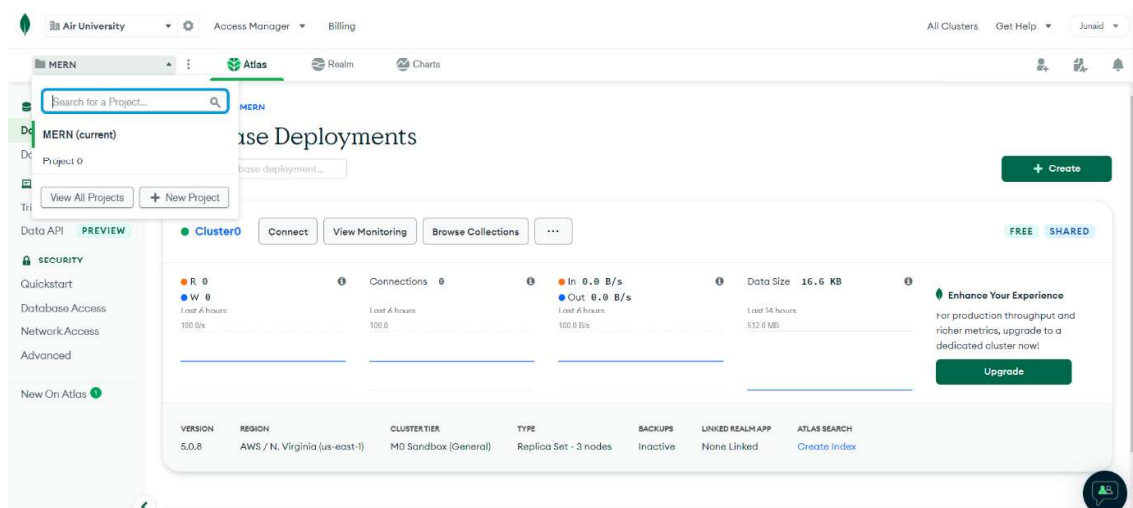


Statement Purpose:

To familiarize the students with

Learning Objectives

- Connectivity with MongoDB
 - Data Find form MongoDB
 - Data Insert into MongoDB
- Download MongoDB from <https://www.mongodb.com/try/download/community>
 - Create an atlas account on <https://cloud.mongodb.com/>
 - After sign-in on new account, create new project



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

Air University

Access Manager

Billing

ORGANIZATION

Projects

Alerts

Activity Feed

Settings

Integrations

Access Manager

Billing

Support

Live Migration

AIR UNIVERSITY > PROJECTS

Create a Project

Name Your Project Add Members

Name Your Project

Project names have to be unique within the organization (and other restrictions).

Mernstack

Cancel Next

Click on “Create project”

AIR UNIVERSITY > PROJECTS

Create a Project

✓ Name Your Project Add Members

Add Members and Set Permissions

Invite new or existing users via email address...

Give your members access permissions below.

junaidcs122.aaur@gmail.com (you)	Project Owner
----------------------------------	---------------

Cancel Go Back Create Project

Click on “Build a Database”

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

Cloud Provider & Region

AWS, N. Virginia (us-east-1) ▼

aws

Google Cloud

Azure

★ Recommended region ⓘ

🏷️ Dedicated tier region ⓘ

NORTH AMERICA	EUROPE	AUSTRALIA
🇺🇸 Oregon (us-west-2) ★	🇫🇷 Paris (eu-west-3) ★	🇦🇺 Sydney (ap-southeast-2) ★
🇺🇸 N. Virginia (us-east-1) ★	🇩🇪 Frankfurt (eu-central-1) ★	ASIA
🇺🇸 Ohio (us-east-2) ★ 🏷️	🇸🇪 Stockholm (eu-north-1) ★	🇰🇷 Seoul (ap-northeast-2)
🇺🇸 N. California (us-west-1) 🏷️	🇮🇪 Ireland (eu-west-1) ★	🇯🇵 Tokyo (ap-northeast-1)

FREE

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Back

Create Cluster

You can also change the cluster name

Cluster Tier

M0 Sandbox (Shared RAM, 512 MB Storage) Encrypted ^

Additional Settings

MongoDB 5.0, No Backup ^

Cluster Name

Cluster0 ▼

One time only: once your cluster is created, you won't be able to change its name.

Cluster0

Cluster names can only contain ASCII letters, numbers, and hyphens.

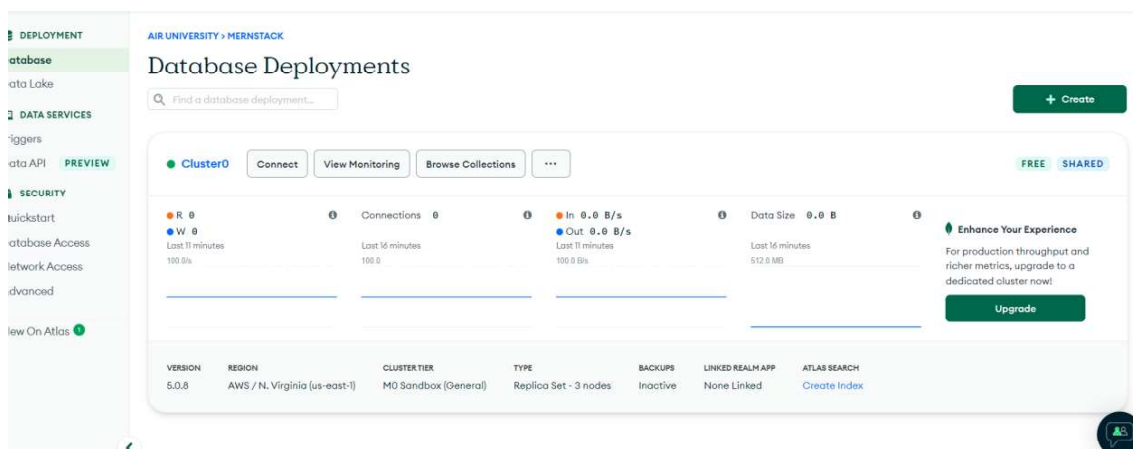
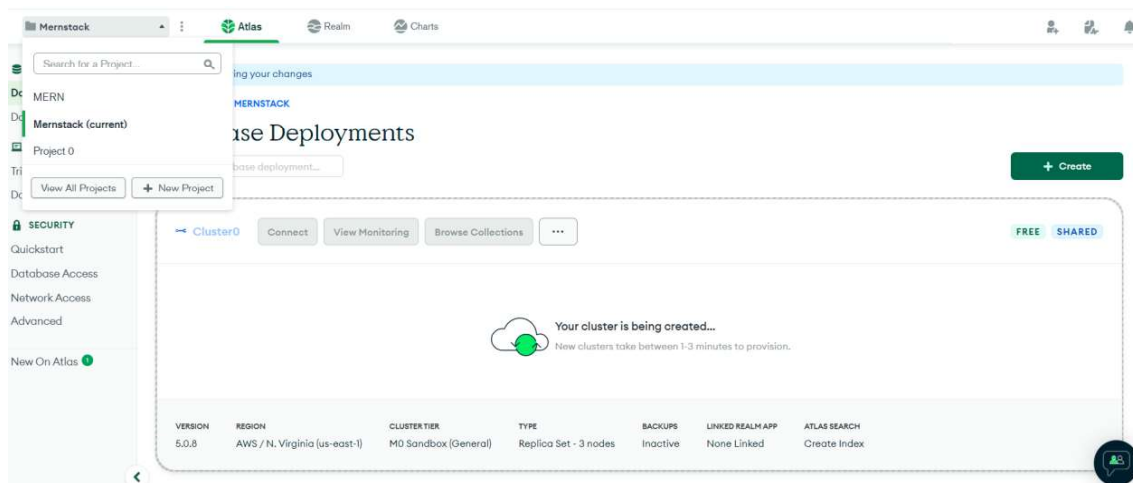
FREE

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Back

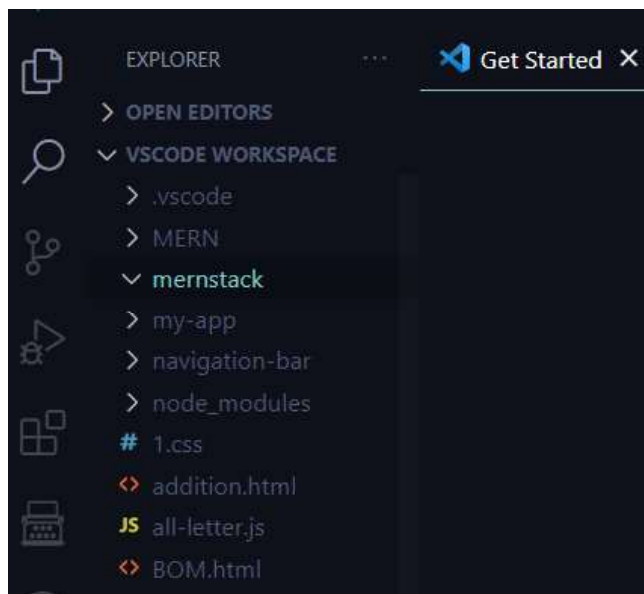
Create Cluster

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

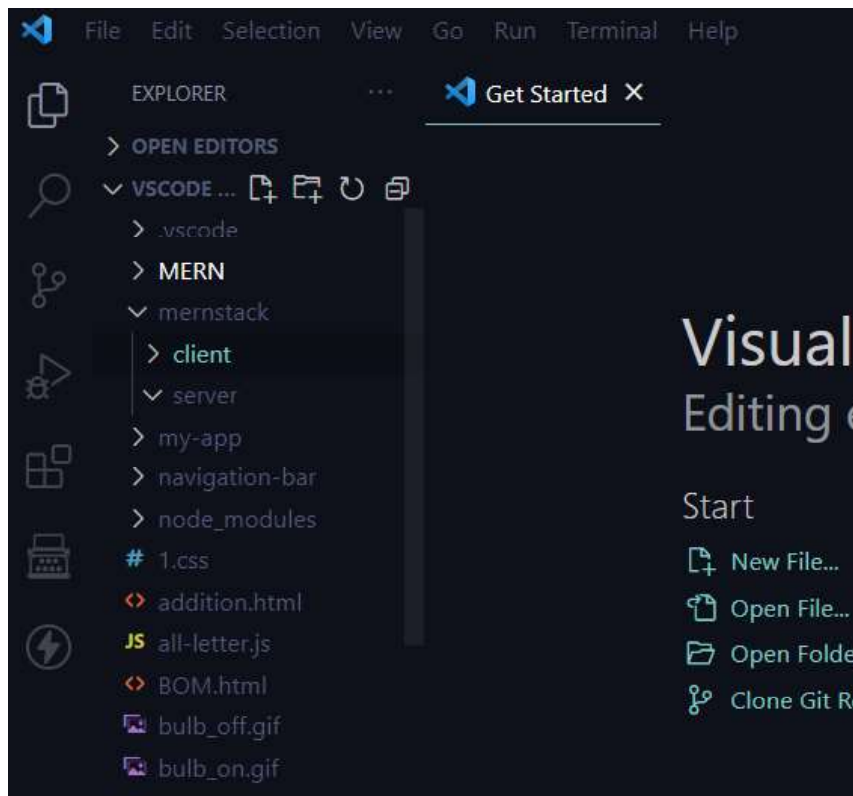


After creating a cluster

Create a folder in your workspace



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



Open Terminal and write

```
cd mernstack
```

```
cd server
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

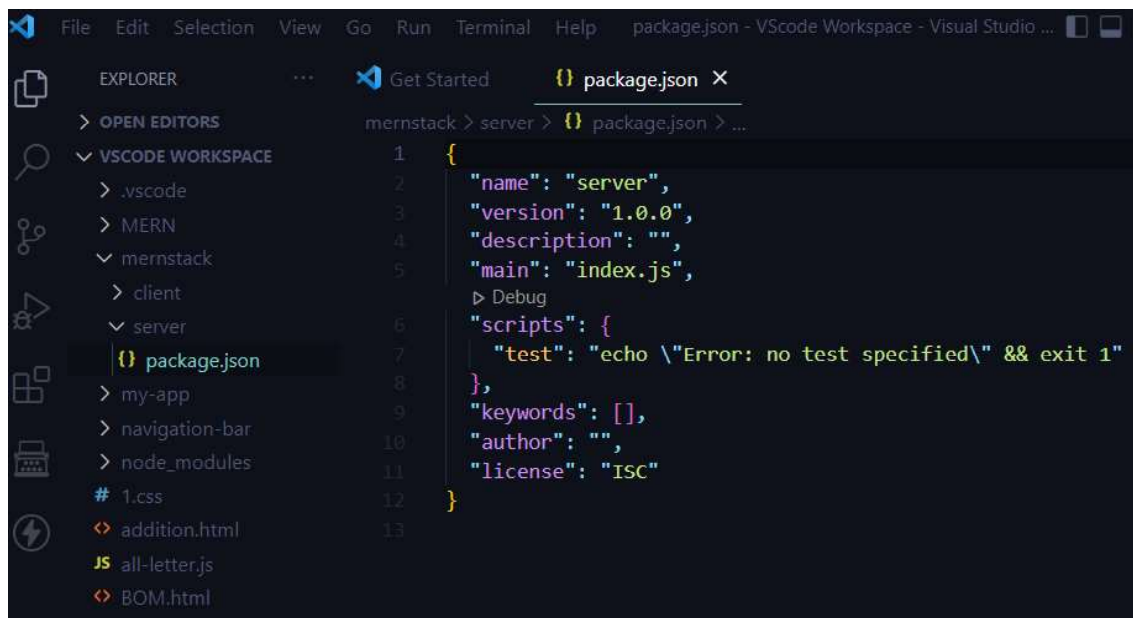
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace> cd mernstack
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack> cd server
PS E:\1. AIR UNIVERSITY\Spring-2022\FSWD\VScode Workspace\mernstack\server>
```

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\FSD\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

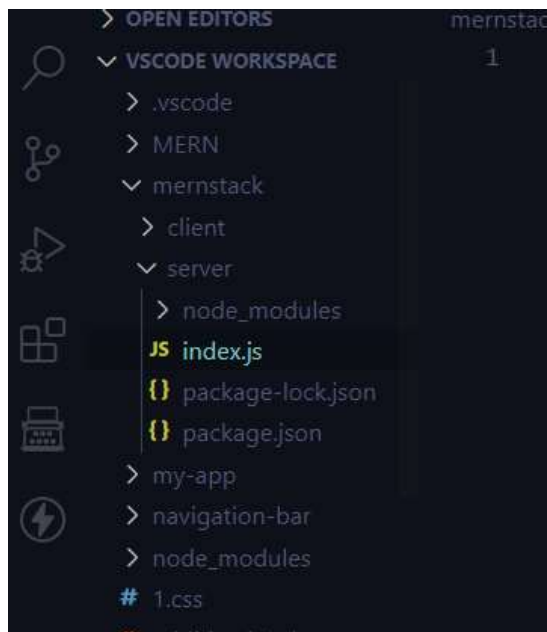
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

"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\FSD\VScode Workspace\mernstack\server>
PS E:\1. AIR UNIVERSITY\Spring-2022\FSD\VScode Workspace\mernstack\server> npm install express mongoose cors nodemon
[.....] | idealTree:server: $!!! idealTree buildDeps

Ln 1, Col 1 Spaces: 2
```

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

json

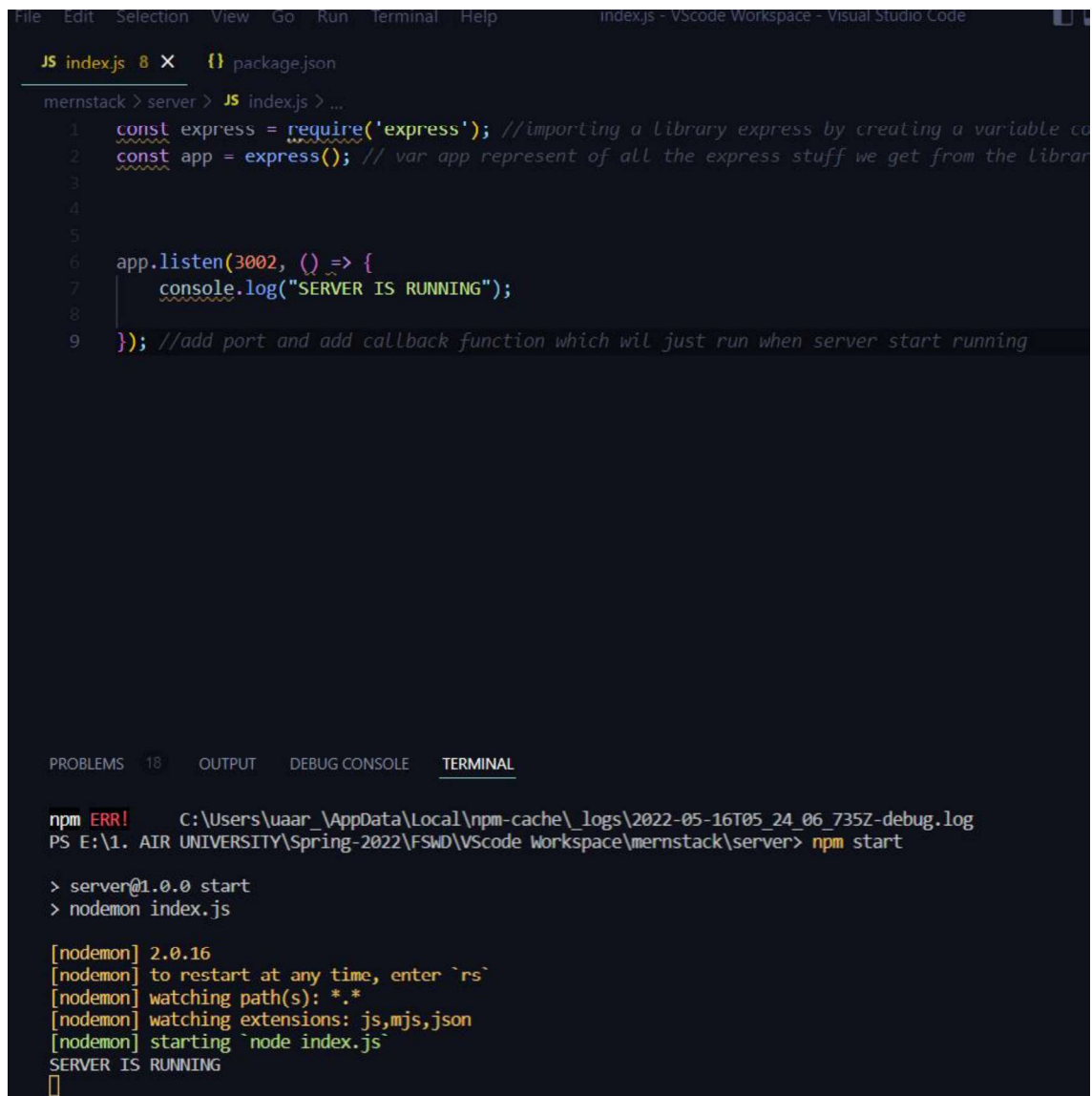
html

html

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
```



The image shows a Visual Studio Code editor window with a file named `index.js` open. The code in the editor is as follows:

```
1  const express = require('express'); //importing a library express by creating a variable co
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  }); //add port and add callback function which wil just run when server start running
```

Below the editor, the **TERMINAL** tab is active, showing the following output:

```
npm ERR! C:\Users\vaar\AppData\Local\npm-cache\_logs\2022-05-16T05_24_06_735Z-debug.log
PS E:\1. AIR UNIVERSITY\Spring-2022\FSD\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

DEPLOYMENT

Database

ta Lake

DATA SERVICES

ggers

ta API PREVIEW

SECURITY

ickstart

atabase Access

twork Access

vanced

w On Atlas 1

Find a database deployment...

Cluster0

Connect

View Monitoring

Browse Collections

...

R 0

W 0

Last 2 hours

100.0/s

Connections 0

Last 2 hours

100.0

In 0.0 B/s

Out 0.0 B/s

Last 2 hours

100.0 B/s

VERSION	REGION	CLUSTER TIER	TYPE
5.0.8	AWS / N. Virginia (us-east-1)	M0 Sandbox (General)	Replica Set - 3 nodes

Click on “Add your IP address”

access your cluster now. [Read more](#)

You can't connect yet. Set up your firewall access and user security permission below.

1 Add a connection IP address

Add Your Current IP Address

Add a Different IP Address

Allow Access from Anywhere

2 Create a Database User

This first user will have [atlasAdmin](#) permissions for this project.

Keep your credentials handy, you'll need them for the next step.

Username

ex. dbUser

Password

ex. dbUserPassword

SHOW

Autogenerate Secure Password

Copy

Create Database User

You can't connect yet. Set up your firewall access and user security permission below.

1 Add a connection IP address

IP Address	Description (Optional)
<input type="text" value="43.245.8.105"/>	<input type="text" value="An optional comment describing this en"/>
<div>Cancel Add IP Address</div>	

2 Create a Database User

This first user will have [atlasAdmin](#) permissions for this project.

Keep your credentials handy, you'll need them for the next step.

Username

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

You can't connect yet. Set up your user security permission below.

1 Add a connection IP address

✓ An IP address has been added to the IP Access List. [Add another address in the IP Access List tab.](#)

2 Create a Database User

This first user will have [atlasAdmin](#) permissions for this project.

Keep your credentials handy, you'll need them for the next step.

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

2 Create a Database User

This first user will have [atlasAdmin](#) permissions for this project.

Keep your credentials handy, you'll need them for the next step.

Username

Password

 HIDE 🔍 Autogenerate Secure Password 📋 Copy

Create Database User

Close

Choose a connection method

After creating username and password, click chosen your connection method

Connect to Cluster0

Setup connection security

Choose a connection method

Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

You're ready to connect. Choose how you want to connect in the next step.

1 Add a connection IP address

✓ An IP address has been added to the IP Access List. *Add another address in the [IP Access List](#) tab.*

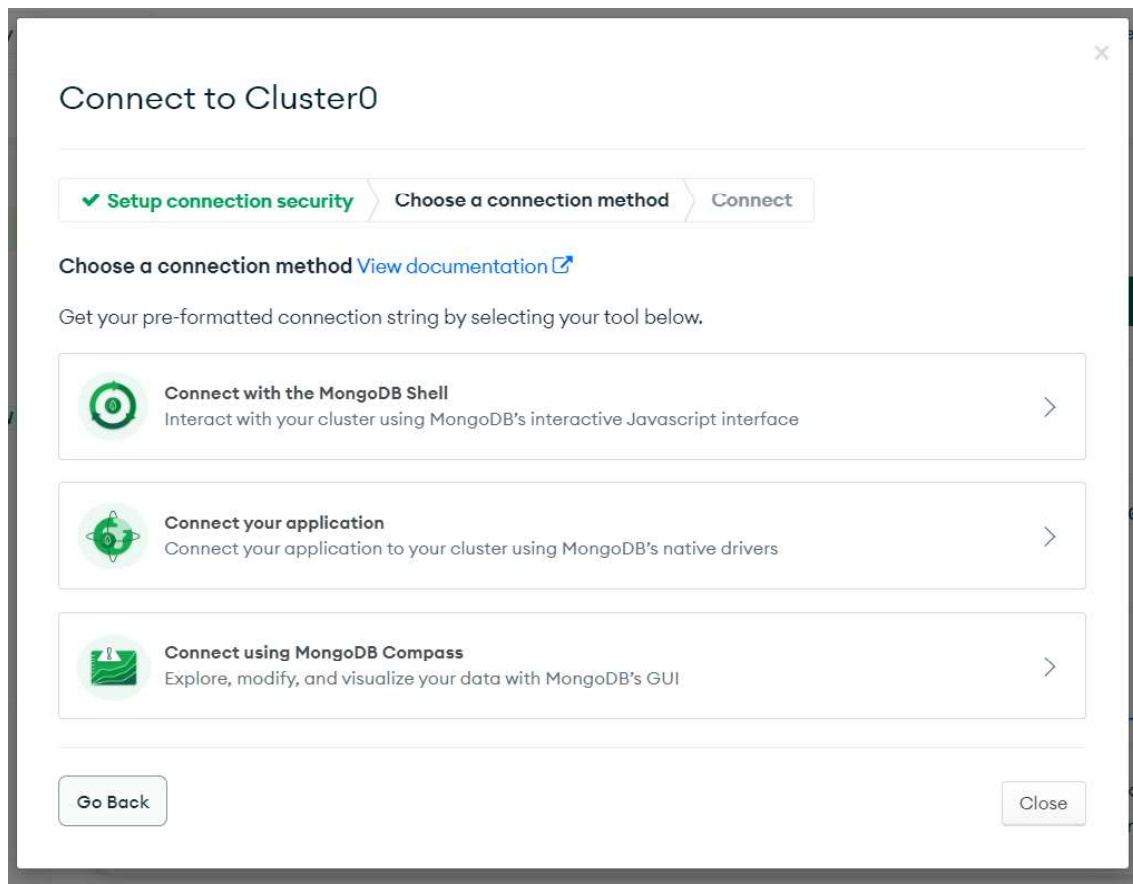
2 Create a Database User

✓ A MongoDB user has been added to this project. *Not yours? Create one in the [MongoDB Users](#) tab.*

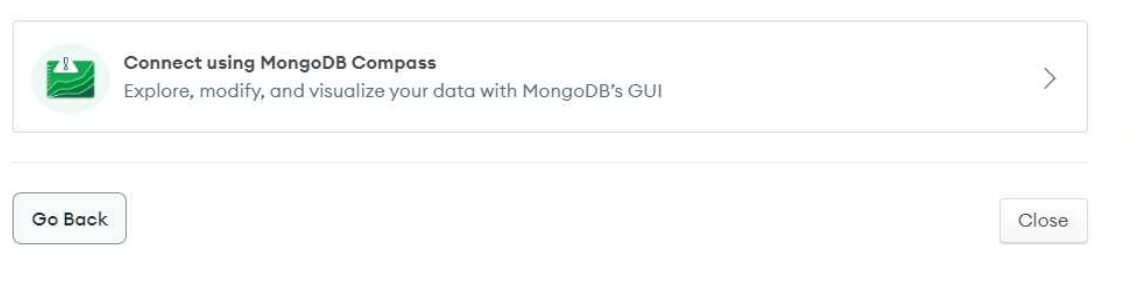
You'll need your MongoDB user's credentials in the next step.

Close

Choose a connection method



Click on connect using MongoDB Compass



Click on I have MongoDB Compass and copy the connection string

Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

I do not have MongoDB Compass

I have MongoDB Compass

1 Choose your version of Compass:

1.12 or later

See your Compass version in "About Compass"

2 Copy the connection string, then open MongoDB Compass.

mongodb+srv://user123:<password>@cluster0.ygqyq.mongodb.net/test

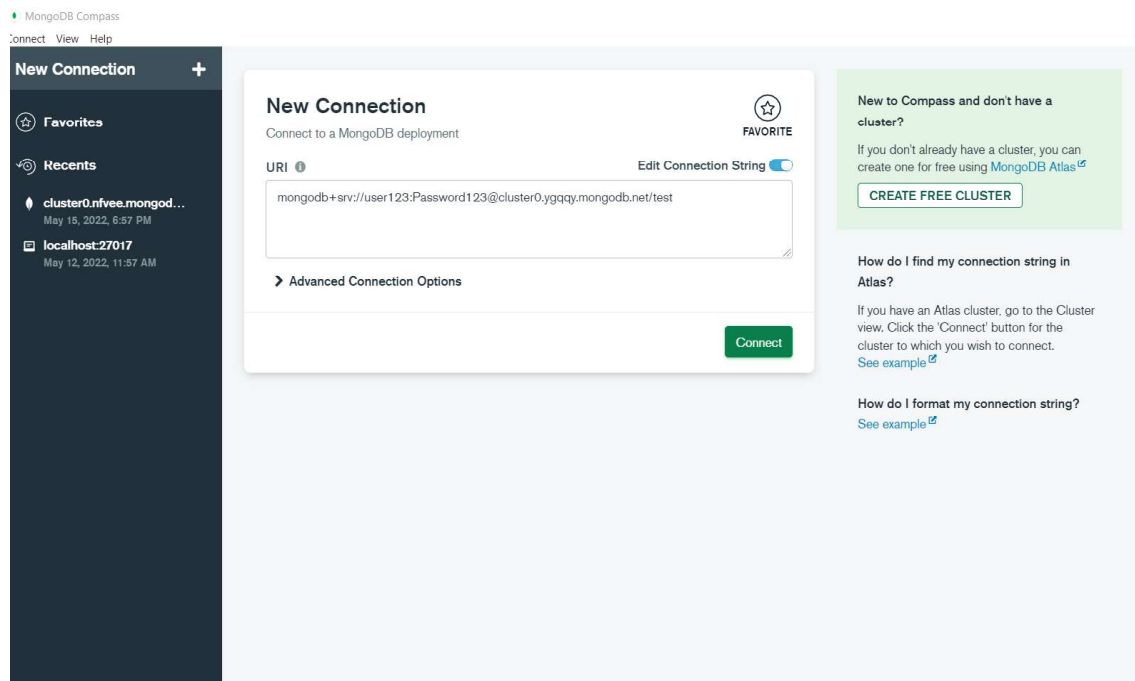
You will be prompted for the password for the **user123** user's (Database User) username. When entering your password, make sure that any special characters are [URL encoded](#).

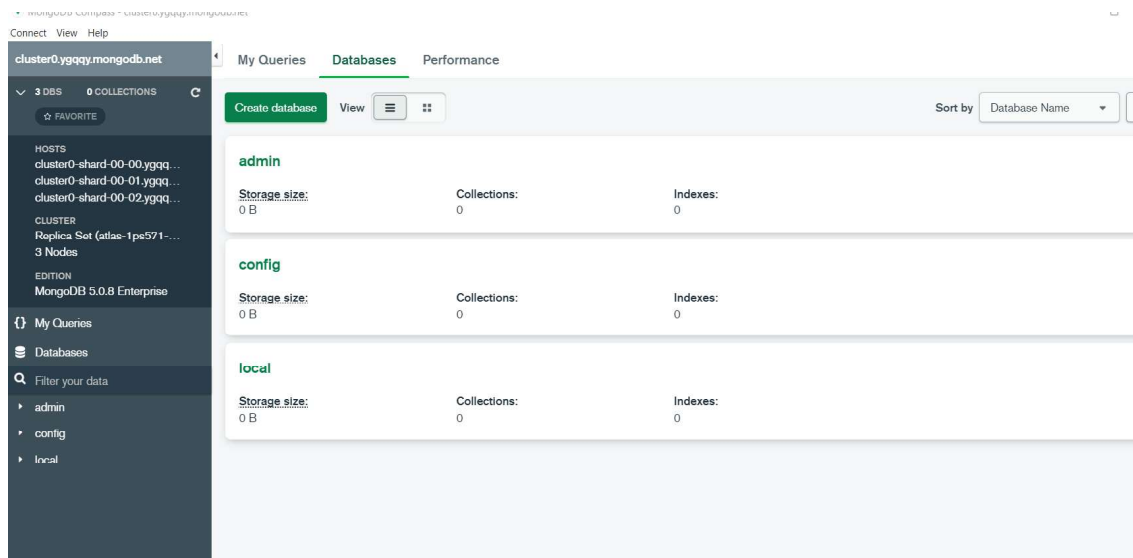
Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

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



Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

1 Select your driver and version

DRIVER

Node.js

VERSION

4.0 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://user123:
<password>@cluster0.ygqqy.mongodb.net/myFirstDatabase?
retryWrites=true&w=majority
```



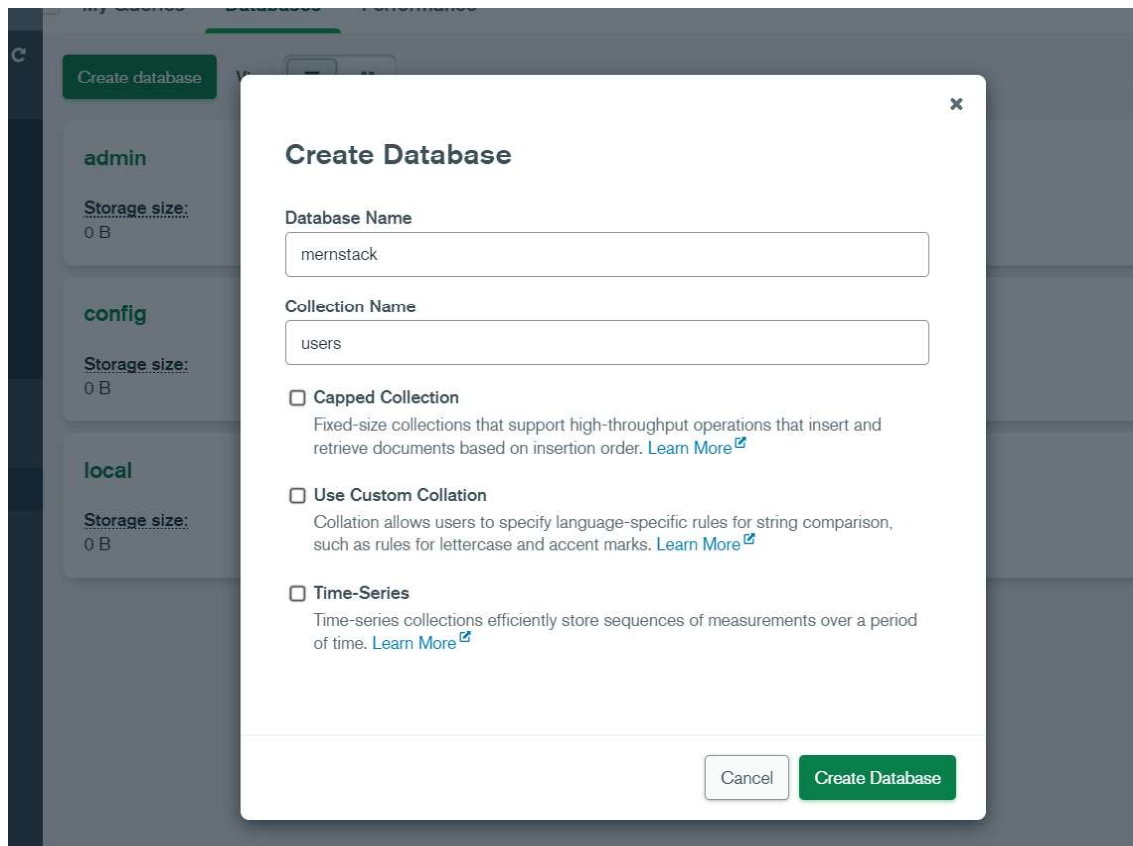
Replace **<password>** with the password for the **user123** user. Replace **myFirstDatabase** with the name of the database that connections will use by default. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

Then open MogoDB compass and create a Database



Connect to Cluster0

✓ Setup connection security

Choose a connection method

Connect

Choose a connection method [View documentation](#)

Get your pre-formatted connection string by selecting your tool below.



Connect with the MongoDB Shell

Interact with your cluster using MongoDB's interactive Javascript interface



Connect your application

Connect your application to your cluster using MongoDB's native drivers



Connect using MongoDB Compass

Explore, modify, and visualize your data with MongoDB's GUI



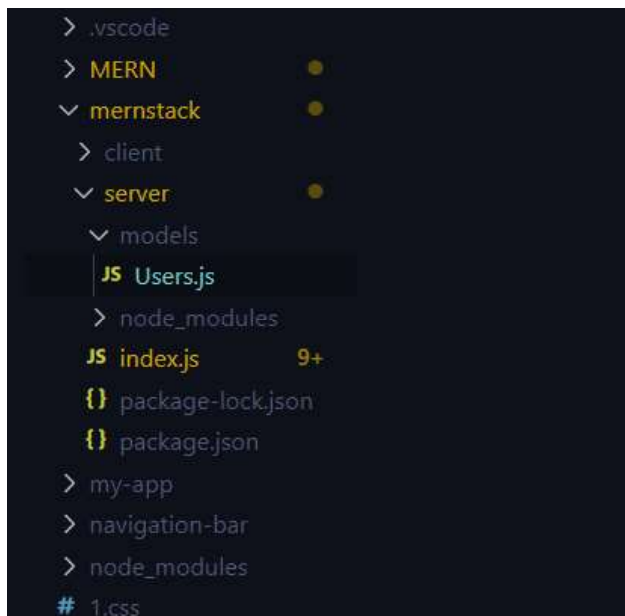
Go Back

Close

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

```
index.js 9+ x {} package.json
mernstack > server > JS index.js > ...
1 const express = require('express'); //importing a library express by creating a variable code express
2 const app = express(); // var app represent of all the express stuff we get from the library
3 const mongoose = require('mongoose');
4
5 mongoose.connect("mongodb+srv://user123:Password@cluster0.ygqyq.mongodb.net/mernstack?retryWrites=true&w=majority"); //represent the connect
6
7
8 app.listen(3002, () => {
9   console.log("SERVER IS RUNNING");
10
11 }); //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


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

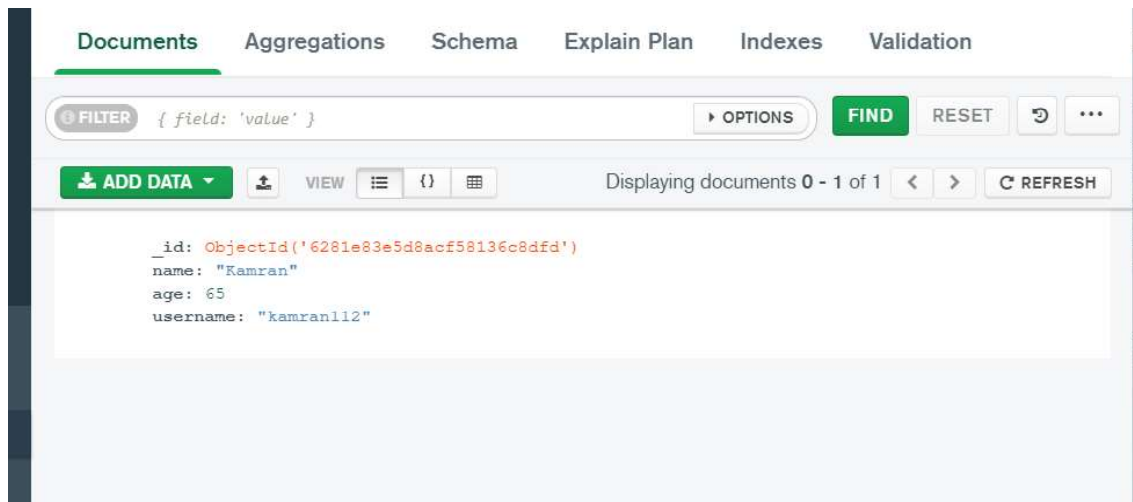
Import Data

Insert to Collection mernstack.users

VIEW {}

```
1 /**
2  * Paste one or more documents here
3  */
4 {
5   _id: {
6     $oid: "6281e83e5d8acf58136c8dfd"
7   },
8   name: "Kamran",
9   age: 65,
10  username: "kamran112"
11 }
```

Cancel Insert

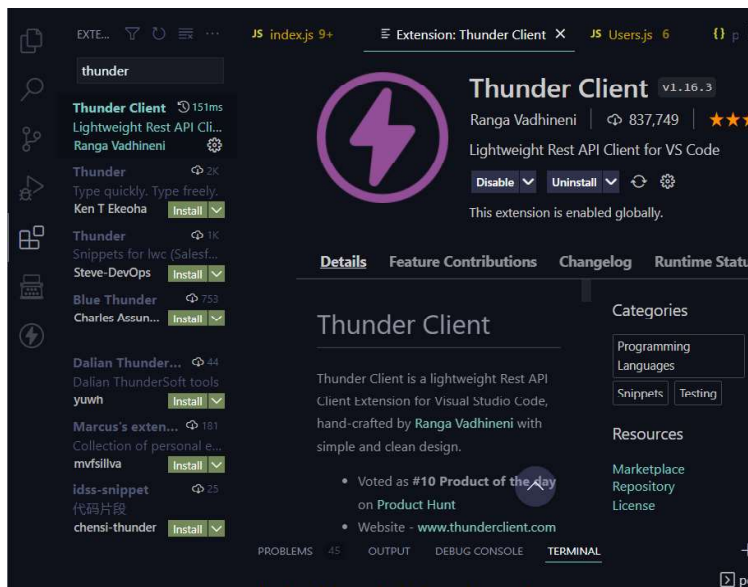


Now add in file “index.js”

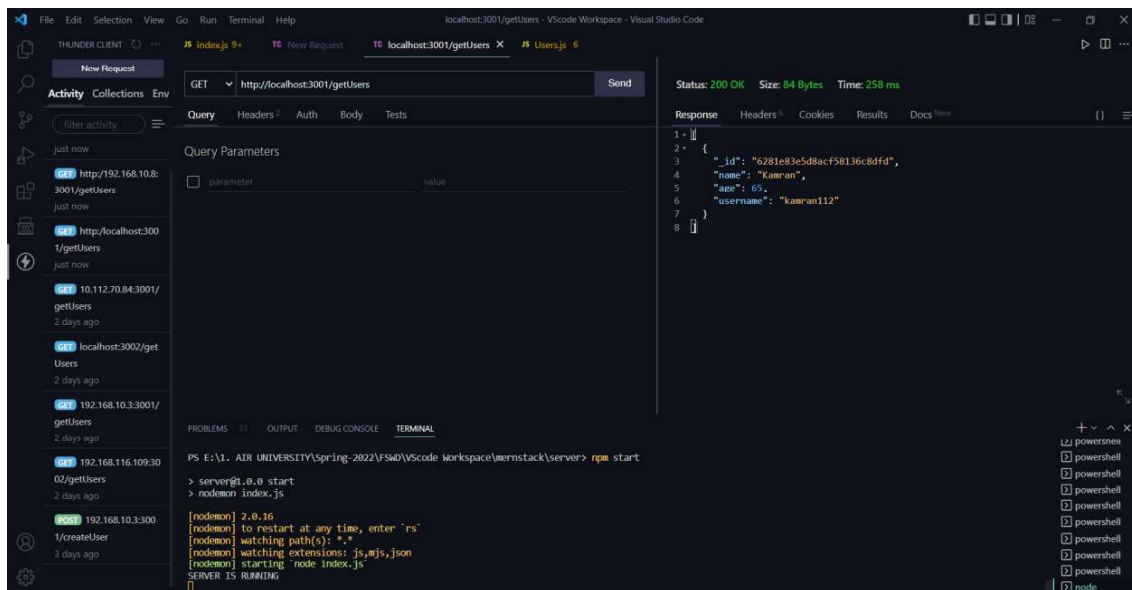
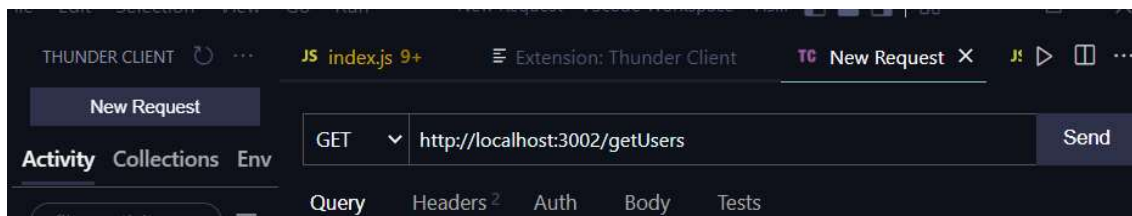
```
const UserModel = require ('./models/Users');
```

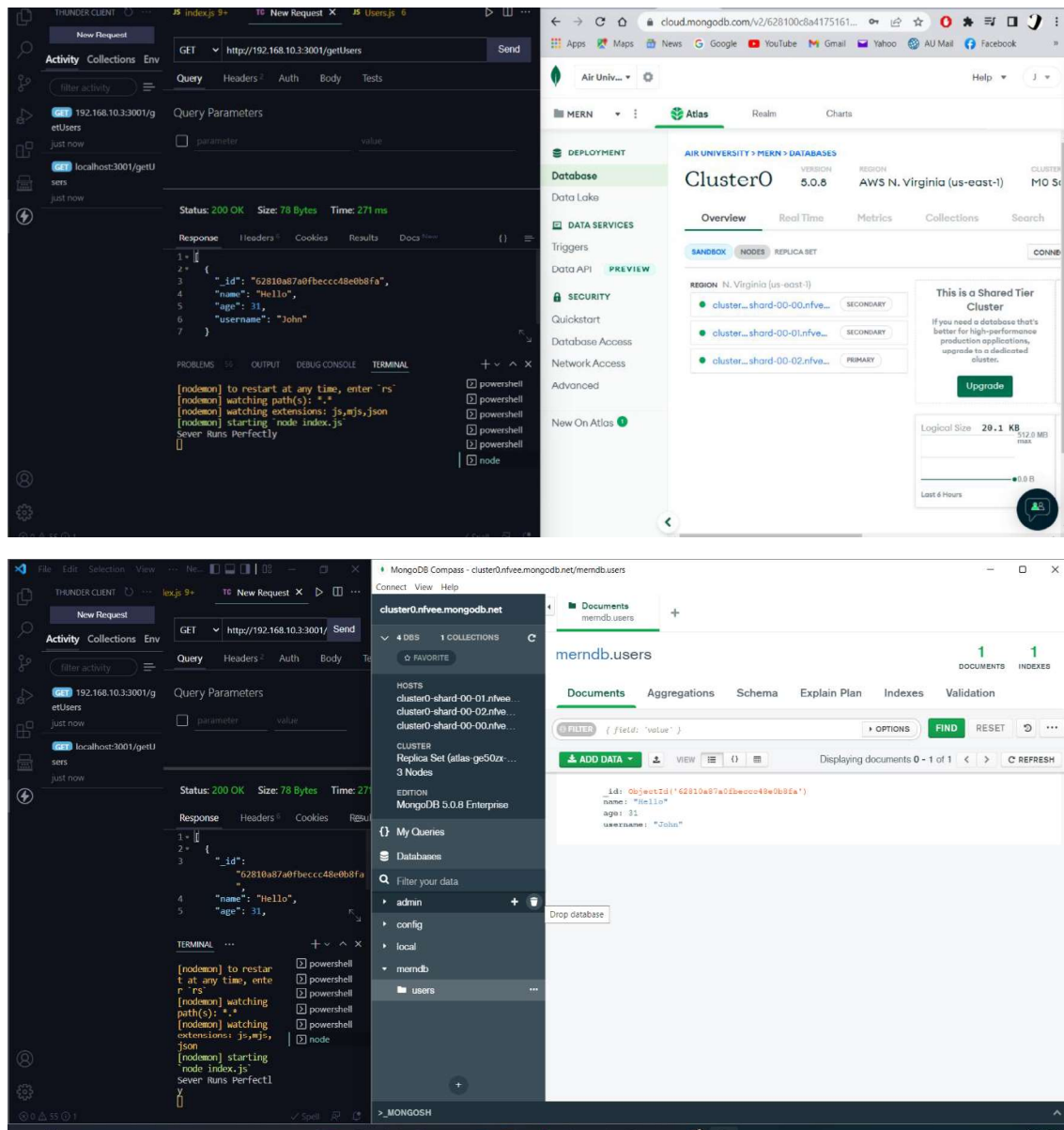
```
app.get("/getUsers", (req, res) => {  
  UserModel.find({}, (err, result) => {  
    if(err){  
      res.json(err);  
    }  
    else{  
      res.json(result);  
    }  
  });  
});
```

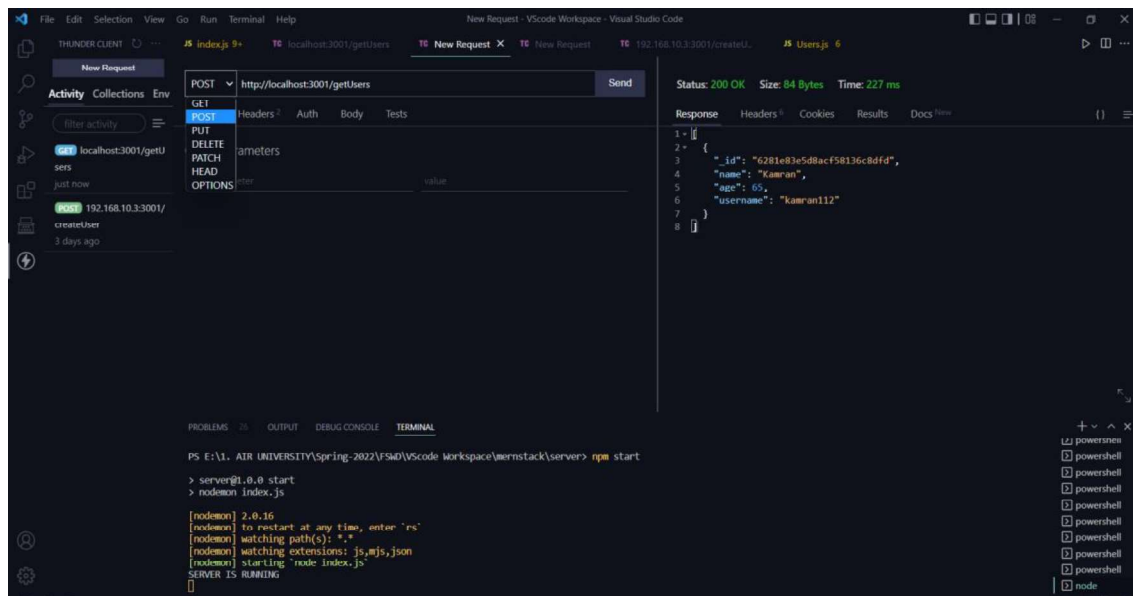
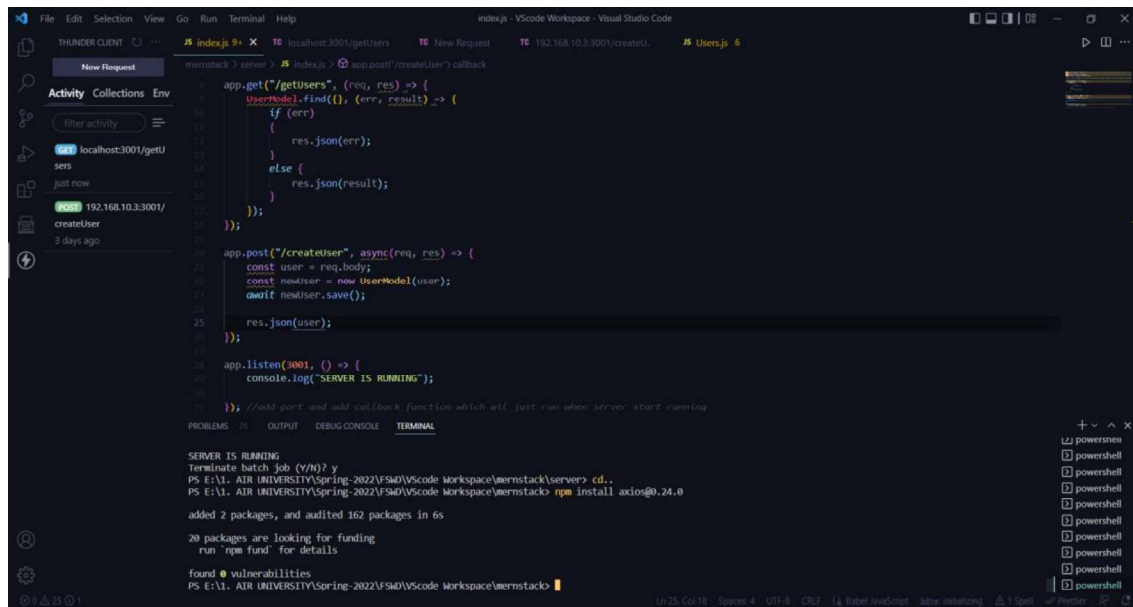
Add this extension “Thunder Client”

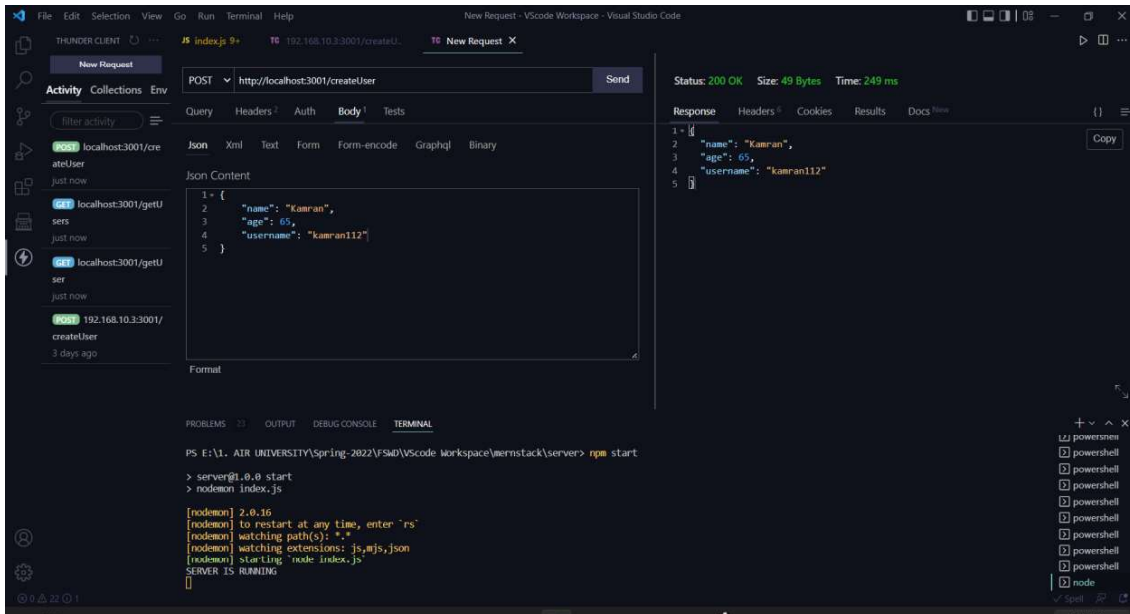
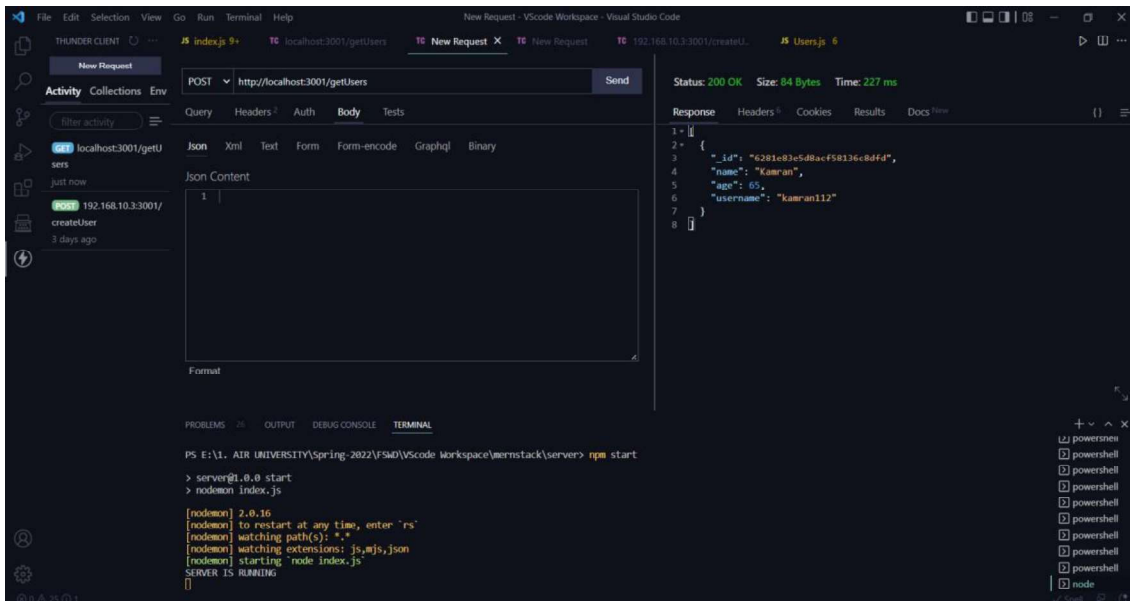


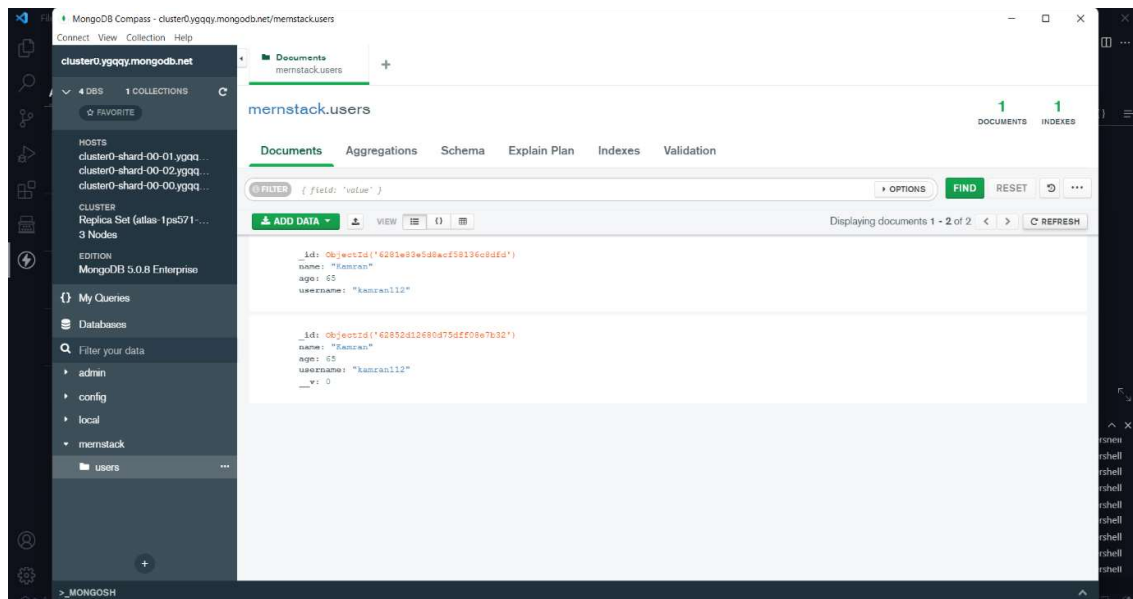
Open Thunder client and click “New Request”











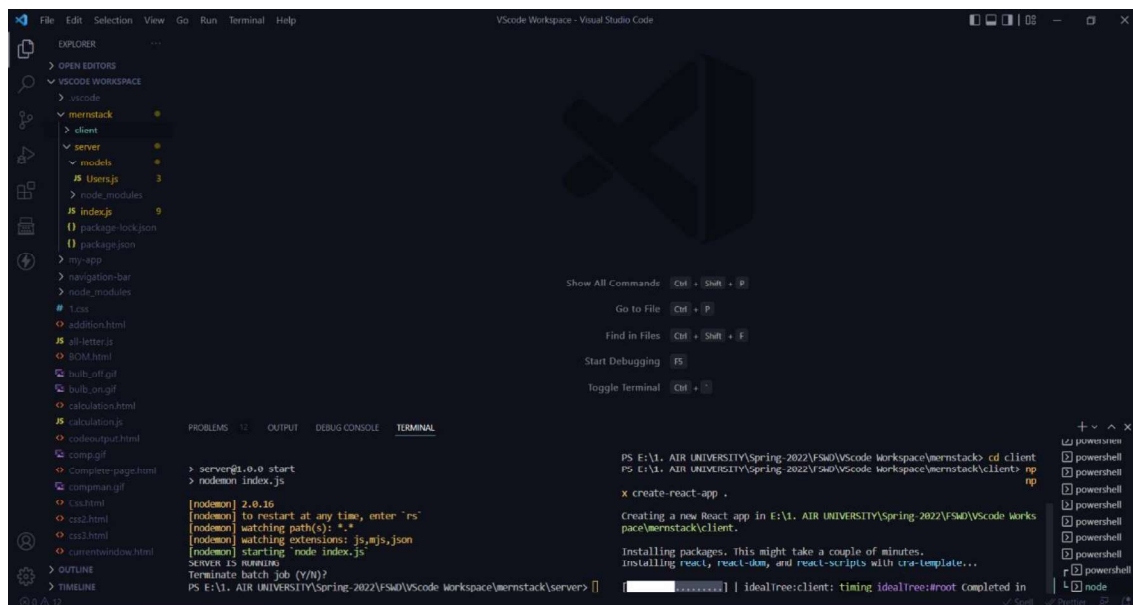
Client

New terminal

Create react app in client

cd client

npx create-react-app .



Install axios

```

{
  "name": "client",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.4",
    "@testing-library/react": "^13.2.0",
    "@testing-library/user-event": "^13.5.0",
    "react": "^18.1.0",
    "react-dom": "^18.1.0",
    "react-scripts": "5.0.1",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  }
}

```

```

> nodeemon 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
Terminate batch job (Y/N)?
PS E:\1. AIR UNIVERSITY\Spring-2022\F540\VS code Workspace\mernstack\server>

```

```

npm run eject
Removes this tool and copies build dependencies, configuration files
and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

cd E:\1. AIR UNIVERSITY\Spring-2022\F540\VScode workspace\mernstack\client
npm start

Happy hacking!
PS E:\1. AIR UNIVERSITY\Spring-2022\F540\VScode workspace\mernstack\client> npm install axios

```

After instaling

Delete 4 files

```

// jest-dom adds custom jest matchers for asserting on DOM nodes.
// allows you to do things like:
// expect(element).toHaveTextContent(/react/i)
// learn more: https://github.com/testing-library/jest-dom
import '@testing-library/jest-dom';

```

```

> nodeemon 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
Terminate batch job (Y/N)?
PS E:\1. AIR UNIVERSITY\Spring-2022\F540\VS code Workspace\mernstack\server>

```

```

Compiled successfully!

You can now view client in the browser.

Local:    http://localhost:3000
On Your Network: http://192.168.10.8:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully

```

Write in app.js

```

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

function App() {

```

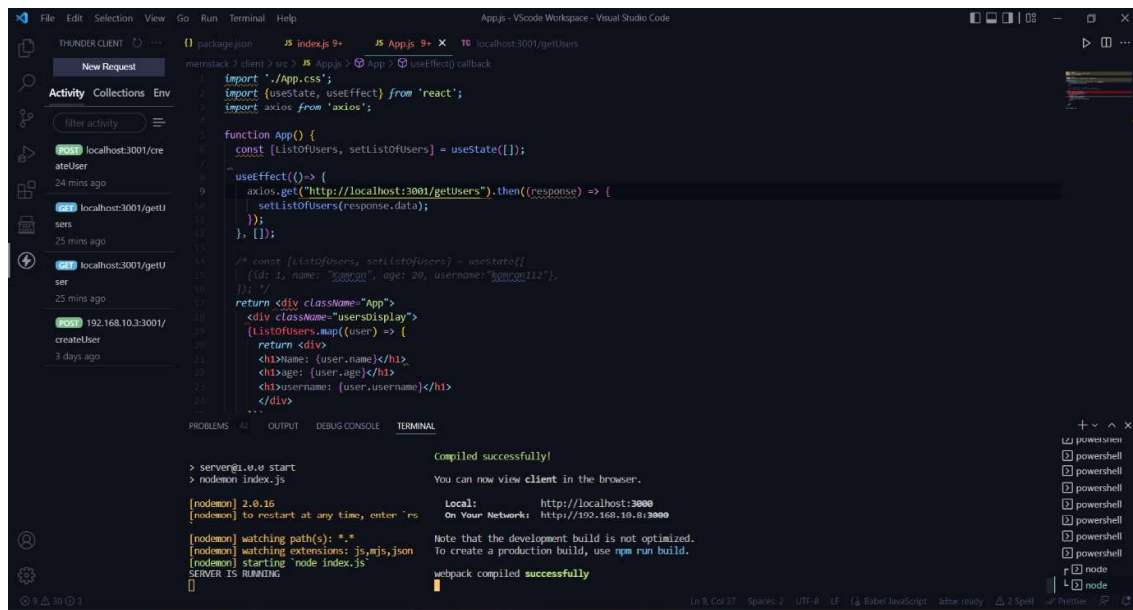
```

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>
}

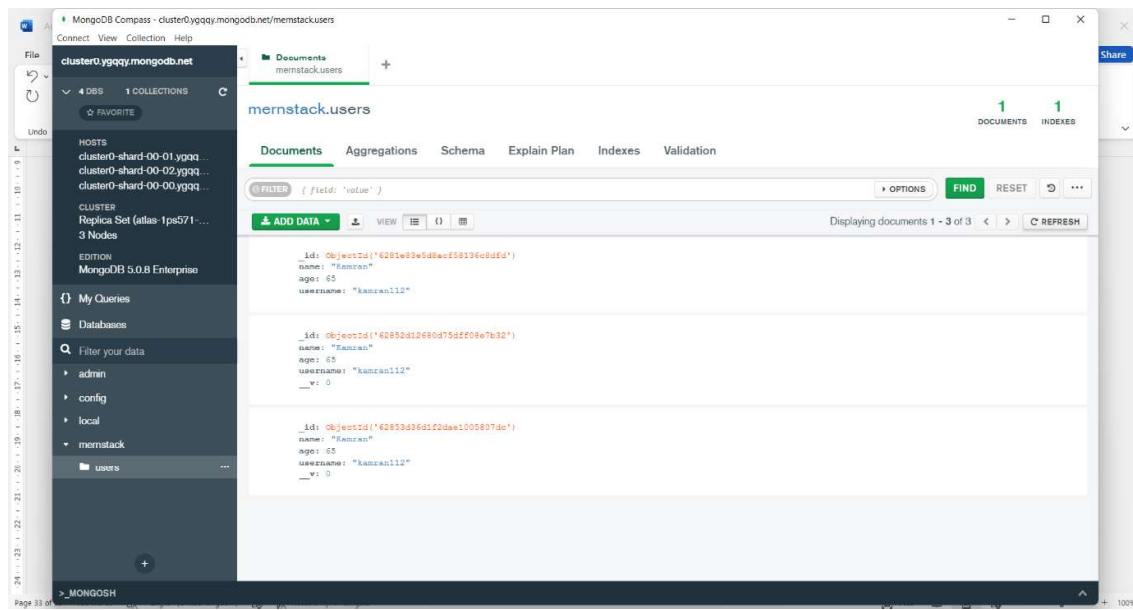
export default App;

```

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
---------	--------	-------------	-------------

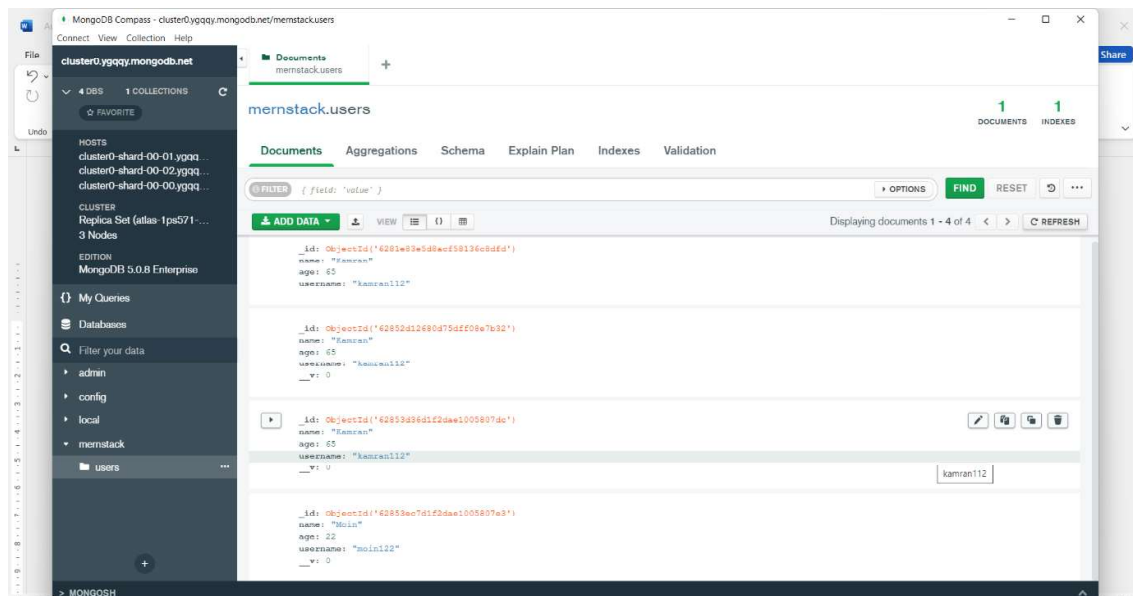


Add this for autoupdate

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

age: 65
 username: kamran112
 Name: Kamran
 age: 65
 username: kamran112
 Name: Kamran
 age: 65
 username: kamran112
 Name: Moin
 age: 22
 username: moin122

Moin 22 moin122 Create User



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;

```