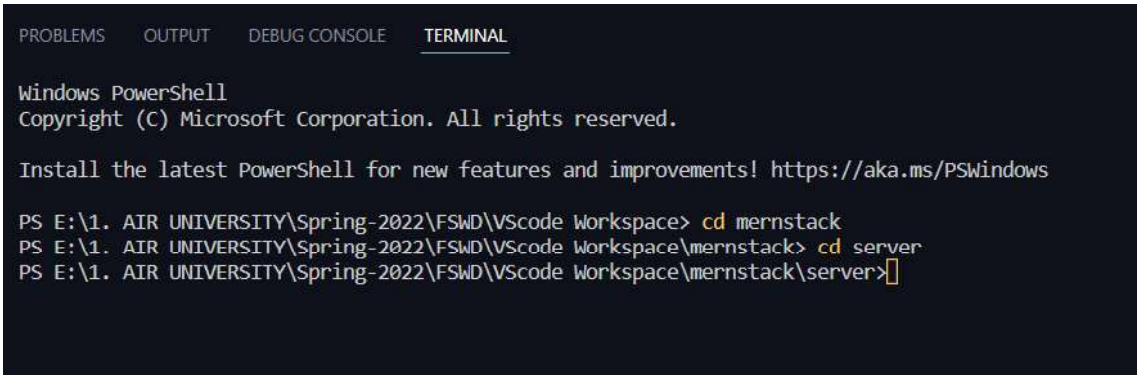
Create 2 folders folder in **mernstack** -> **server** and **client**



Open Terminal and write

cd mernstack

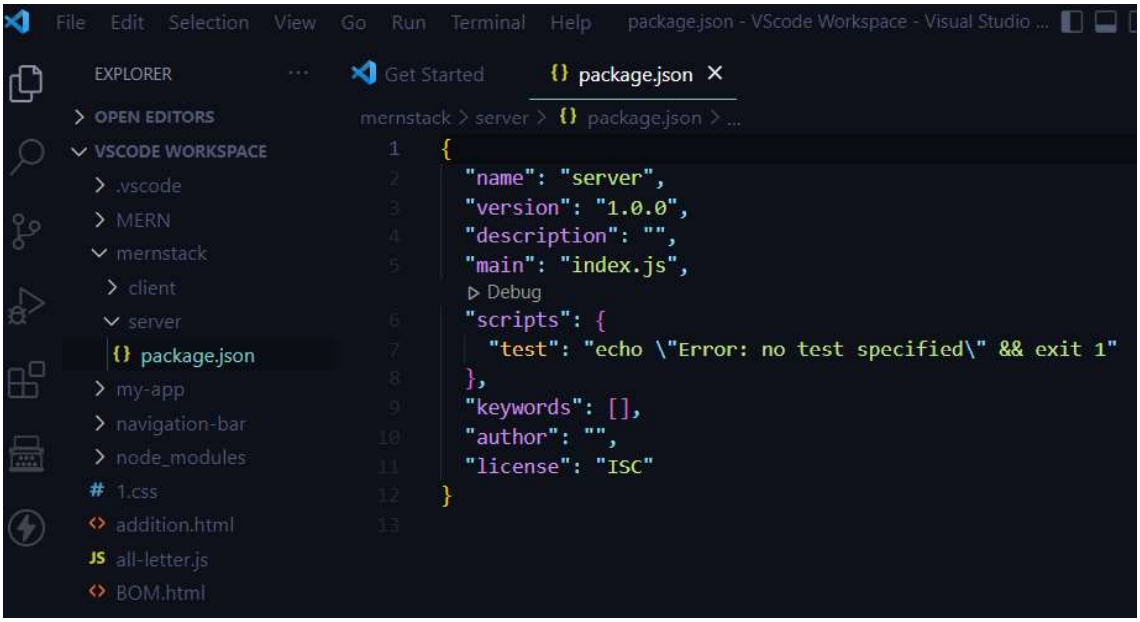
cd server



I want to initialize an express server and express application in this folder

npm init -y (It will create a package.json file)

```
"version": "1.0.0",
"description": "",
"main": "index.js",
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1"
},
"keywords": [],
"author": "",
"license": "ISC"

PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server>
```



Now installing different dependencies and package that we are installing that we want in our application

Install express

mongoose (to communicate with mongodb in express or using node(.js), we are using this library)

cors (Cross-Origin Resource Sharing (CORS) is an HTTP-header based mechanism that allows a server to indicate any origins (domain, scheme, or port) other than its own from which a browser should permit loading resources)

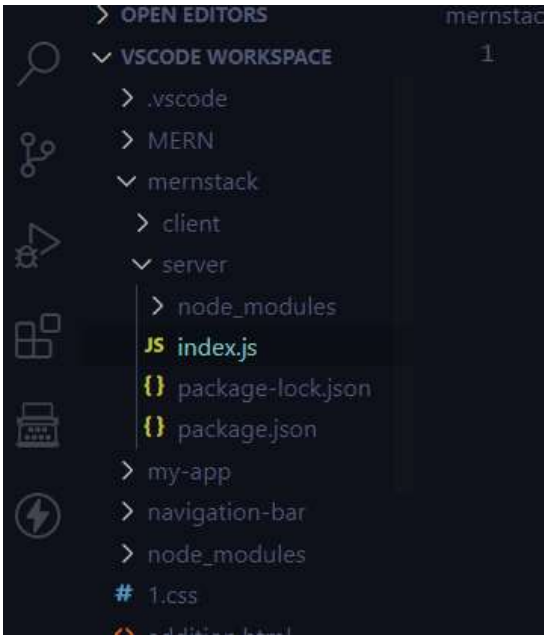
nodemon (nodemon is a tool that helps develop Node.js based applications by automatically restarting the node application when file changes in the directory are detected)

Write command

npm install express mongoose cors nodemon

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server>

PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> npm install express mongoose cors nodemon
[..................] | idealTree:server: sill idealTree buildDeps
```

Create a file “index.js” in server folder



Initially perform declaration in index.js file and run in terminal

node index.js

```
JS index.js 8 ×
mernstack > server > JS index.js > ...
const express = require('express'); //importing a library express by creating a variable code exp
const app = express(); // var app represent of all the express stuff we get from the library



app.listen(3002, () => {
    console.log("SERVER IS RUNNING");

}); //add port and add callback function which wil just run when server start running
```

```
PROBLEMS 18   OUTPUT   DEBUG CONSOLE   TERMINAL

PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> node index.js
SERVER IS RUNNING
```

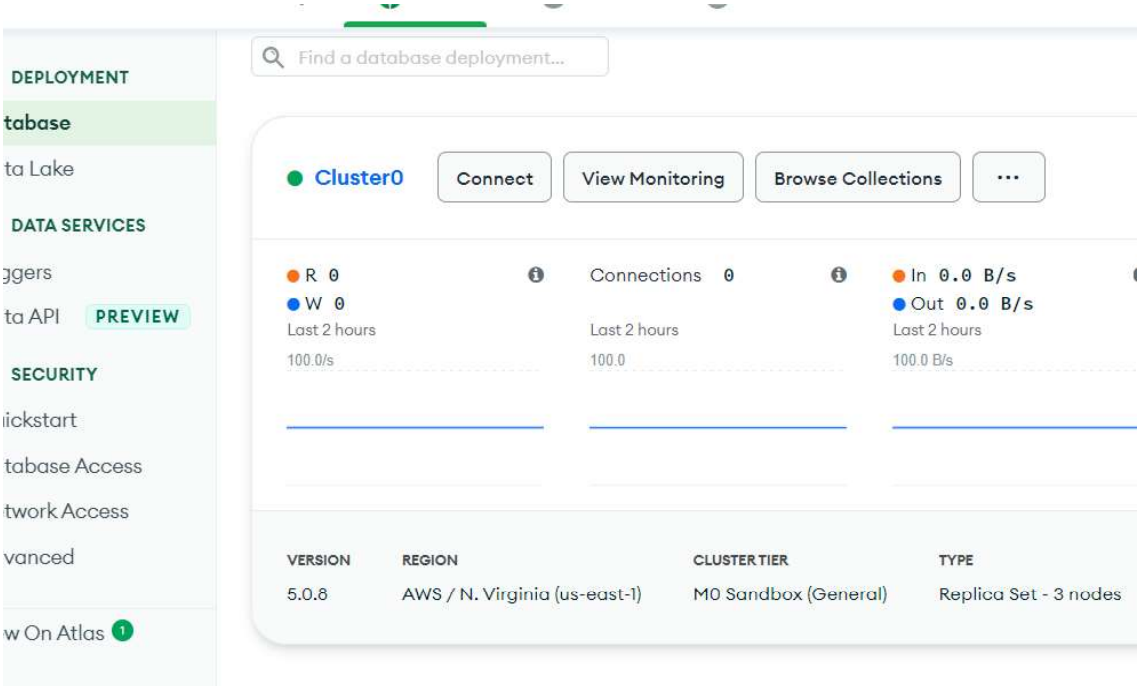
Then open package.json file and add

```
"start": "nodemon index.js"
//whenever we make any changes in file it will restart the server
```

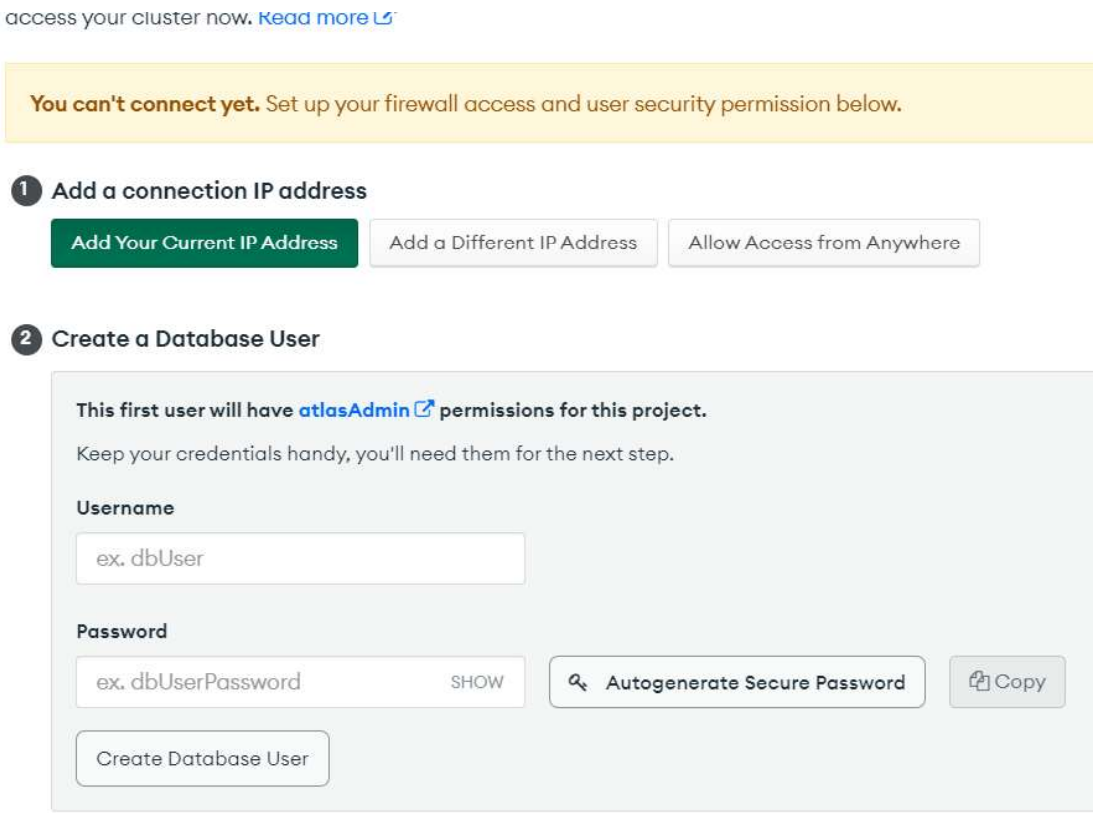
```
File  Edit  Selection  View  Go  Run  Terminal  Help        index.js - VScode Workspace - Visual Studio Code

JS index.js 8 ×   {} package.json
mernstack > server > JS index.js > ...
1   const express = require('express'); //importing a library express by creating a variable ca
2   const app = express(); // var app represent of all the express stuff we get from the librar
3
4
5
6   app.listen(3002, () => {
7       console.log("SERVER IS RUNNING");
8
9   }); //add port and add callback function which wil just run when server start running

PROBLEMS 18   OUTPUT   DEBUG CONSOLE   TERMINAL

npm ERR!     C:\Users\uaar_\AppData\Local\npm-cache\_logs\2022-05-16T05_24_06_735Z-debug.log
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> npm start

> server@1.0.0 start
> nodemon index.js

[nodemon] 2.0.16
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
SERVER IS RUNNING
```

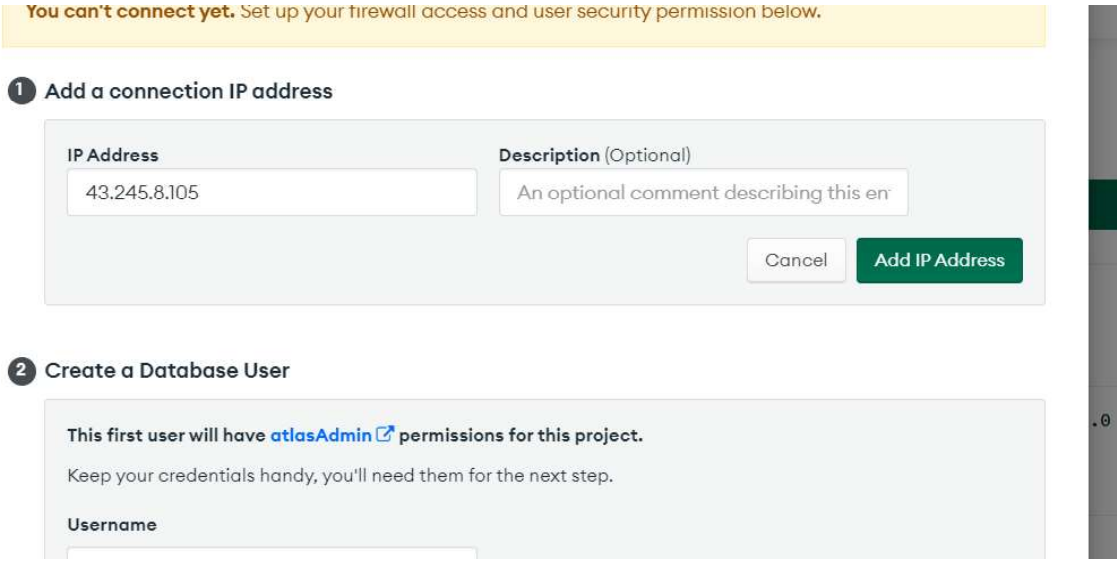
Start setting up MondoDB connection

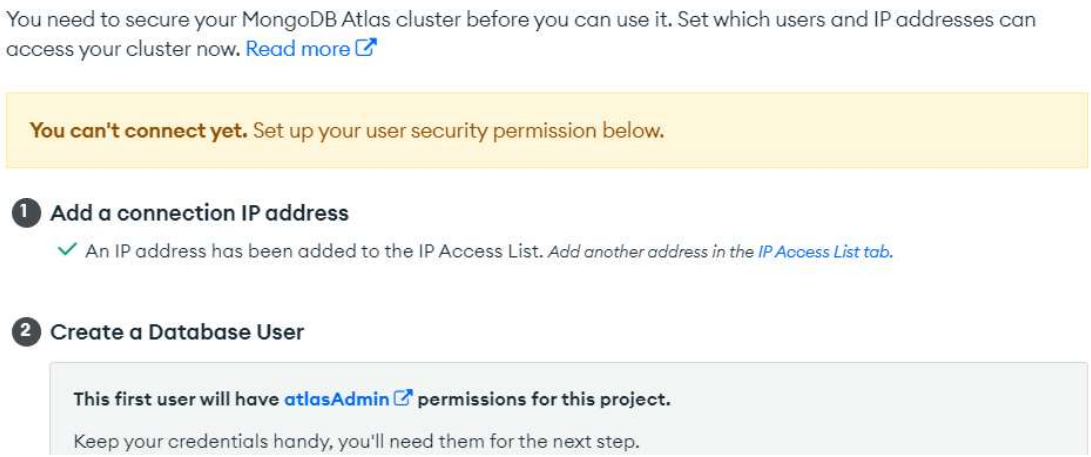
Click on connect



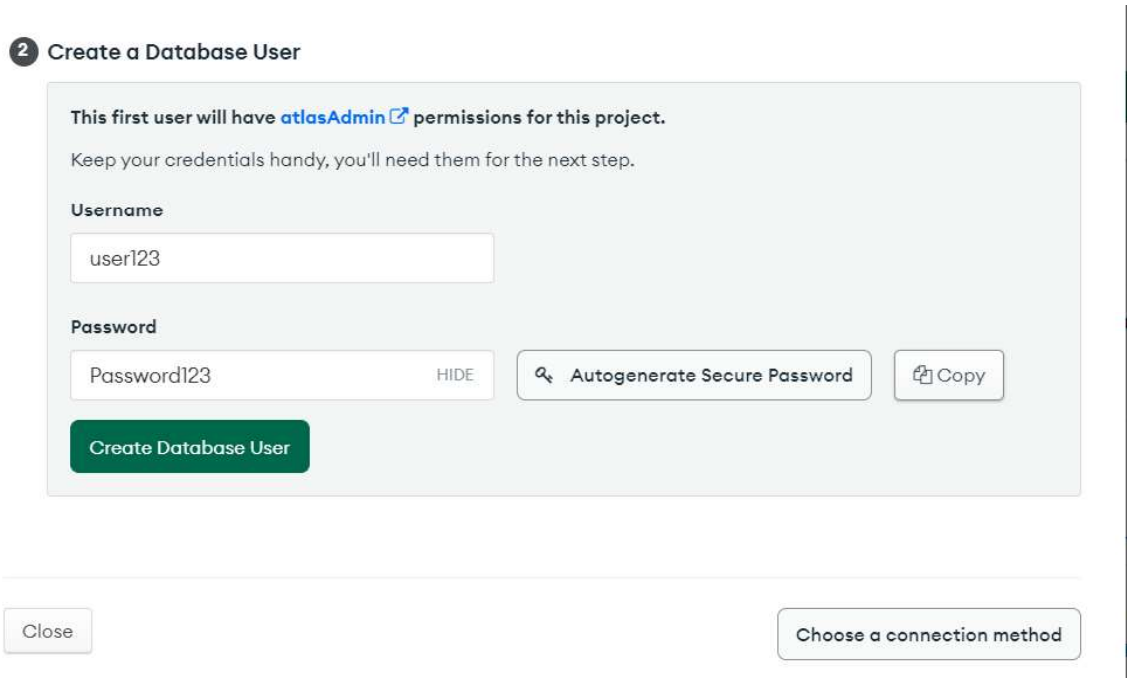
Click on “Add your IP address”



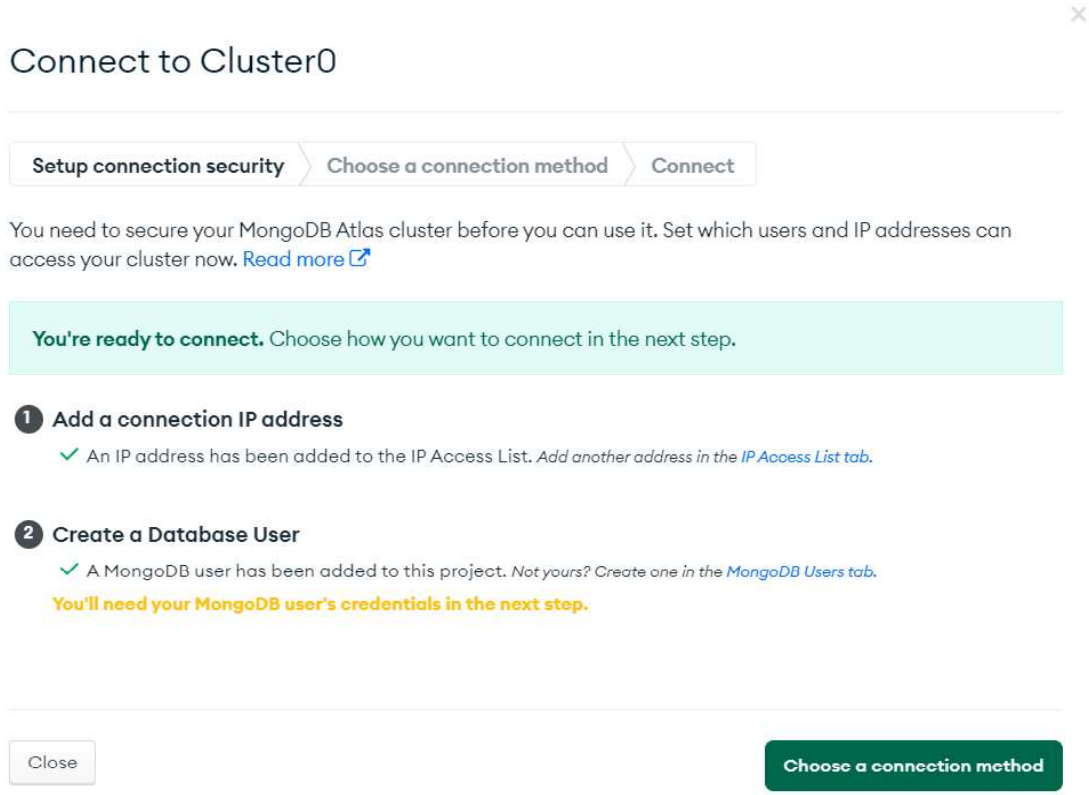


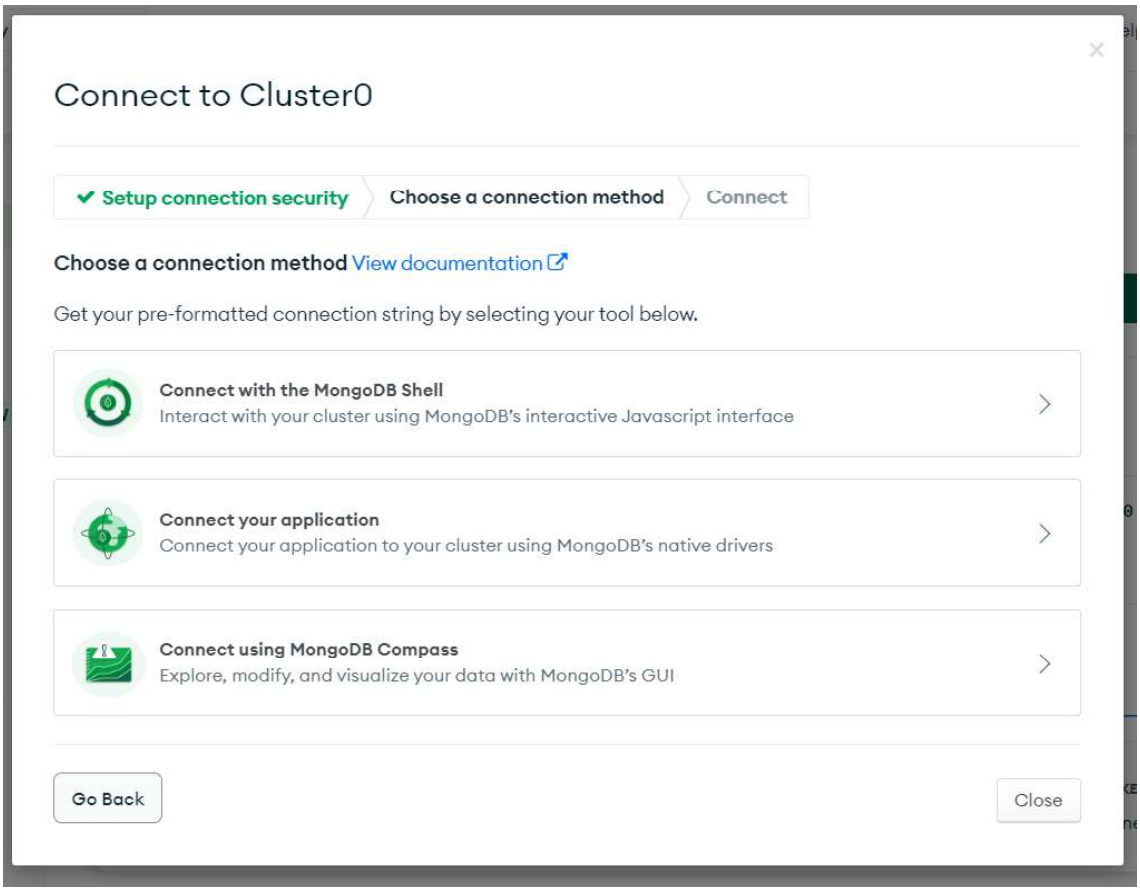


Create a **username** and **password**

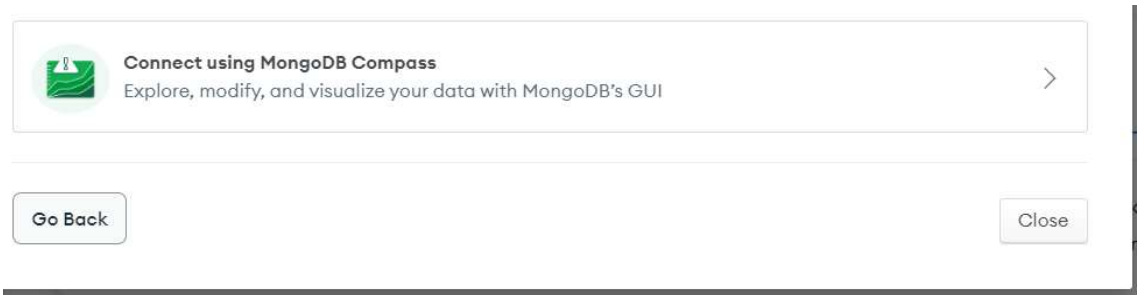


After creating username and password, click chosen your connection method

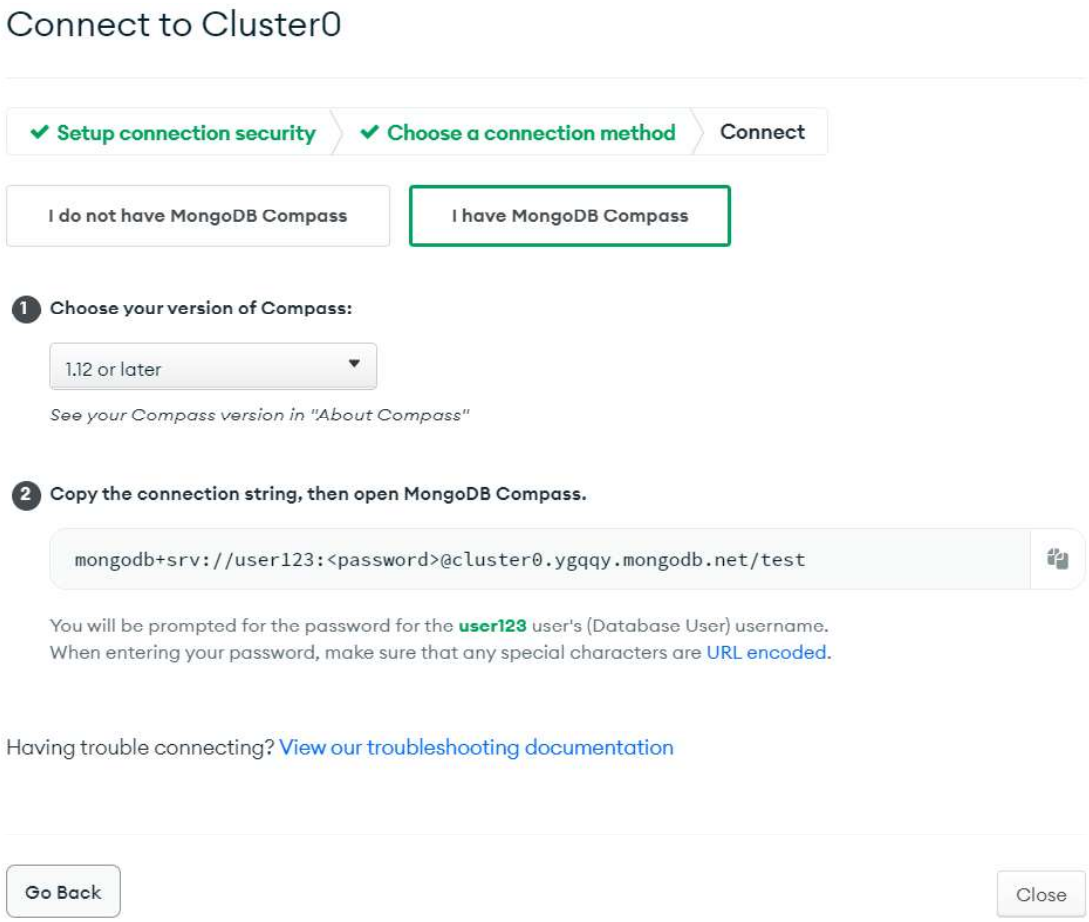




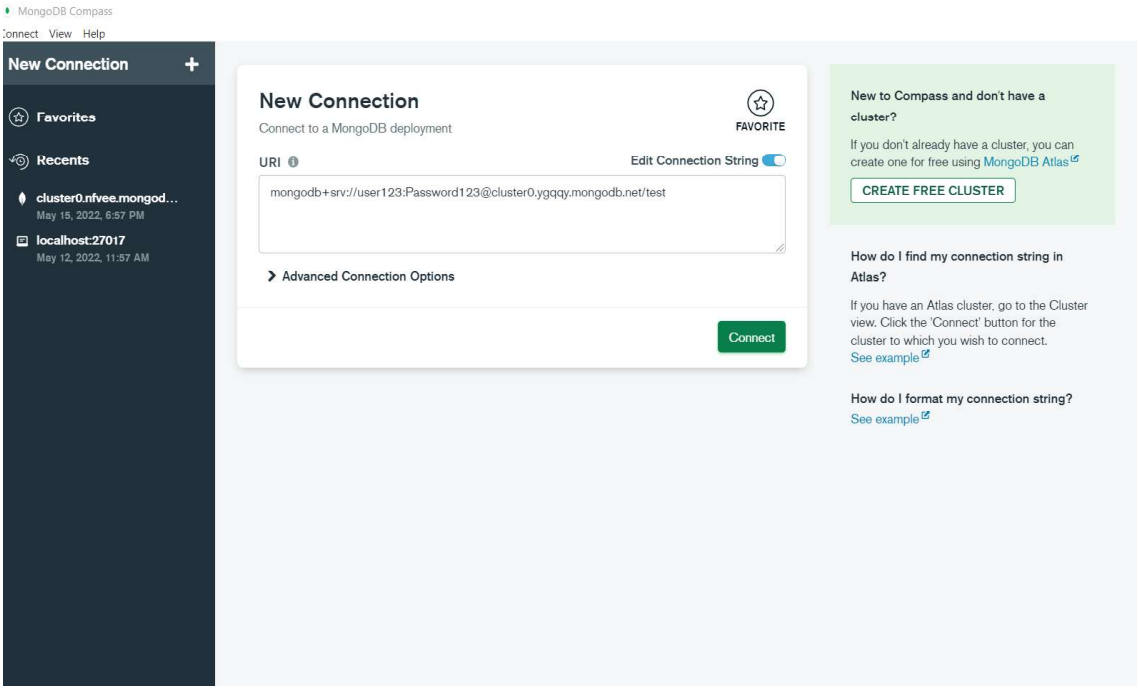
Click on connect using MongoDB Compass

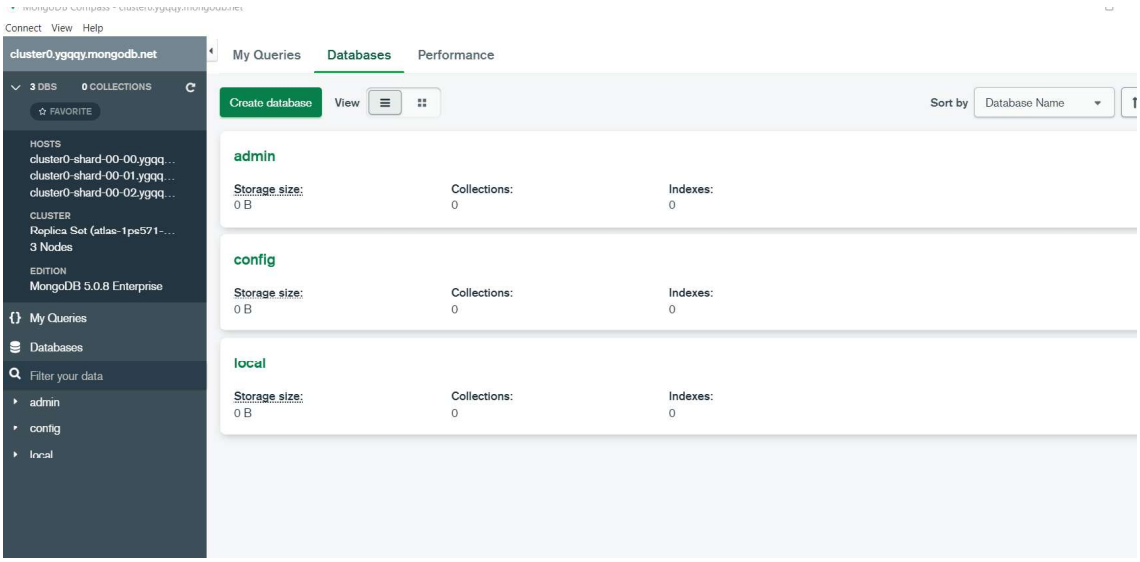


Click on I have MongoDB Compass and copy the connection string

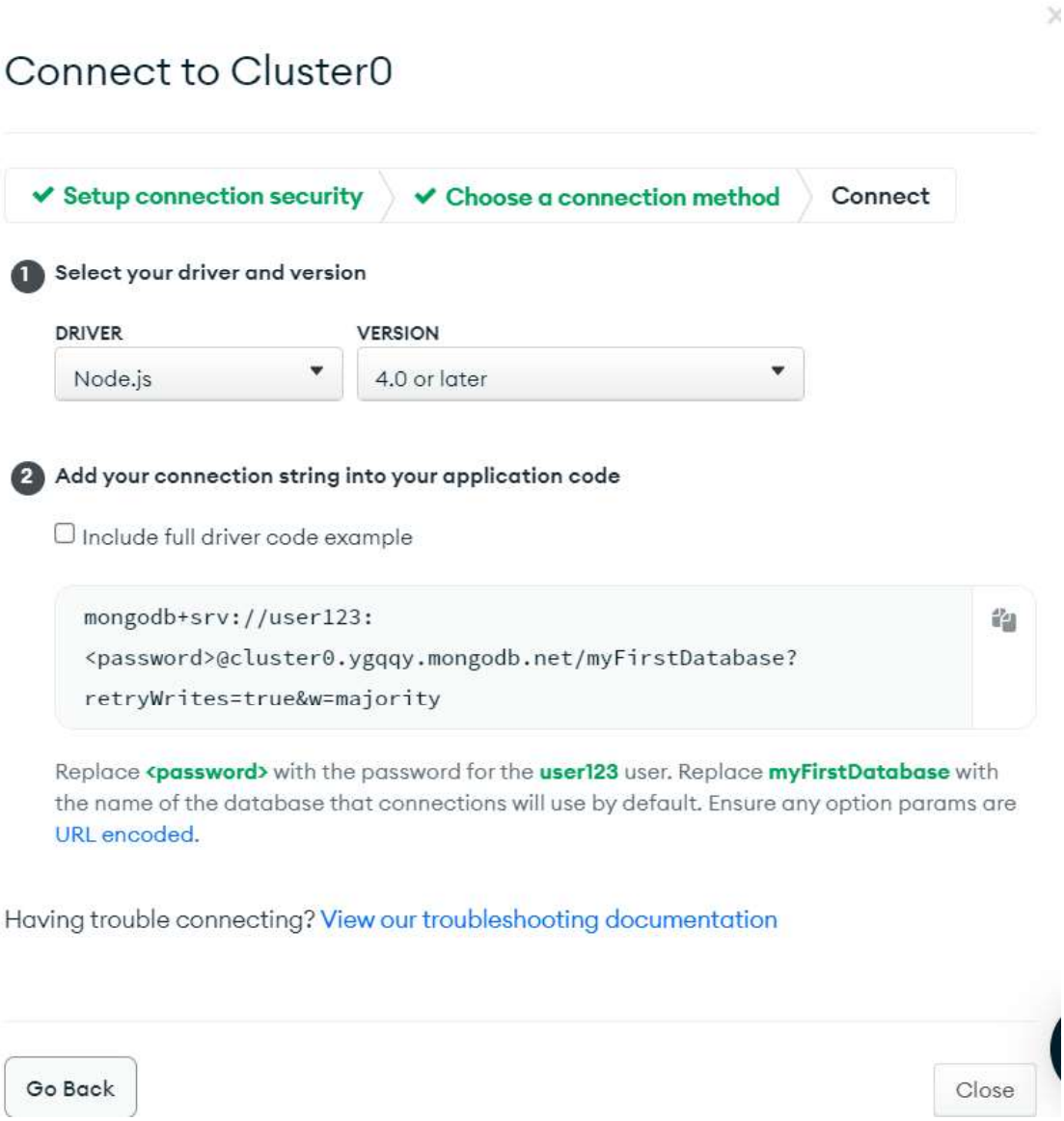


Open installed MongoDB compass and paste the connection string in MongoDB campus and write username and password correctly and click connect

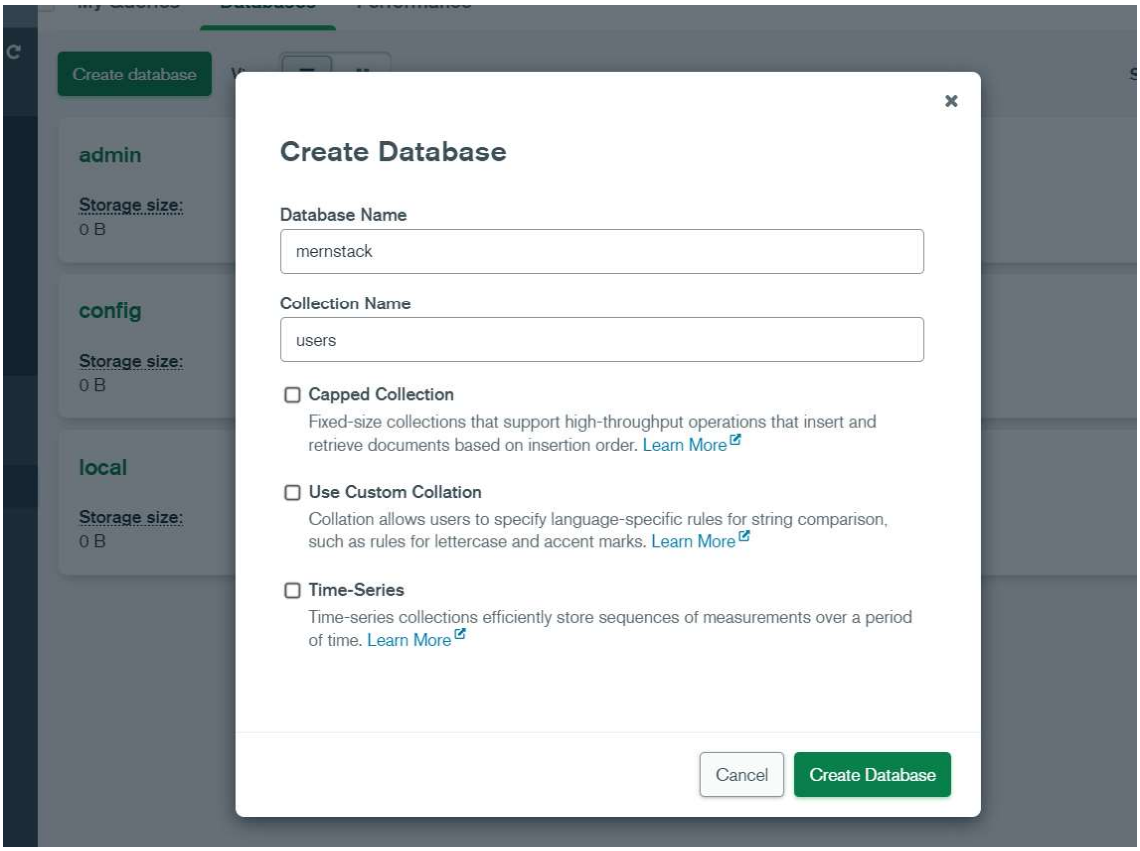


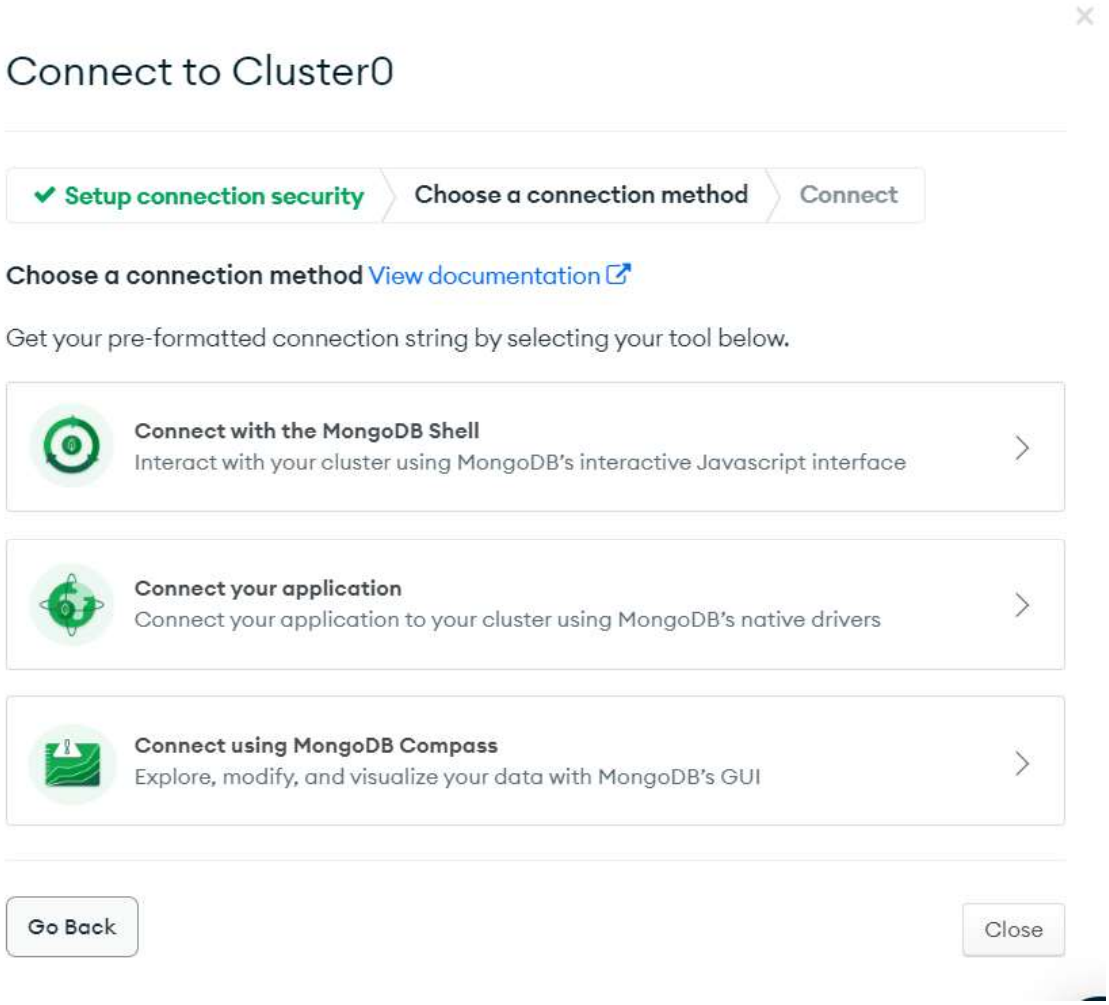


Open MongoDB cloud and click on connect again. Then click connect your application and copy the string

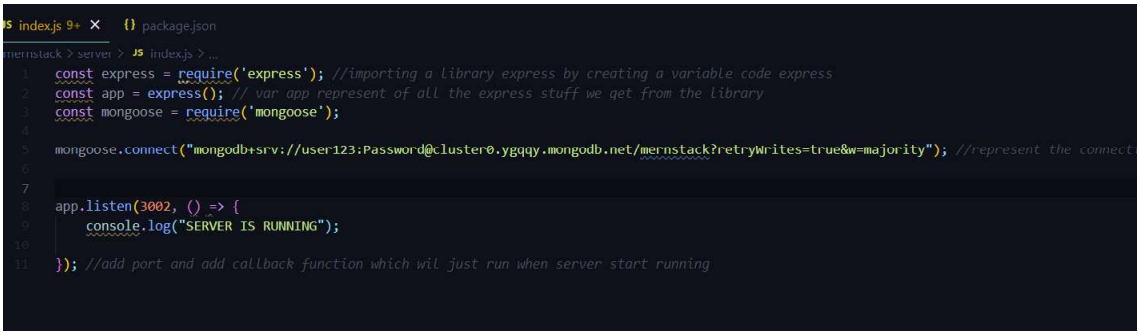


Then open MogoDB compass and create a Database


Create database
admin
Storage size:
0 B
config
Storage size:
0 B
local
Storage size:
0 B
Create Database
Database Name
mernstack
Collection Name
users
Capped Collection
Fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. Learn More
Use Custom Collation
Collation allows users to specify language-specific rules for string comparison, such as rules for lettercase and accent marks. Learn More
Time-Series
Time-series collections efficiently store sequences of measurements over a period of time. Learn More
Cancel
Create Database

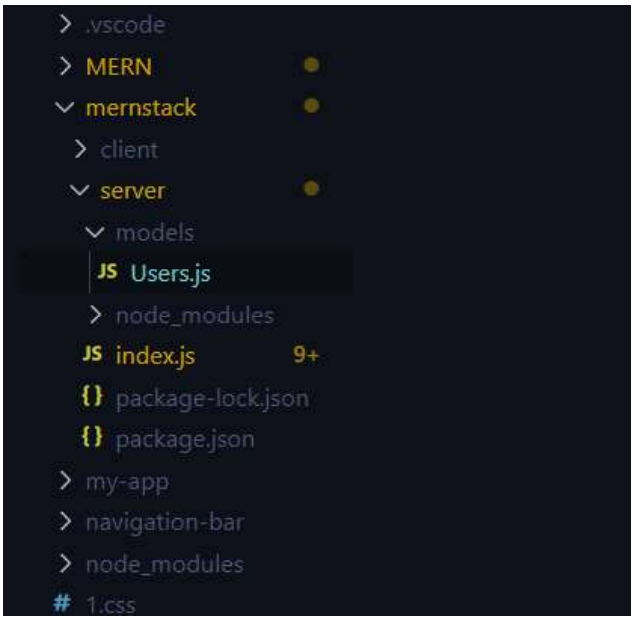


Add connection string in “index.js” file. Write your DB name instead of myfirstdatabase



```
const express = require('express'); //importing a library express by creating a variable code express
const app = express(); // var app represent of all the express stuff we get from the library
const mongoose = require('mongoose');

mongoose.connect("mongodb+srv://user123:Password@cluster0.ygqqy.mongodb.net/mernstack?retryWrites=true&w=majority"); //represent the connect


app.listen(3002, () => {
    console.log("SERVER IS RUNNING");

}); //add port and add callback function which wil just run when server start running
```

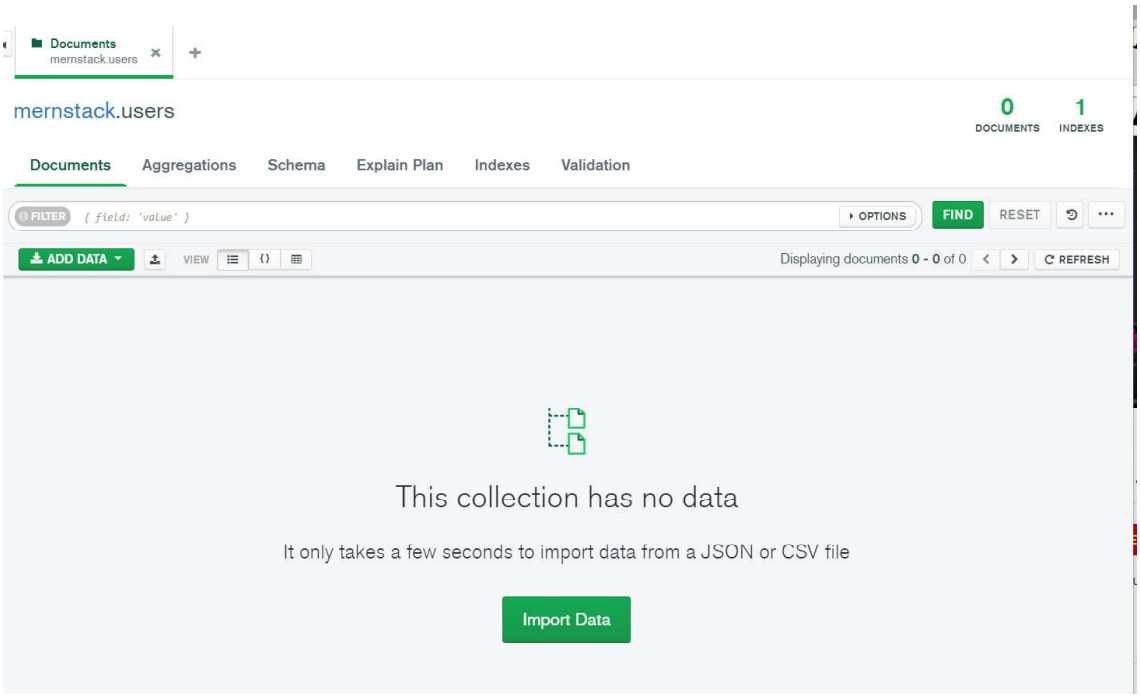
Create a new folder named “models” and create new file inside models folder named “Users.js”

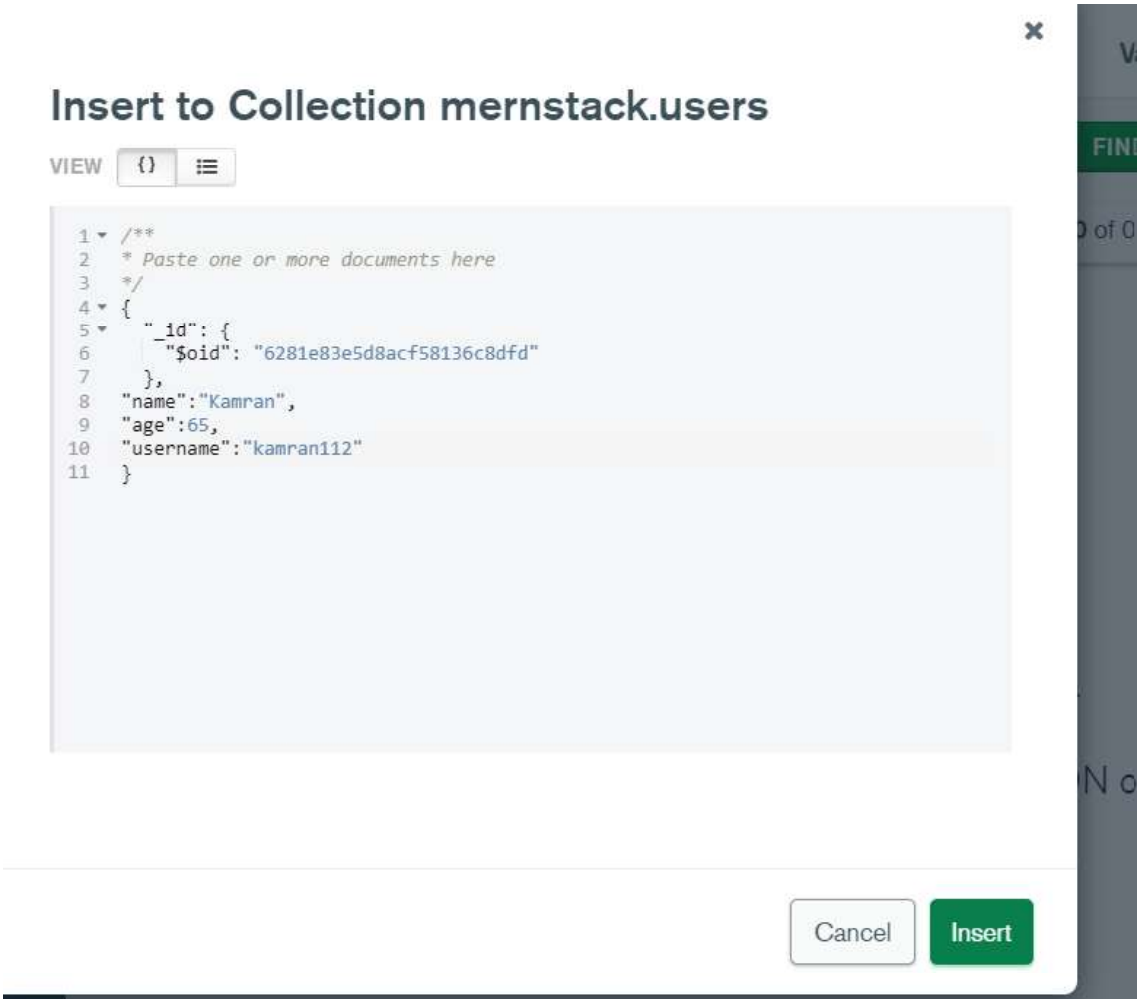


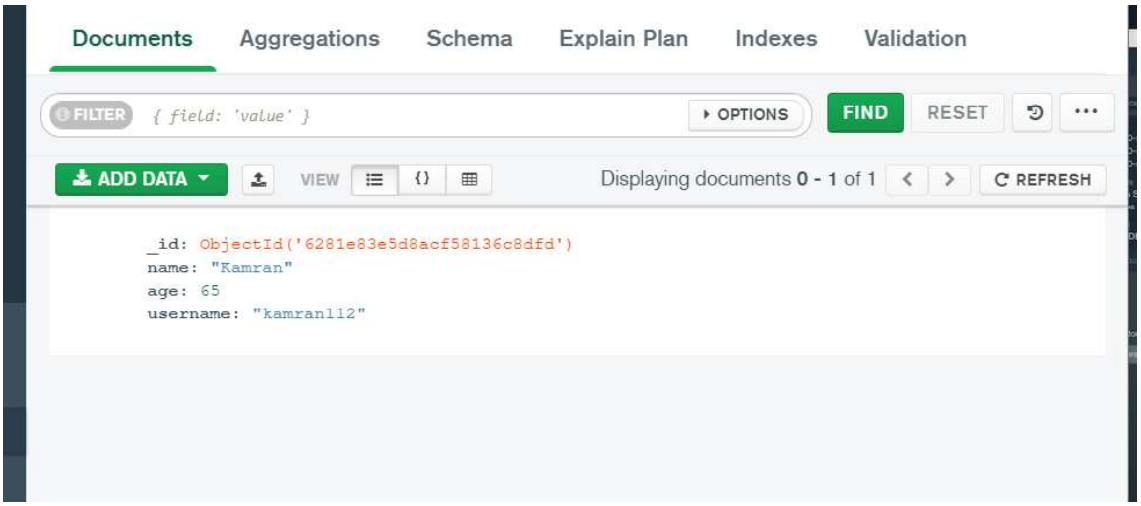
Users.js file

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

const UserSchema = new mongoose.Schema ({
    name: {
        type: String,
        required: true,
    },
    age: {
        type: number,
        required: true,
    },
    username: {
        type: String,
        required: true,
    },

});

const UserModel = mongoose.model("users", UserSchema); //collection and schema
module.exports = UserModel;
```

open MongoDB compass and add data manually by selecting Insert Document


Documents
mernstack.users
mernstack.users
0
DOCUMENTS
1
INDEXES
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
FILTER
{ field: 'value' }
OPTIONS
FIND
RESET
ADD DATA
VIEW
Displaying documents 0 - 0 of 0
REFRESH
This collection has no data
It only takes a few seconds to import data from a JSON or CSV file
Import Data


Insert to Collection mernstack.users
VIEW
/**
* Paste one or more documents here
*/
{
"_id": {
"$oid": "6281e83e5d8acf58136c8dfd"
},
"name":"Kamran",
"age":65,
"username":"kamran112"
}
Cancel
Insert

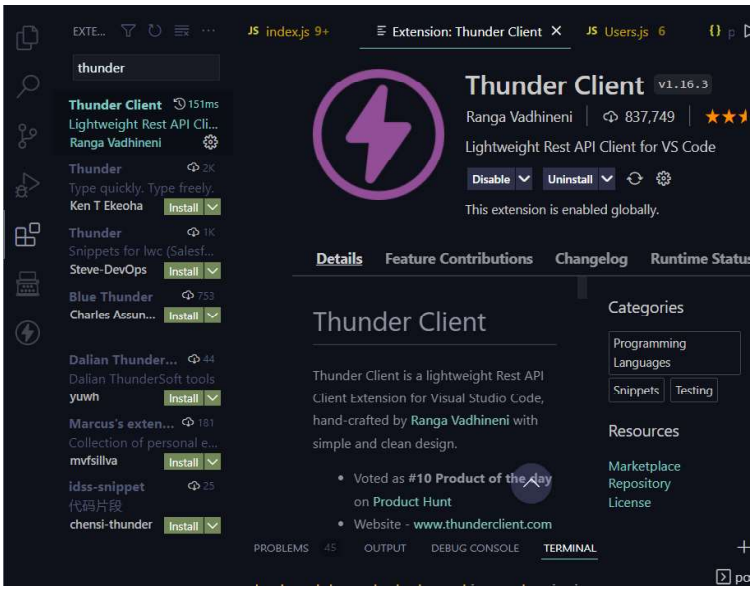


Now add in file “index.js”

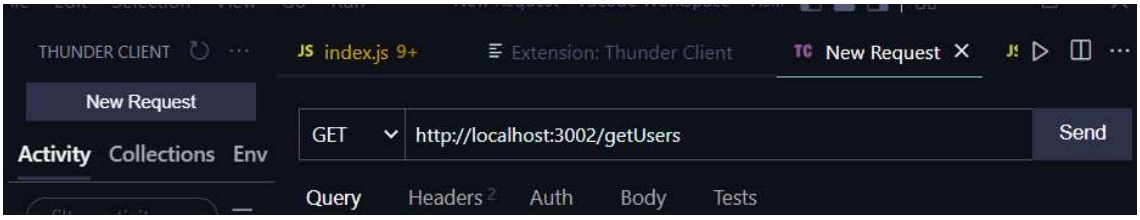
```
const UserModel = require ('./models/Users');
```

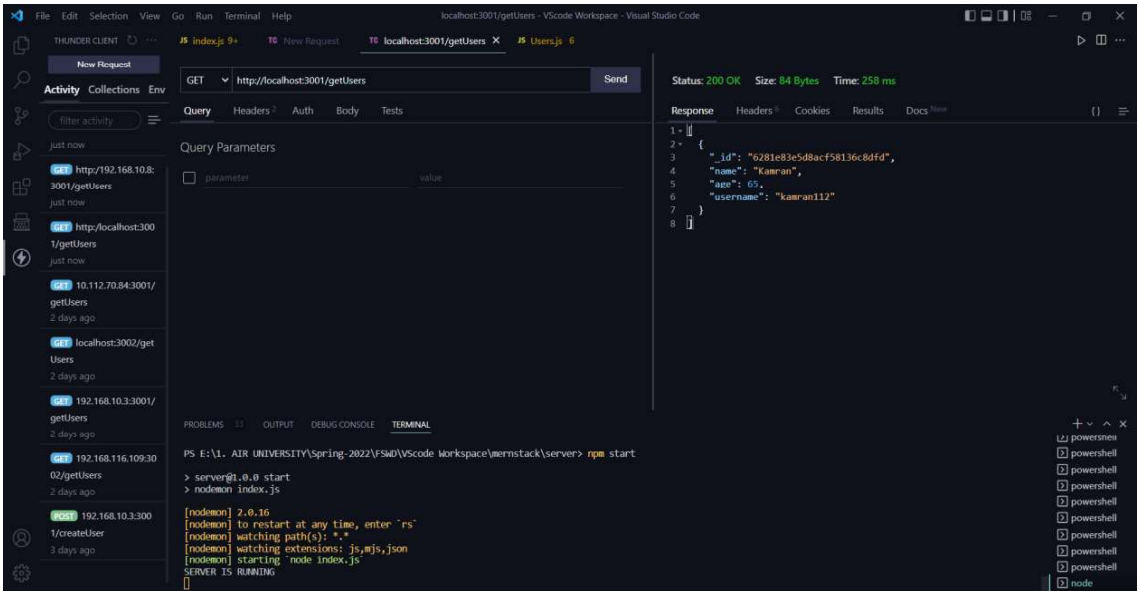
```
app.get("/getUsers", (req, res) => {
    UserModel.find({}, (err, result) => {
        if(err){
            res.json(err);

        }
        else{
            res.json(result);
        }
    });
});
```

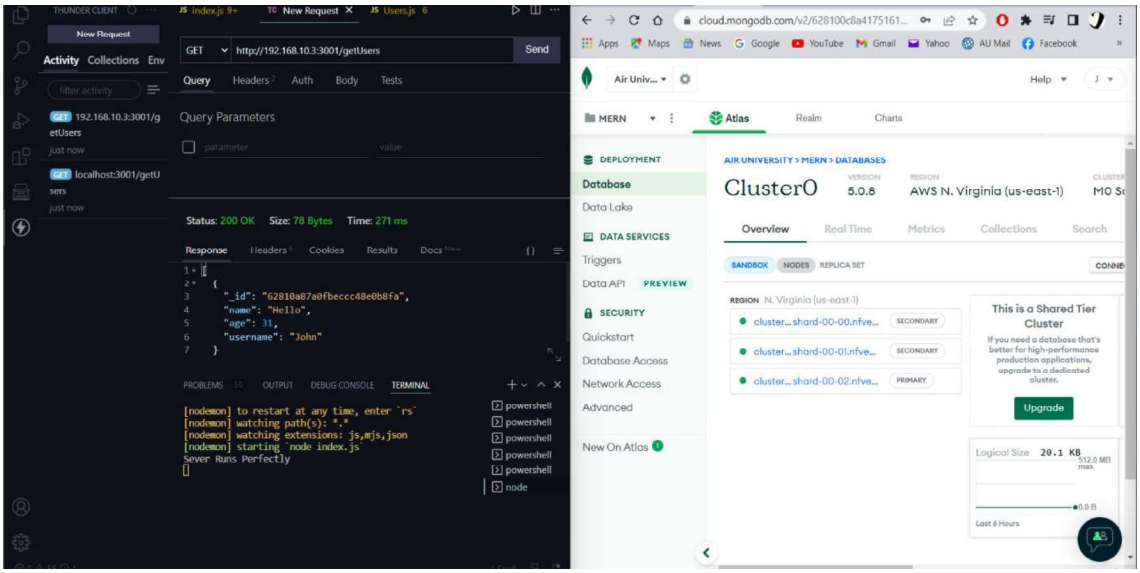
Add this extension “Thunder Client”

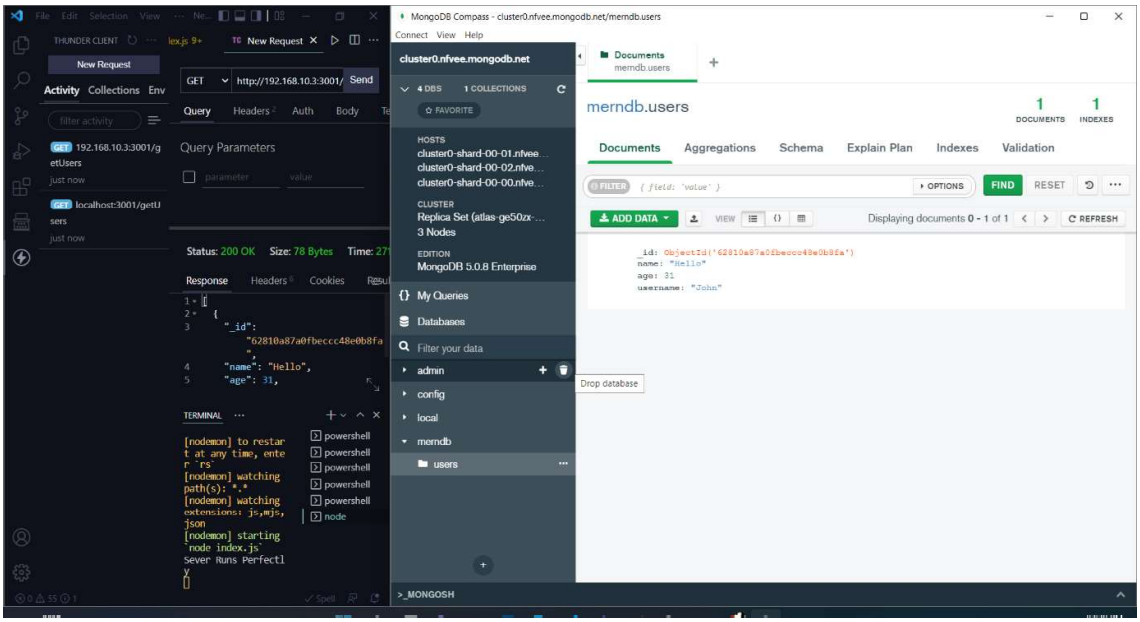


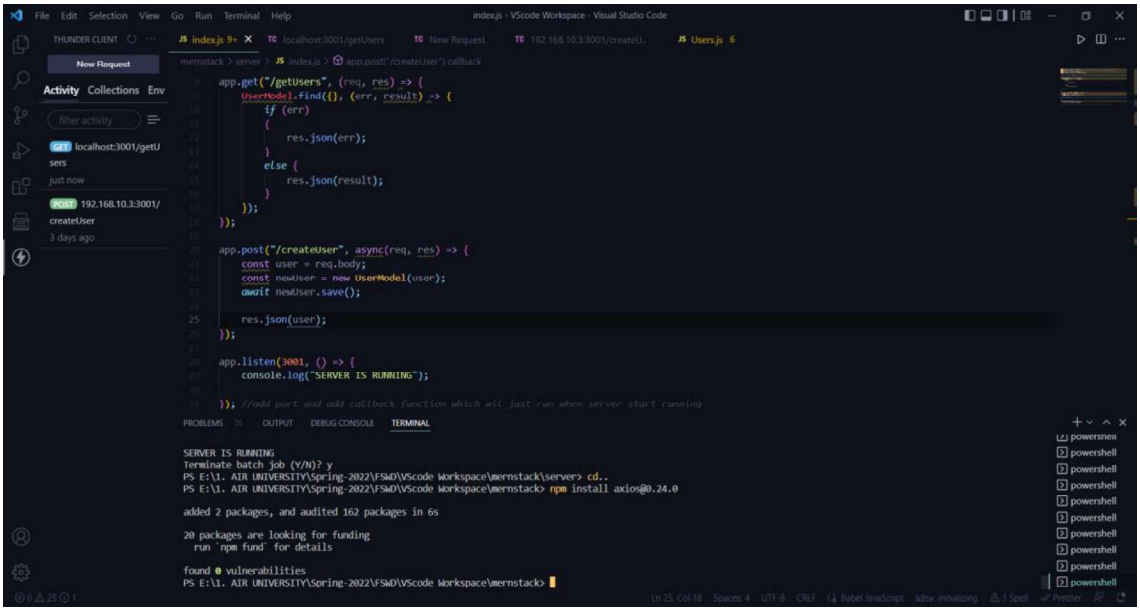
Open Thunder client and click “New Request”

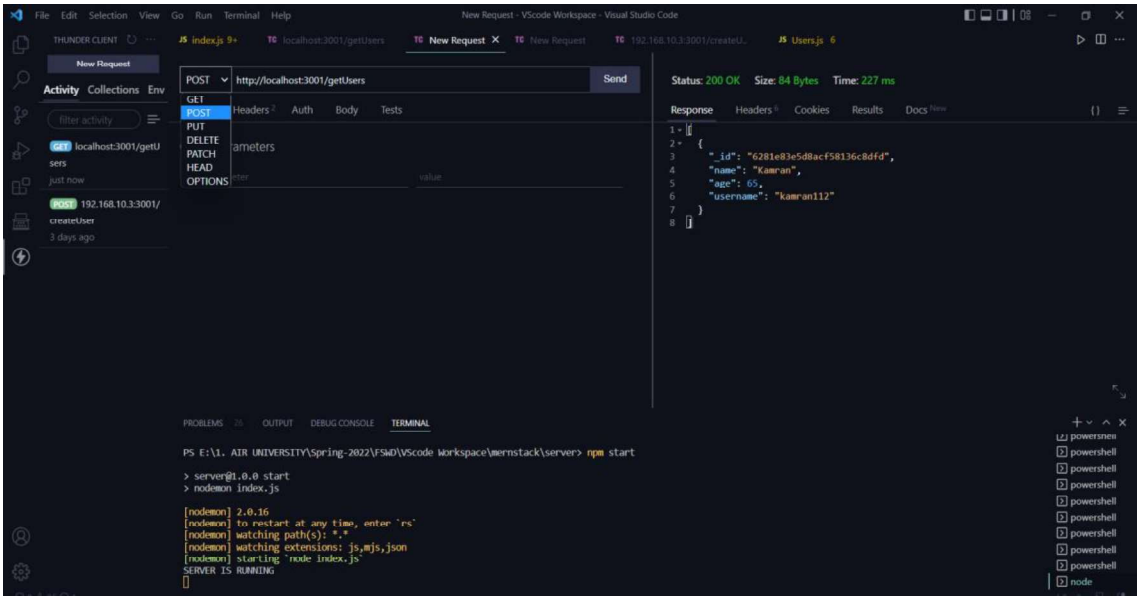


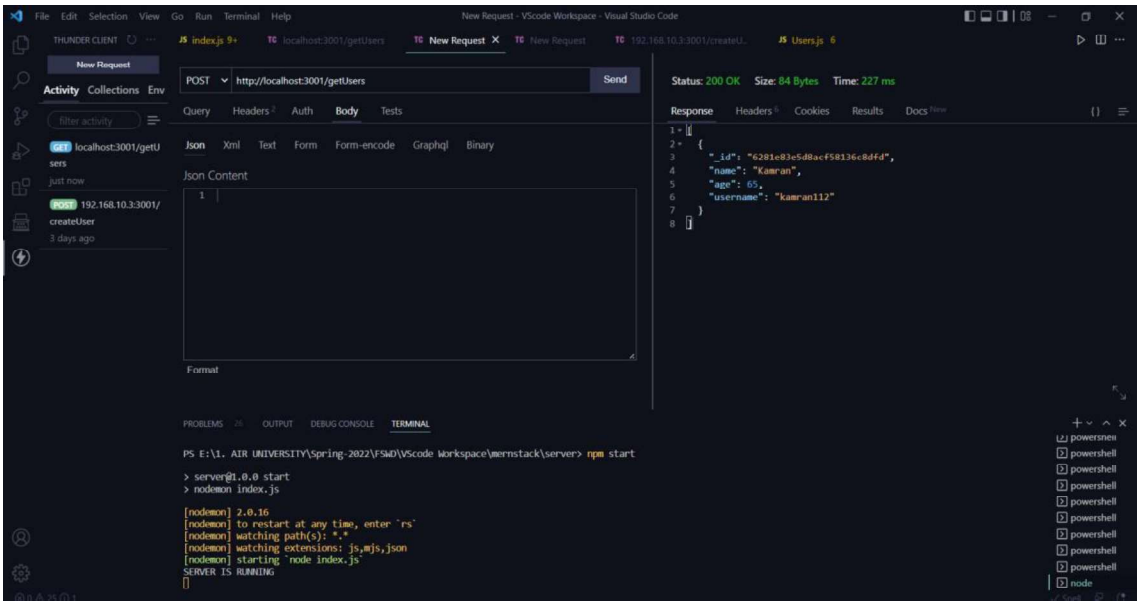


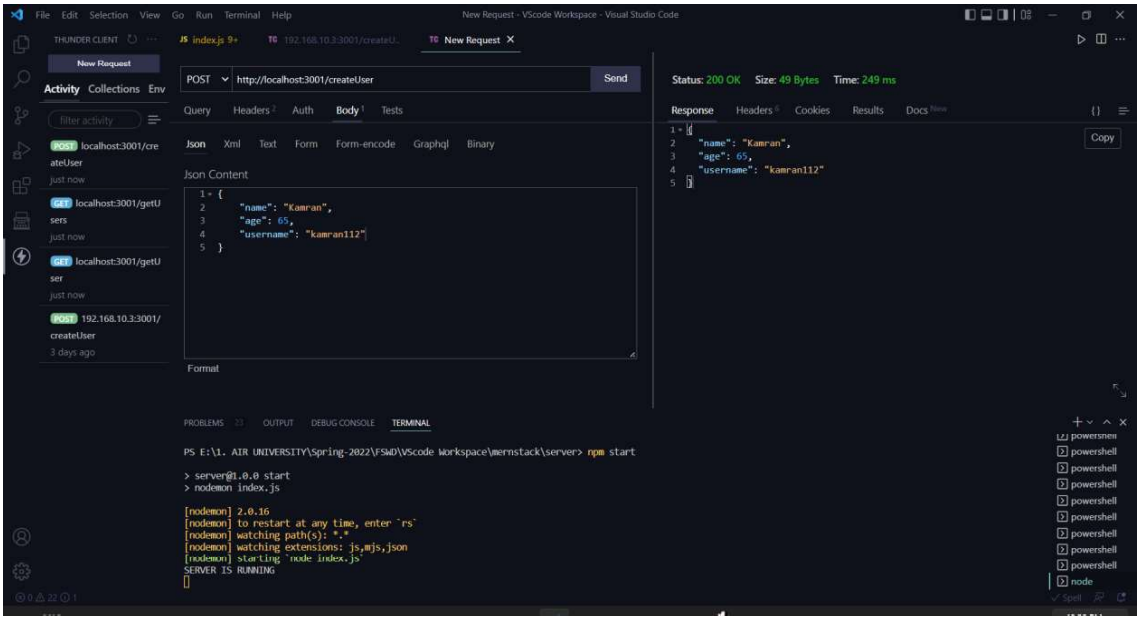

New Request
Activity Collections Env
GET http://192.168.10.3:3001/getUsers
Send
Query Headers Auth Body Tests
Query Parameters
Status: 200 OK Size: 78 Bytes Time: 271 ms
Response Headers Cookies Results Docs
"_id": "62810a87a0fbeccc48e0b8fa",
"name": "Hello",
"age": 31,
"username": "John"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Sever Runs Perfectly
Atlas Realm Charts
DEPLOYMENT
Database
Data Lake
DATA SERVICES
Triggers
Data API
SECURITY
Quickstart
Database Access
Network Access
Advanced
New On Atlas
AIR UNIVERSITY > MERN > DATABASES
Cluster0
AWS N. Virginia (us-east-1)
Overview Real Time Metrics Collections Search
This is a Shared Tier Cluster
Upgrade

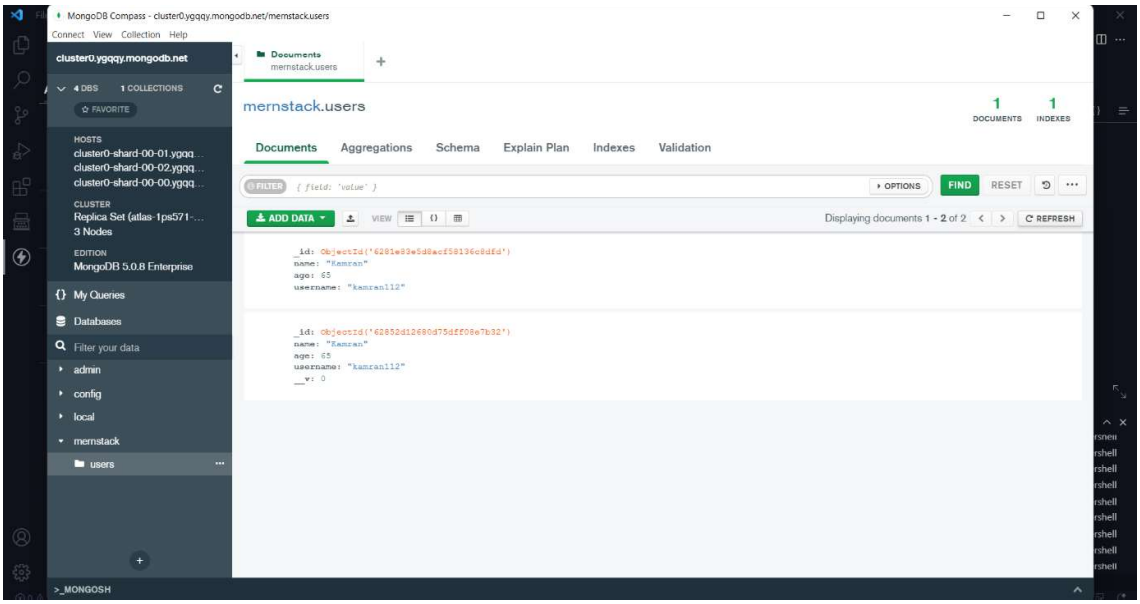

MongoDB Compass - cluster0.nfvee.mongodb.net/merndb.users
cluster0.nfvee.mongodb.net
merndb.users
Documents Aggregations Schema Explain Plan Indexes Validation
FIND RESET
ADD DATA
Displaying documents 0 - 1 of 1
admin
config
local
merndb
users
Drop database
>_MONGOSH


index.js - VScode Workspace - Visual Studio Code
app.get("/getUsers", (req, res) => {
UserModel.find({}, (err, result) => {
if (err)
{
res.json(err);
}
else {
res.json(result);
}
});
});
app.post("/createUser", async(req, res) => {
const user = req.body;
const newUser = new UserModel(user);
await newUser.save();
res.json(user);
});
app.listen(3001, () => {
console.log("SERVER IS RUNNING");
});
SERVER IS RUNNING
Terminate batch job (Y/N)? y
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> cd..
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack> npm install axios@0.24.0
added 2 packages, and audited 162 packages in 6s
20 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack>


New Request - VScode Workspace - Visual Studio Code
POST
http://localhost:3001/getUsers
Send
Status: 200 OK
Size: 84 Bytes
Time: 227 ms
"_id": "6281e83e5d8acf58136c8dfd",
"name": "Kamran",
"age": 65,
"username": "kamran112"
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> npm start
> server@1.0.0 start
> nodemon index.js
[nodemon] 2.0.16
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
SERVER IS RUNNING


New Request - VScode Workspace - Visual Studio Code
POST http://localhost:3001/getUsers
Send
Status: 200 OK Size: 84 Bytes Time: 227 ms
Response
{
"_id": "6281e83e5d8acf58136c8dfd",
"name": "Kamran",
"age": 65,
"username": "kamran112"
}
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> npm start
> server@1.0.0 start
> nodemon index.js
[nodemon] 2.0.16
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
SERVER IS RUNNING


New Request - VScode Workspace - Visual Studio Code
POST http://localhost:3001/createUser
Send
Status: 200 OK Size: 49 Bytes Time: 249 ms
Response
{
"name": "Kamran",
"age": 65,
"username": "kamran112"
}
Copy
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server> npm start
> server@1.0.0 start
> nodemon index.js
[nodemon] 2.0.16
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
SERVER IS RUNNING



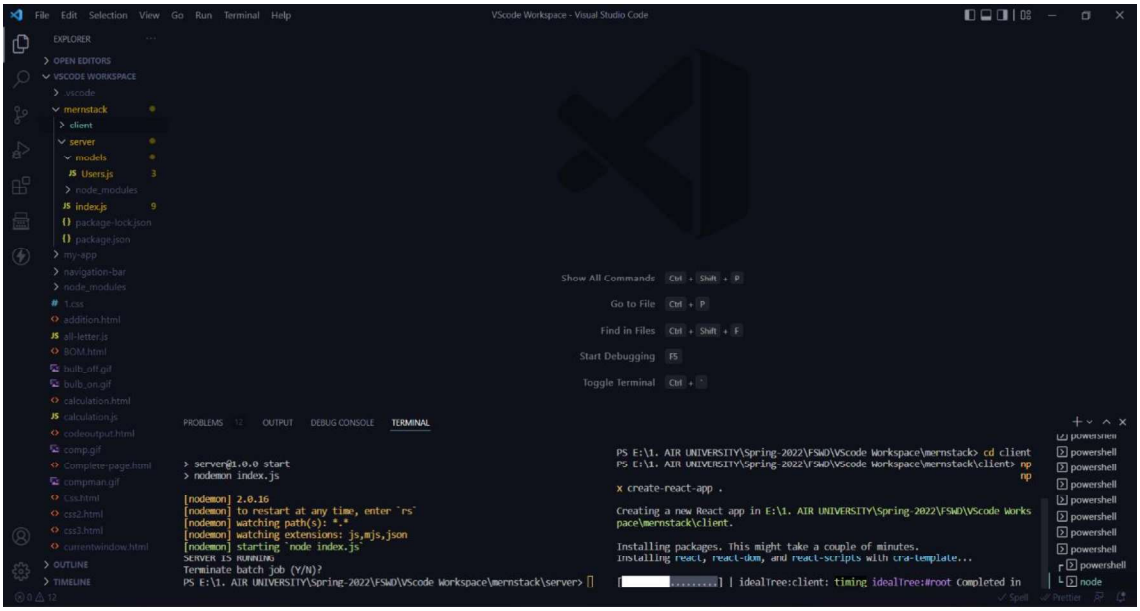
Client

New terminal

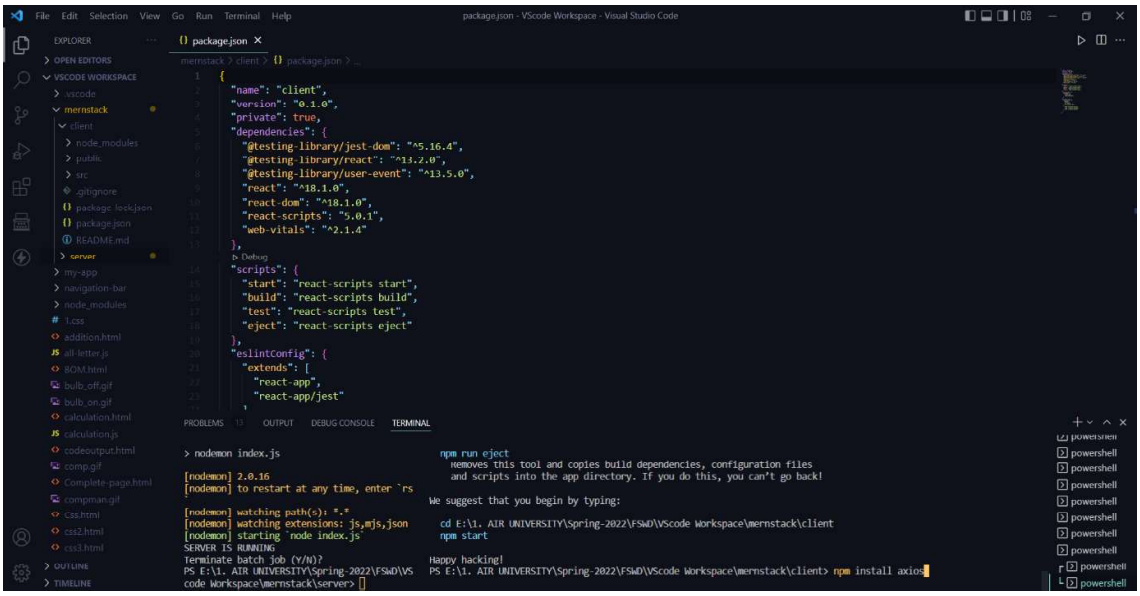
Create react app in client

cd client

npx create-react-app .

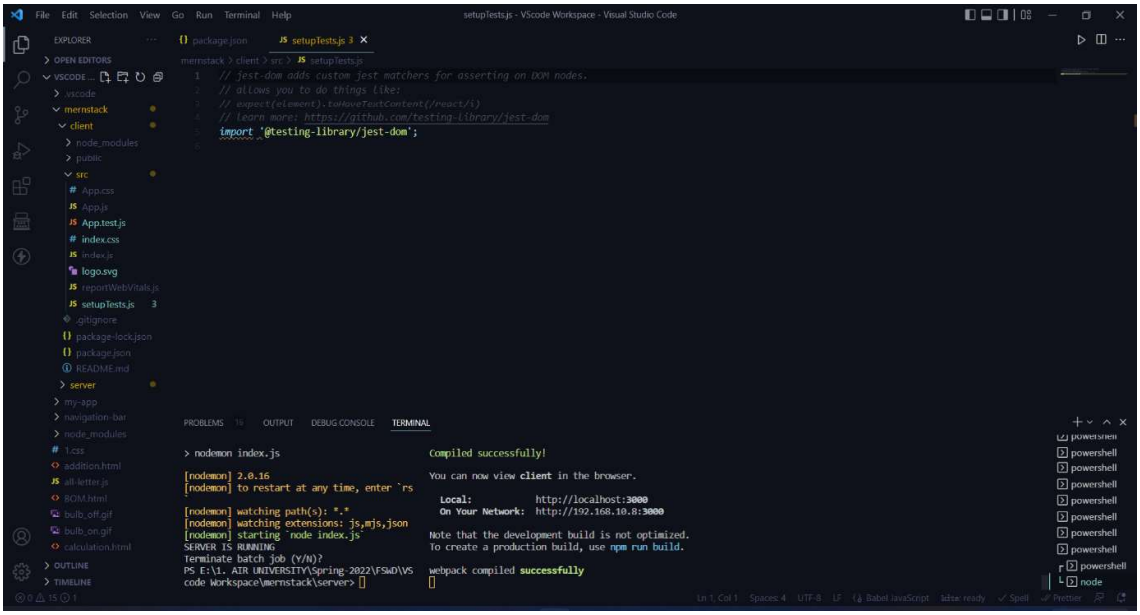


Install axios



After instaling

Delete 4 files



Write in app.js

```
import './App.css';
import {useState} from 'react';

function App() {
```

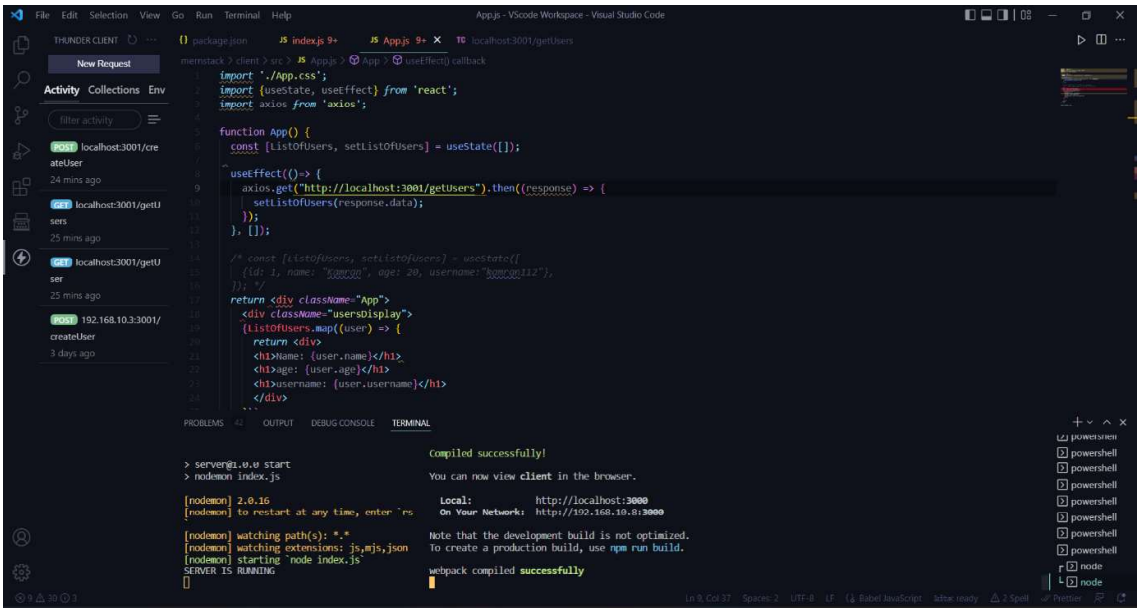
```
  const [ListOfUsers, setListOfUsers] = useState([
    {id: 1, name: "Kamran", age: 20, username:"kamran112"},
  ]);
  return <div className="App">
    <div className="usersDisplay">
    {ListOfUsers.map((user) => {
      return <div>
      <h1>Name: {user.name}</h1>
      <h1>age: {user.age}</h1>
      <h1>username: {user.username}</h1>
      </div>
    })}

    </div>
  </div>
}

export default App;
```

Name: Kamran

age: 20

username: kamran112

Start the server



Name: Kamran

age: 65

username: kamran112

Name: Kamran

age: 65

username: kamran112

Name: Kamran

age: 65

username: kamran112

Name: Kamran

age: 65

username: kamran112

Name... | Age... | Username... | Create User

```
const createUser =() => {
  axios.post("http://localhost:3001/createUser").then((response) => {
    alert("USER CREATED");
  });
};



<button onClick={createUser}>Create User</button>
```

Then

Add in function app()

```
const [name, setName] = useState("");
const [age, setAge] = useState(65);
const [username, setUsername] = useState("");
```

```
useEffect(()=> {
  axios.get("http://localhost:3001/getUsers").then((response) => {
    setListOfUsers(response.data);
  });
}, []);

const createUser =() => {
  axios.post("http://localhost:3001/createUser", {
    name,
    age,
    username,
  }).then((response) => {
    alert("USER CREATED");
  });
};
```

```
<input type="text" placeholder="Name..." onChange={(event) =>
{setName(event.target.value)}}/>
      <input type="number" placeholder="Age..." onChange={(event) =>
{setAge(event.target.value)}}/>
      <input type="text" placeholder="Username..." onChange={(event) =>
{setUsername(event.target.value)}}/>
      <button onClick={createUser}>Create User</button>
```

Complete Code

```
import './App.css';
import {useState, useEffect} from 'react';
import axios from 'axios';

function App() {
  const [ListOfUsers, setListOfUsers] = useState([]);
  const [name, setName] = useState("");
  const [age, setAge] = useState(65);
  const [username, setUsername] = useState("");

  useEffect(()=> {
    axios.get("http://localhost:3001/getUsers").then((response) => {
      setListOfUsers(response.data);
    });
  }, []);

  const createUser =() => {
    axios.post("http://localhost:3001/createUser", {
      name,
      age,
      username,
    }).then((response) => {
      alert("USER CREATED");
    });
  };

  /* const [ListOfUsers, setListOfUsers] = useState([
    {id: 1, name: "Kamran", age: 20, username:"kamran112"},
  ]); */
  return <div className="App">
    <div className="usersDisplay">
    {ListOfUsers.map((user) => {
      return <div>
      <h1>Name: {user.name}</h1>
      <h1>age: {user.age}</h1>
```

```
      <h1>username: {user.username}</h1>
      </div>
    })}

    </div>

    <div>
      <input type="text" placeholder="Name..." onChange={(event) =>
{setName(event.target.value)}}/>
      <input type="number" placeholder="Age..." onChange={(event) =>
{setAge(event.target.value)}}/>
      <input type="text" placeholder="Username..." onChange={(event) =>
{setUsername(event.target.value)}}/>
      <button onClick={createUser}>Create User</button>
    </div>
  </div>
}

export default App;
```

Refresh page

Name: Kamran

age: 65

username: kamran112

Name: Kamran

age: 65

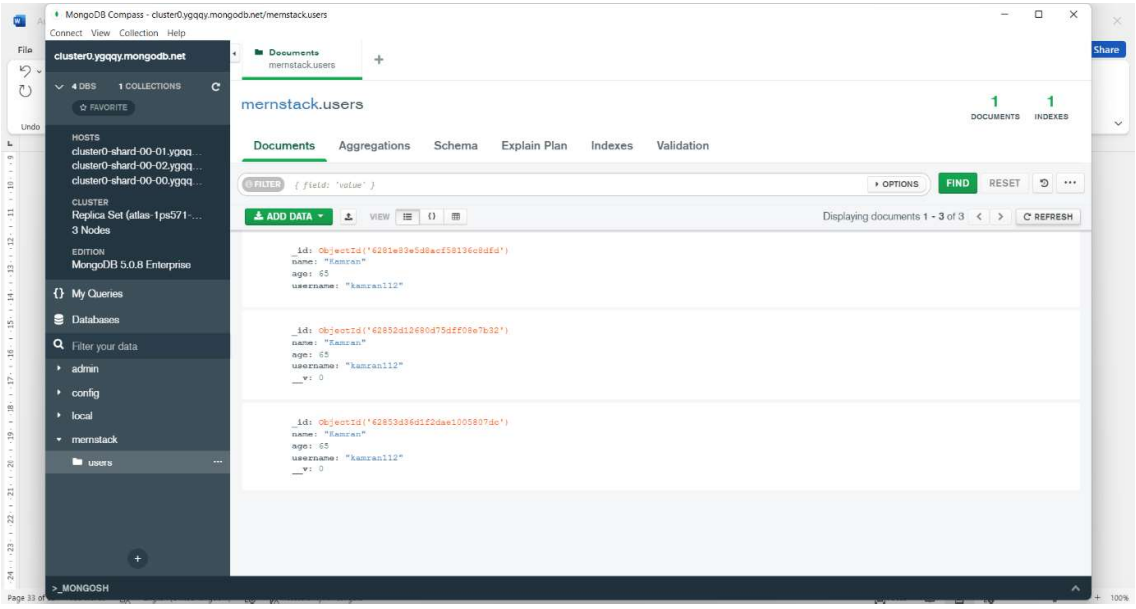
username: kamran112

Name: Kamran

age: 65

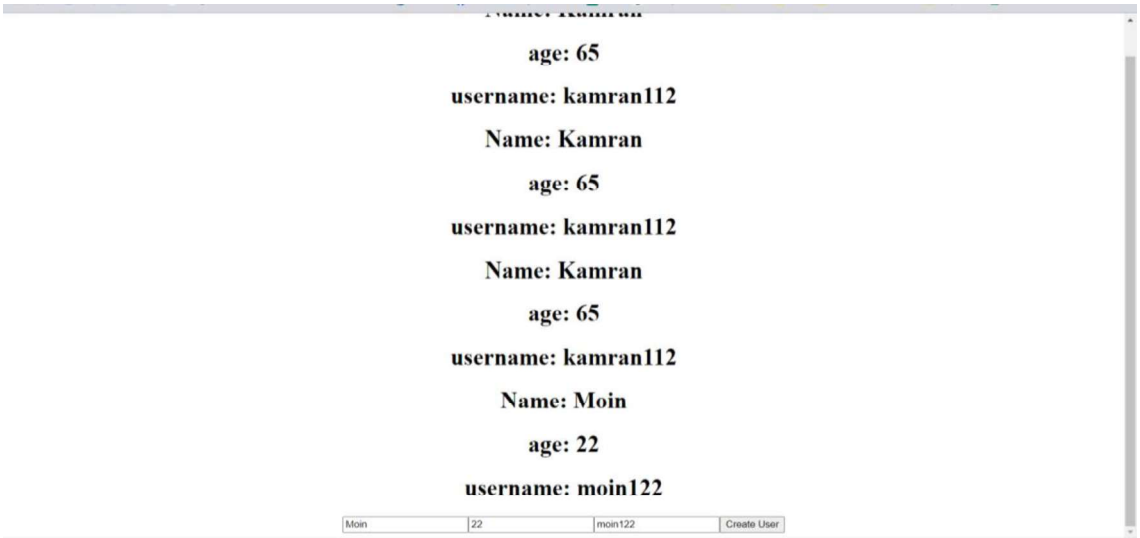
username: kamran112

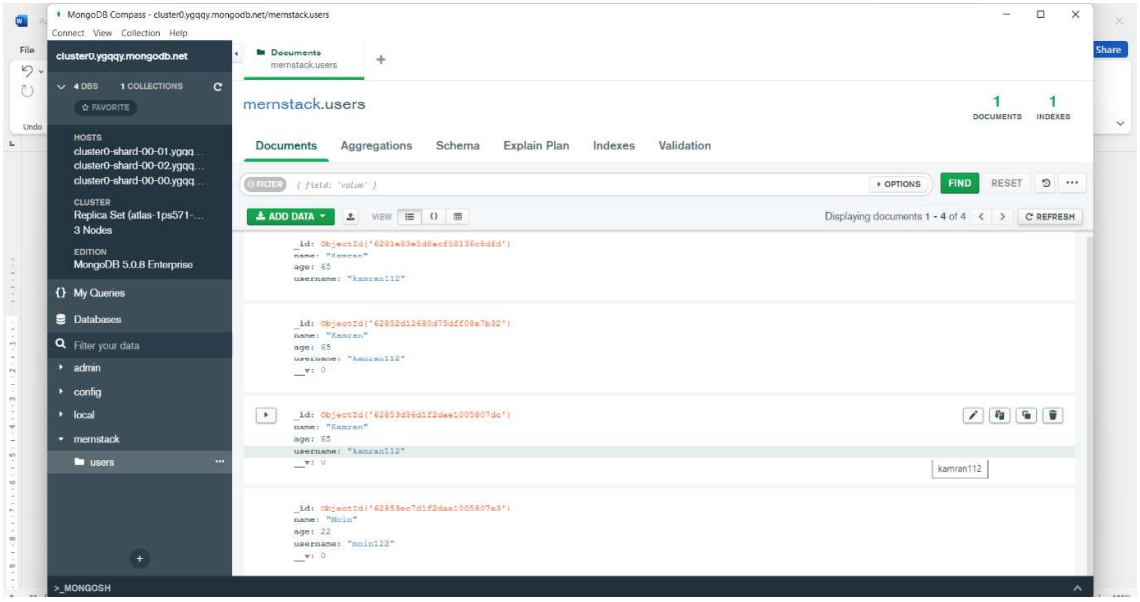
Name... | Age... | Username... | Create User



Add this for autoupdate

```
const createUser =() => {
  axios.post("http://localhost:3001/createUser", {
    name,
    age,
    username,
  }).then((response) => {
    setListOfUsers([...ListOfUsers,
      {
        name,
        age,
        username,
      },
    ]);
    //alert("USER CREATED");
  });
};
```





Project name:

Mernstack

Folder

client

server

index.js file

```
const express = require('express'); //importing a library express by creating a variable code express
```

```
const app = express(); // var app represent of all the express stuff we get from the library
const mongoose = require('mongoose');
const UserModel = require ('./models/Users');

const cors = require('cors');

app.use(express.json());
app.use(cors());

mongoose.connect("mongodb+srv://user123:Password123@cluster0.ygqqy.mongodb.net/mernstack?retryWrites=true&w=majority"); //represent the connection of mongodb

app.get("/getUsers", (req, res) => {
    UserModel.find({}, (err, result) => {
        if (err)
        {
            res.json(err);
        }
        else {
            res.json(result);
        }
    });
});

app.post("/createUser", async(req, res) => {
    const user = req.body;
    const newUser = new UserModel(user);
    await newUser.save();

    res.json(user);
});

app.listen(3001, () => {
    console.log("SERVER IS RUNNING");

}); //add port and add callback function which wil just run when server start running
```

Folder Models

Users.js file

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

const UserSchema = new mongoose.Schema({
    name: {
        type: String,
        required: true,
    },
```

```
    age: {
        type: Number,
        required: true,
    },
    username:{
        type: String,
        required: true,
    },
});

const UserModel = mongoose.model("users",UserSchema); //collection and schema
module.exports = UserModel;
```

Folder “Client”

App.js file

```
import './App.css';
import {useState, useEffect} from 'react';
import axios from 'axios';

function App() {
  const [ListOfUsers, setListOfUsers] = useState([]);
  const [name, setName] = useState("");
  const [age, setAge] = useState(65);
  const [username, setUsername] = useState("");

  useEffect(()=> {
    axios.get("http://localhost:3001/getUsers").then((response) => {
      setListOfUsers(response.data);
    });
  }, []);

  const createUser =() => {
    axios.post("http://localhost:3001/createUser", {
      name,
      age,
      username,
    }).then((response) => {
      setListOfUsers([...ListOfUsers,
        {
          name,
          age,
          username,
        },
      ]);
      //alert("USER CREATED");
    });
  };
```

```
 /* const [ListOfUsers, setListOfUsers] = useState([
   {id: 1, name: "Kamran", age: 20, username:"kamran112"},
 ]); */
 return <div className="App">
   <div className="usersDisplay">
   {ListOfUsers.map((user) => {
     return <div>
     <h1>Name: {user.name}</h1>
     <h1>age: {user.age}</h1>
     <h1>username: {user.username}</h1>
     </div>
   })}

   </div>

   <div>
     <input type="text" placeholder="Name..." onChange={(event) =>
{setName(event.target.value)}}/>
     <input type="number" placeholder="Age..." onChange={(event) =>
{setAge(event.target.value)}}/>
     <input type="text" placeholder="Username..." onChange={(event) =>
{setUsername(event.target.value)}}/>
     <button onClick={createUser}>Create User</button>
   </div>
 </div>
}

export default App;
```