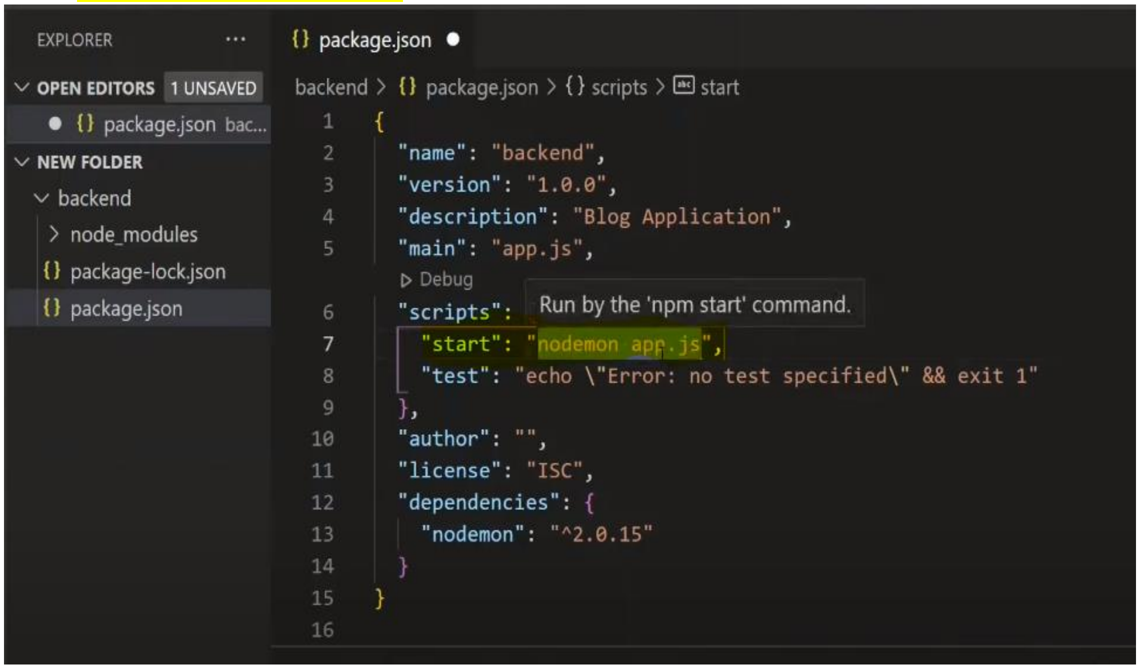
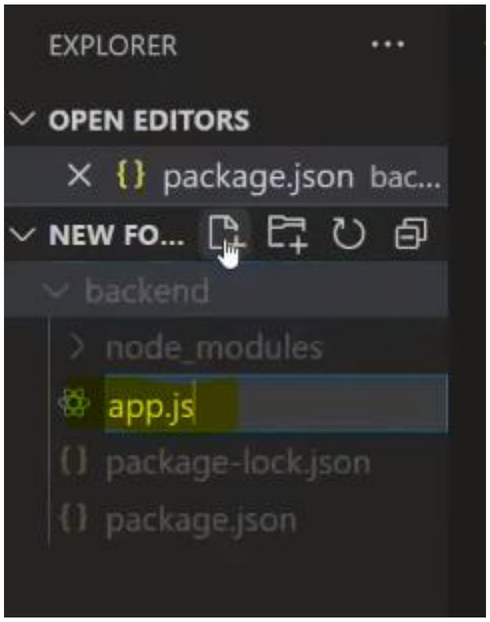
Project

1. First create a separate folder for front end by using React.
   a. Run **npx create-react-app frontend** to create the react frontend application
2. Create another folder for backend.
   a. Run **npm init** command to create the **package.json** file.
   b. Run **npm i nodemon** command to create the **node modules** folder.
   c. Write **"start" : "nodemon app.js"** inside of the **package.json** in **scripts** section



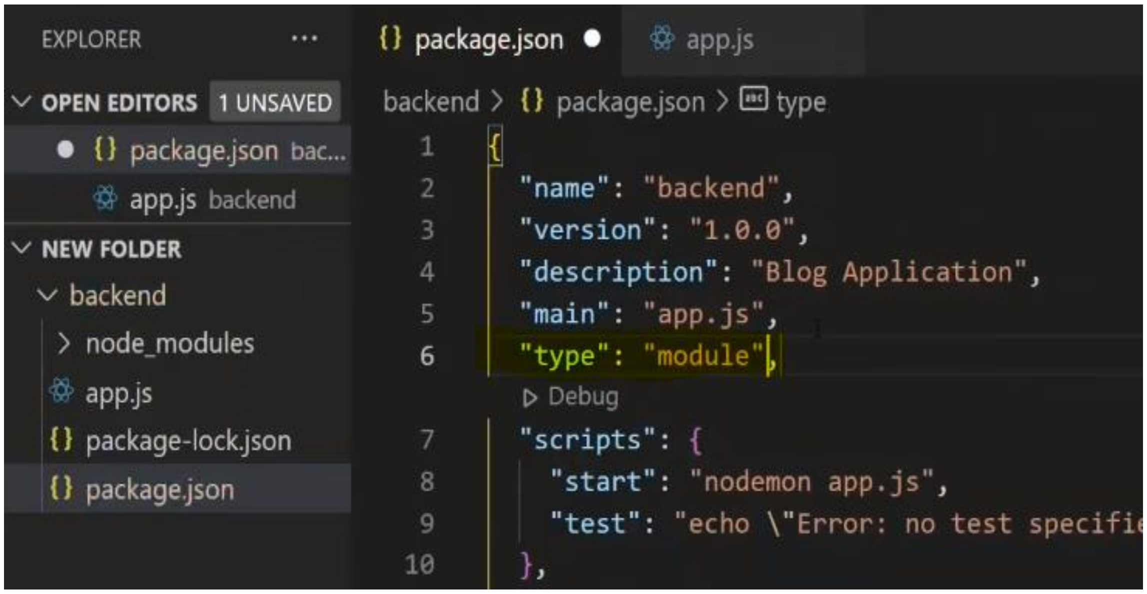
```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "Blog Application",
  "main": "app.js",
  "scripts": {
    "start": "nodemon app.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "nodemon": "^2.0.15"
  }
}
```

   d. Create one **app.js** file in backend folder



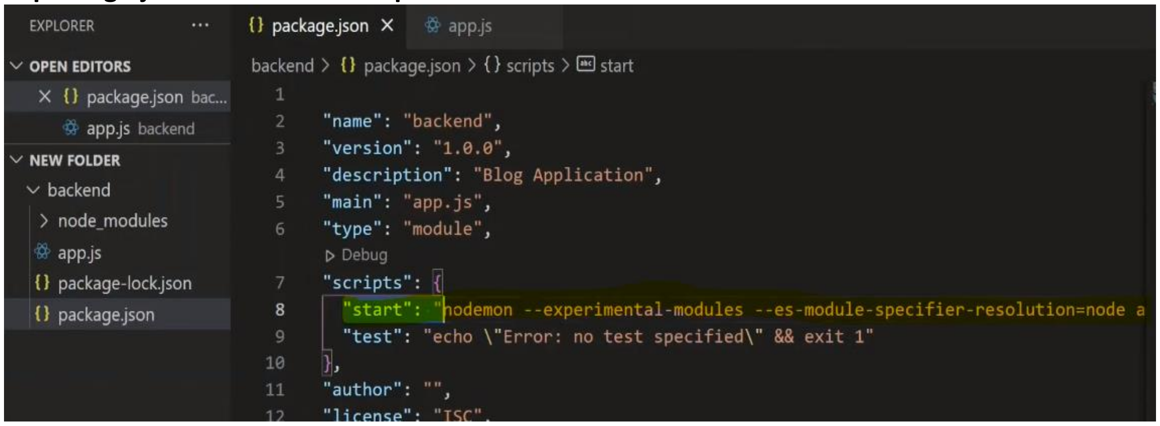
   e. Run **npm i express** to install express in the project

f. Add **“type” : “module”** in **package.json** file



```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "Blog Application",
  "main": "app.js",
  "type": "module",
  "scripts": {
    "start": "nodemon app.js",
    "test": "echo \"Error: no test specifie
  },
```

g. Add `--experimental-modules --es-module-specifier-resolution=node` in **package.json** inside of the **scripts** section in **start**

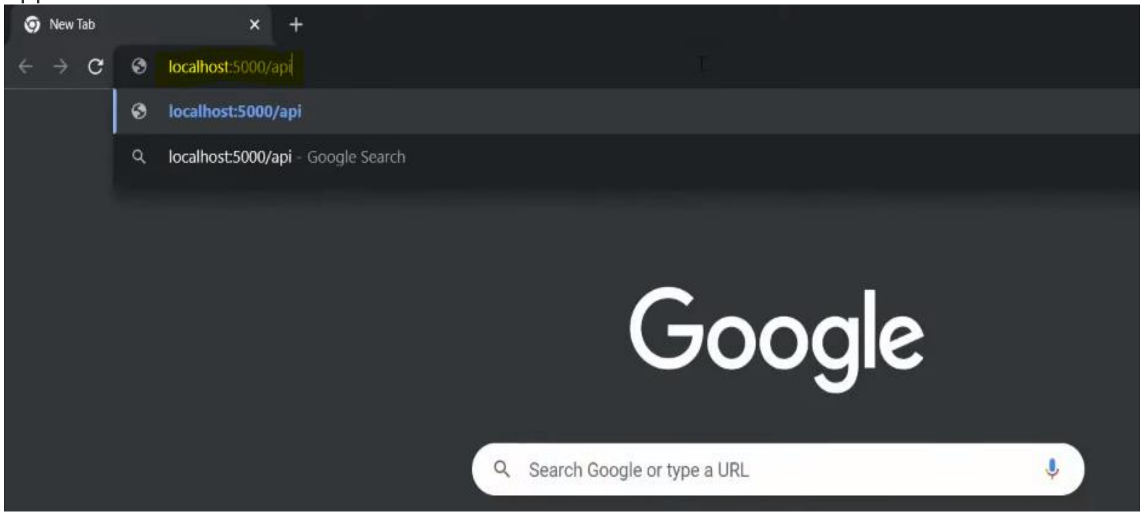


```

  "name": "backend",
  "version": "1.0.0",
  "description": "Blog Application",
  "main": "app.js",
  "type": "module",
  "scripts": {
    "start": "nodemon --experimental-modules --es-module-specifier-resolution=node a
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
```

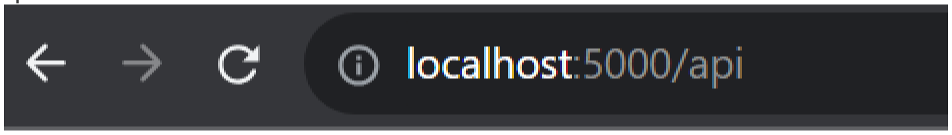
h. Write the following code in **app.js** to run the application in the browser.

```
import  express  from "express";

const app = express();
app.use("/api", (req, res, next)=>{
    res.send("hi hello")
  })

app.listen(5000)
```

i. Now open Chrome and enter **localhost/5000/api** to run the application.


New Tab
localhost:5000/api
localhost:5000/api
localhost:5000/api - Google Search
Google
Search Google or type a URL

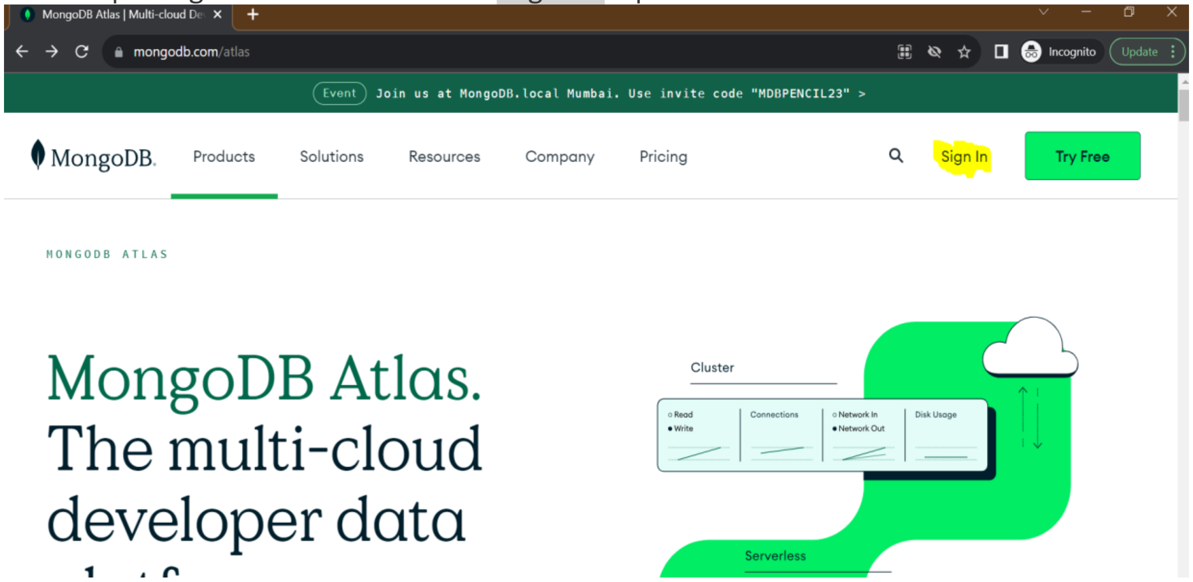
j. Output will be like this



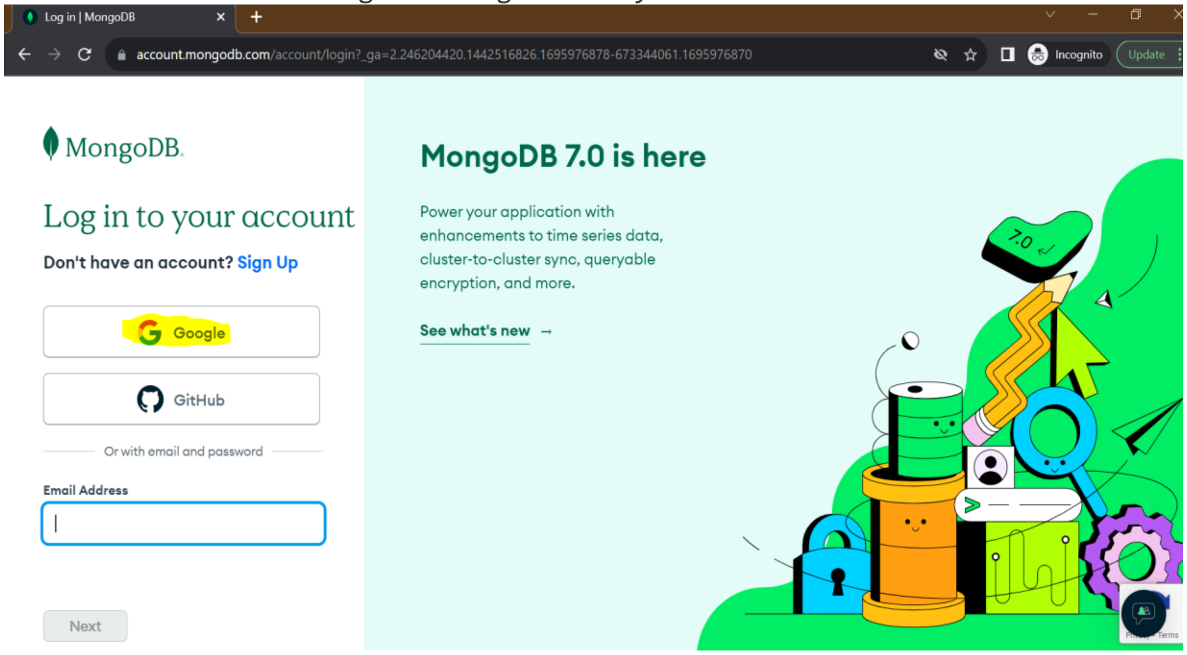
hi hello

k. Now we need to create an account in MongoDB Atlas. https://www.mongodb.com/atlas

l. After opening the link click on Sign In option



m. After that click on Google to login with your Gmail


Log in | MongoDB
account.mongodb.com/account/login?_ga=2.246204420.1442516826.1695976878-673344061.1695976870
Incognito
Update
MongoDB.
Google
GitHub
Or with email and password
Email Address
Next
See what's new →

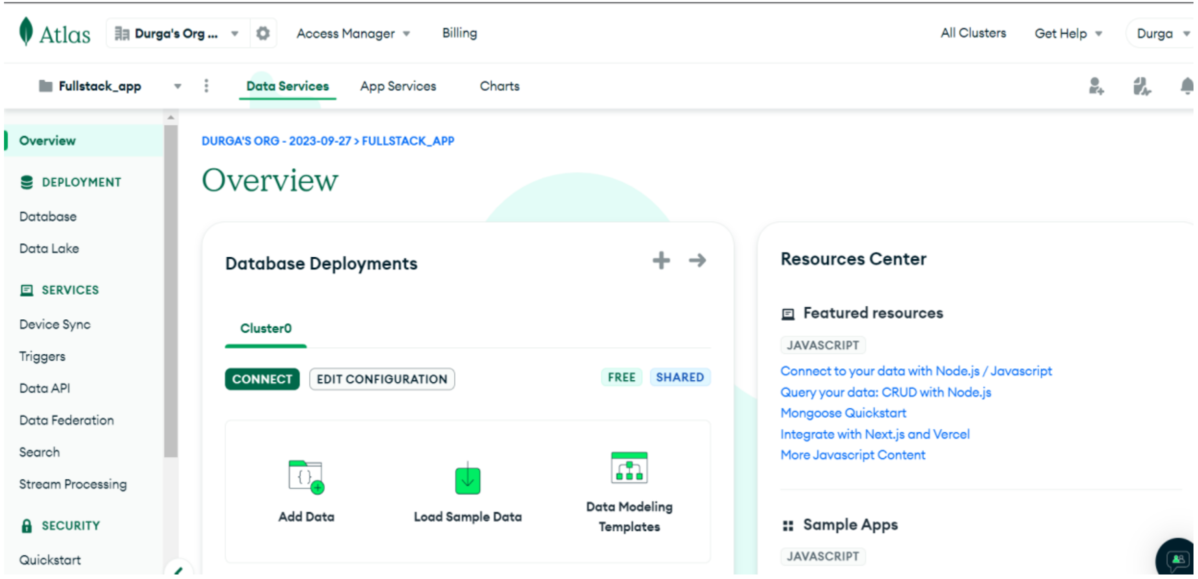
Log in to your account

Don't have an account? Sign Up

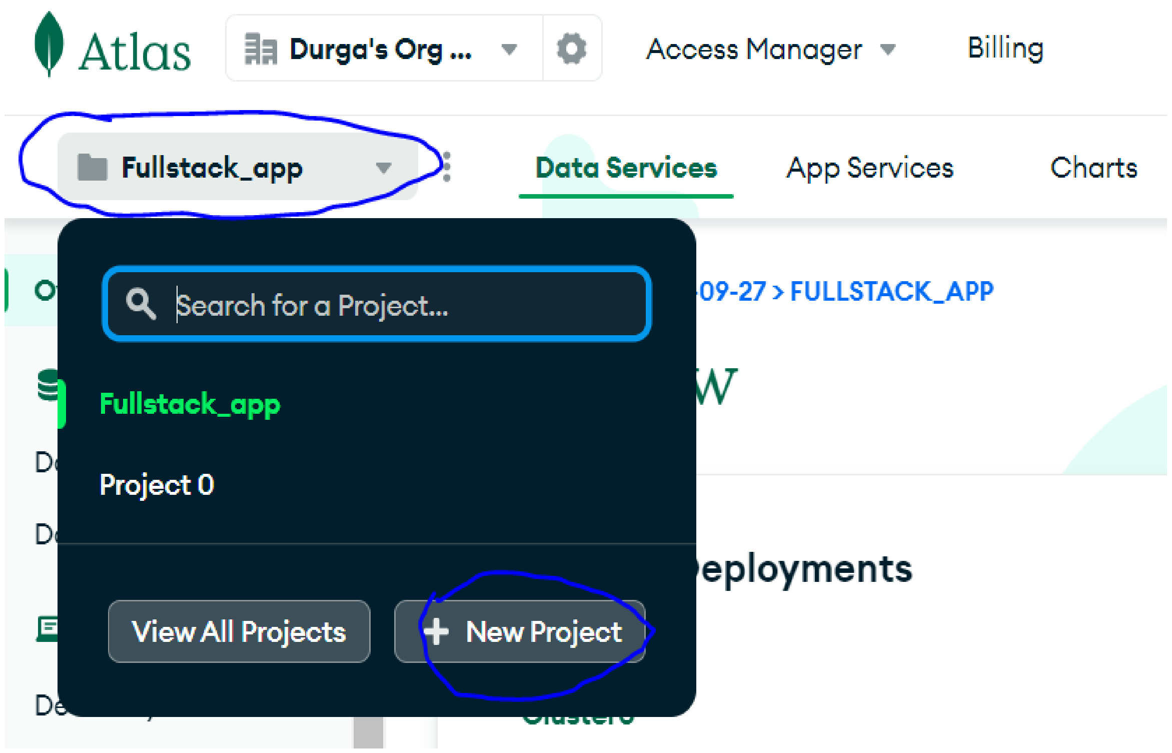
MongoDB 7.0 is here

Power your application with enhancements to time series data, cluster-to-cluster sync, queryable encryption, and more.

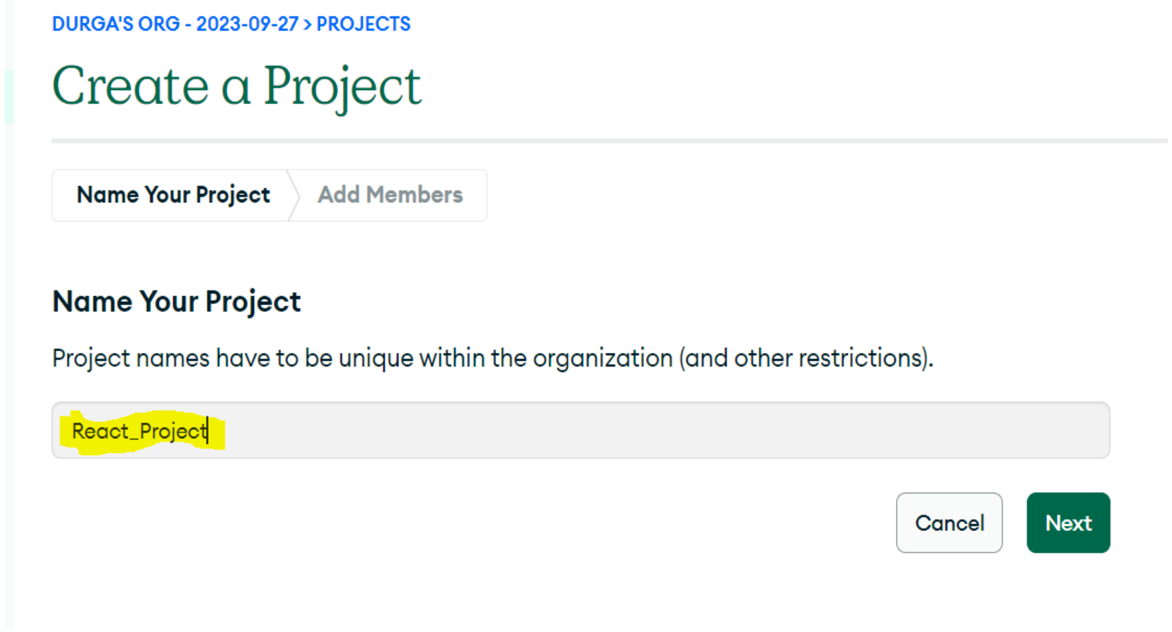
n. Once if the account is successfully created you will get the page like below.



o. Now in the top left side click on project button and then click on **New Project**



p. Enter any project name in the given input box and click on next button



DURGA'S ORG - 2023-09-27 > PROJECTS

Create a Project

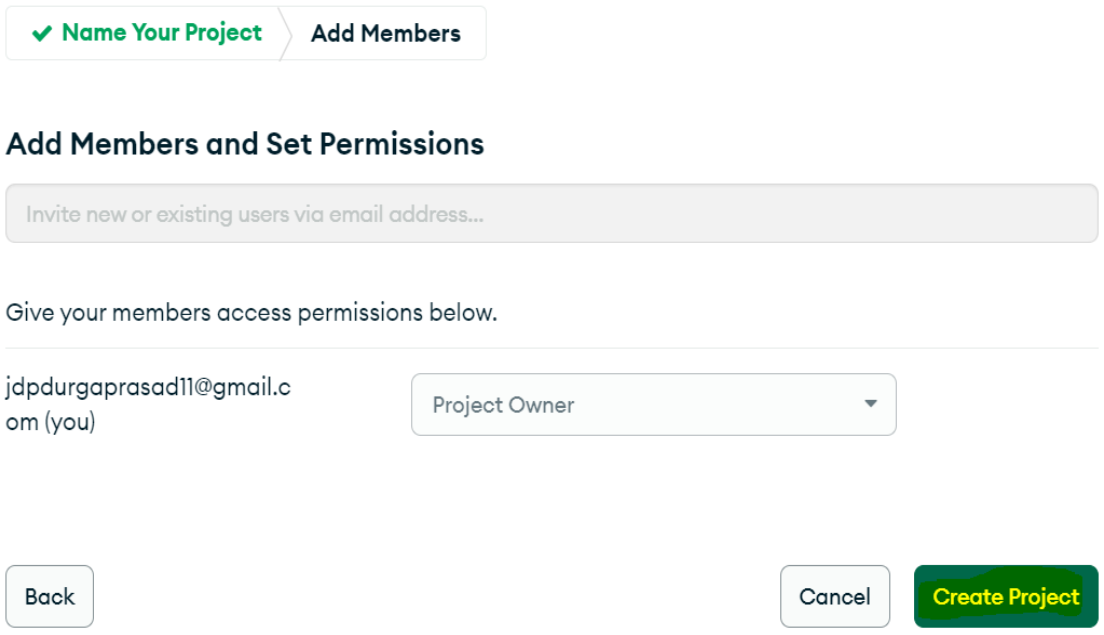
Name Your Project | Add Members

Name Your Project

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

React_Project

q. After that you will get the page like below no need to change anything in the below options, just click on **Create Project** button.



Name Your Project | Add Members

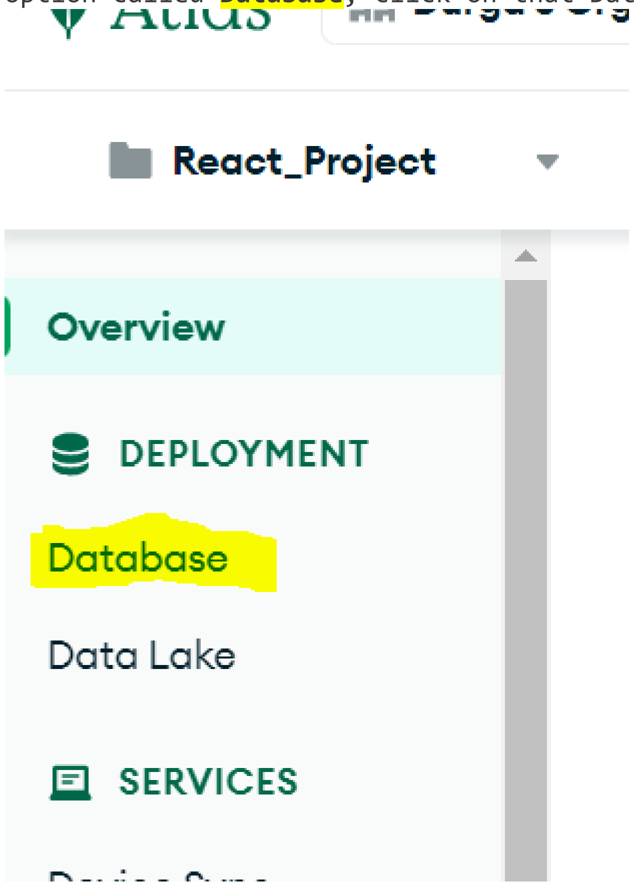
Add Members and Set Permissions

Invite new or existing users via email address...

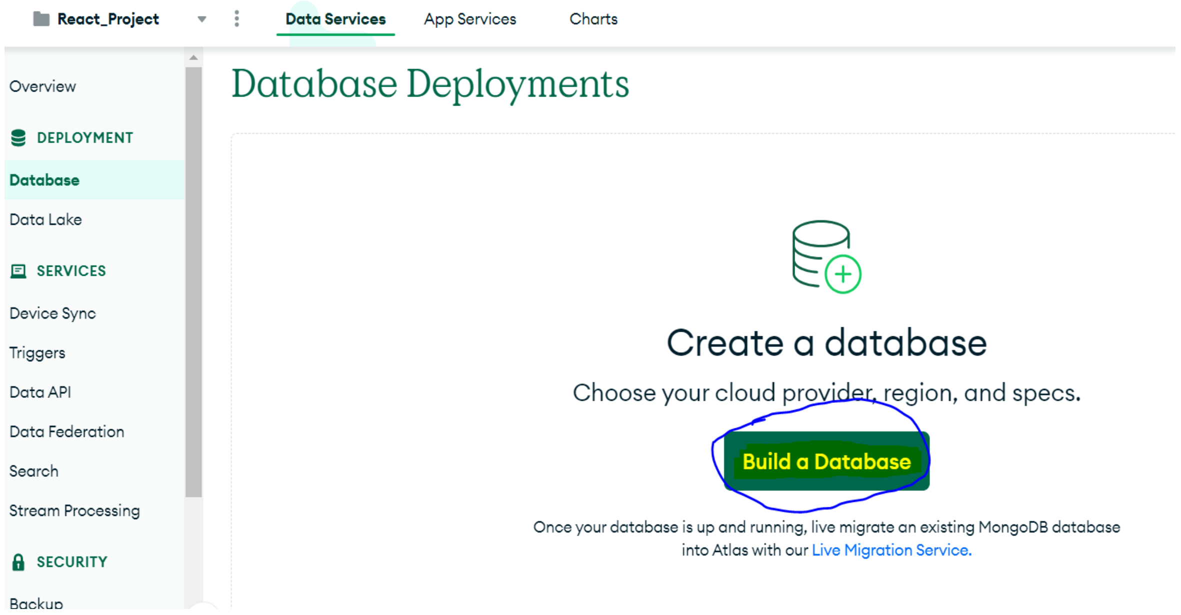
Give your members access permissions below.

jdpdurgaprasad11@gmail.com (you)	Project Owner

r. After that in the left side in Deployment section we have an option called Database, click on that Database option.



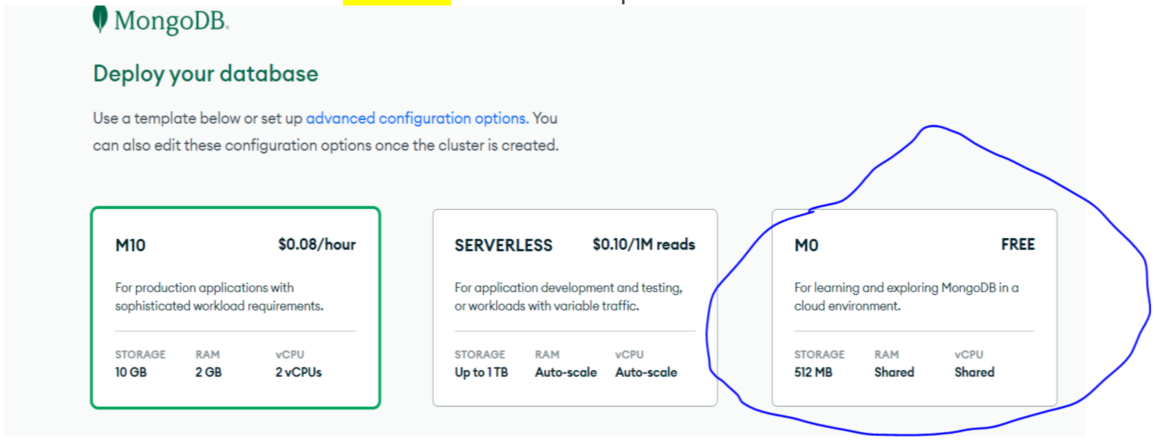
s. After that you have to click on **Build a Database** option.



Choose your cloud provider, region, and specs.

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

t. Now after that select **M0 Free** in the 3 options



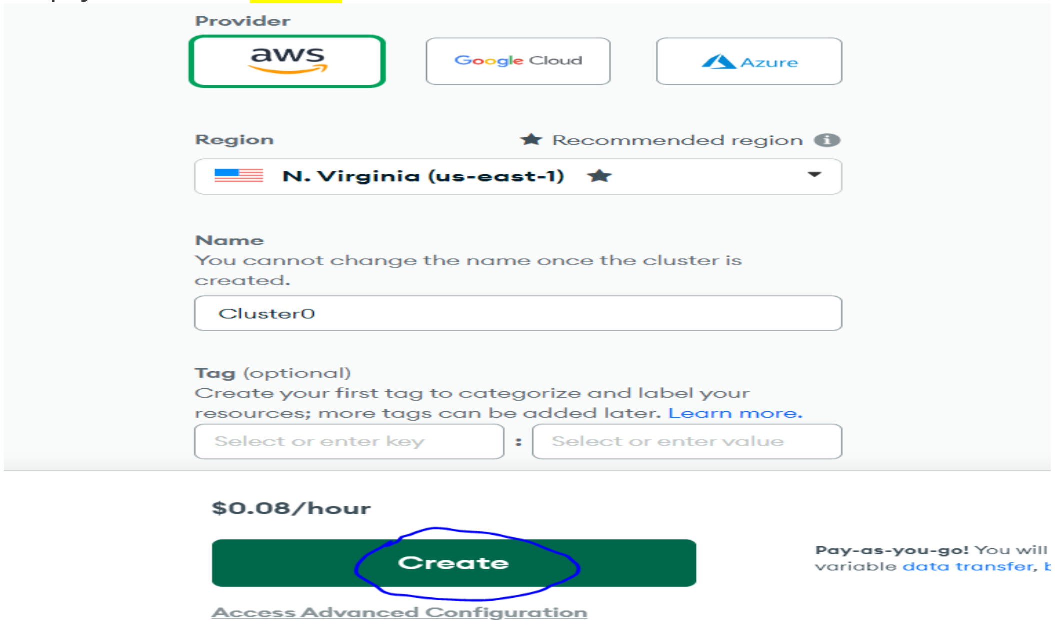
Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

M10 $0.08/hour
For production applications with sophisticated workload requirements.
STORAGE 10 GB RAM 2 GB vCPU 2 vCPUs

SERVERLESS $0.10/1M reads
For application development and testing, or workloads with variable traffic.
STORAGE Up to 1 TB RAM Auto-scale vCPU Auto-scale

M0 FREE
For learning and exploring MongoDB in a cloud environment.
STORAGE 512 MB RAM Shared vCPU Shared

u. After that no need to change anything in the below options, simply click on Create



Provider

Region ★ Recommended region

N. Virginia (us-east-1) ★

Name
You cannot change the name once the cluster is created.

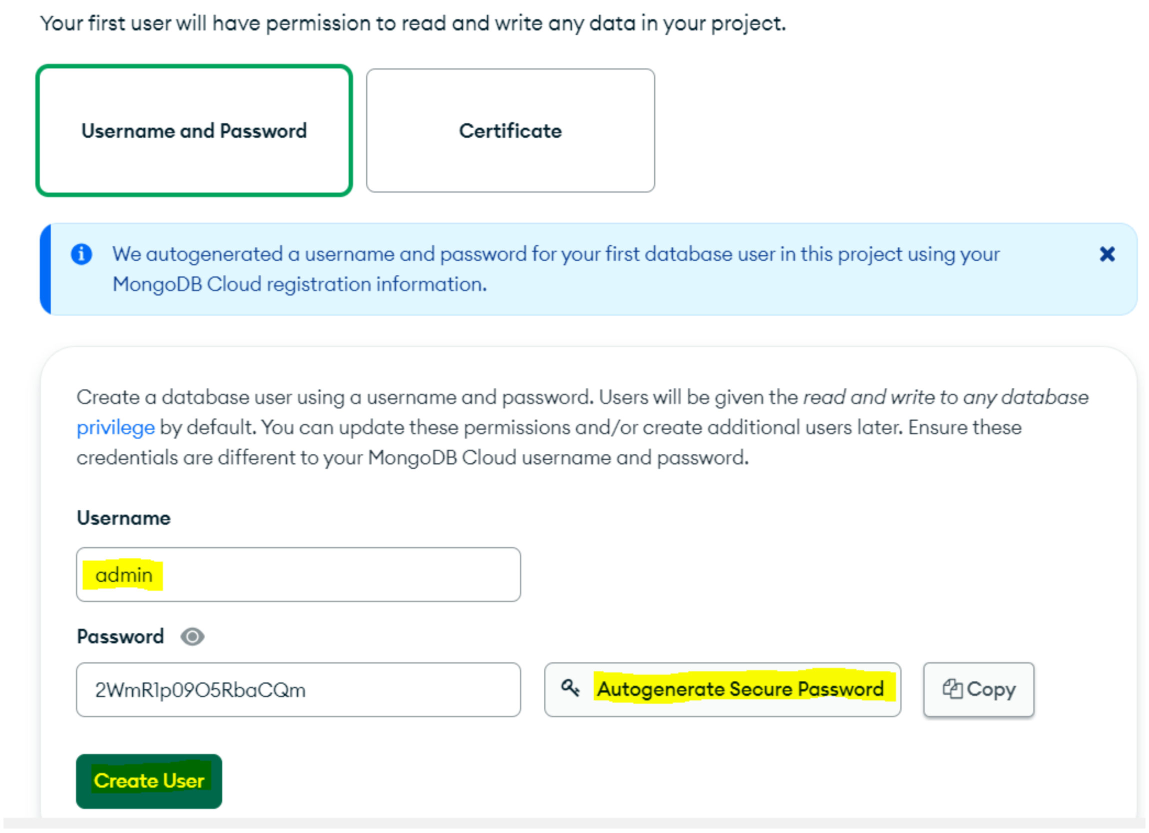
Cluster0

Tag (optional)
Create your first tag to categorize and label your resources; more tags can be added later. Learn more.

$0.08/hour

Pay-as-you-go! You will variable data transfer, b

v. Now create one Username and then after that click on Autogenerate Secure Password. Copy that password and store in some place for future reference. After that click on Create user button.



Your first user will have permission to read and write any data in your project.

We autogenerated a username and password for your first database user in this project using your MongoDB Cloud registration information.

Create a database user using a username and password. Users will be given the *read and write to any database* privilege by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

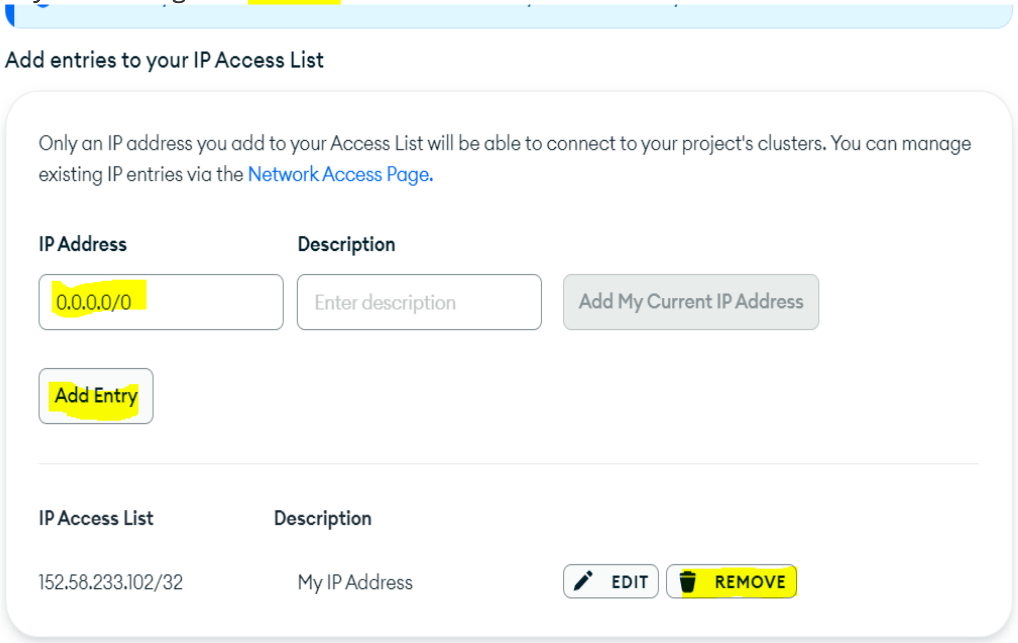
Username

admin

Password

2WmR1p09O5RbaCQm

w. After that you have enter the **IP Adress as 0.0.0.0/0** and click on **Add Entry**. Remove the default IP Adress which is given in the below by clicking on **REMOVE** button



Add entries to your IP Access List

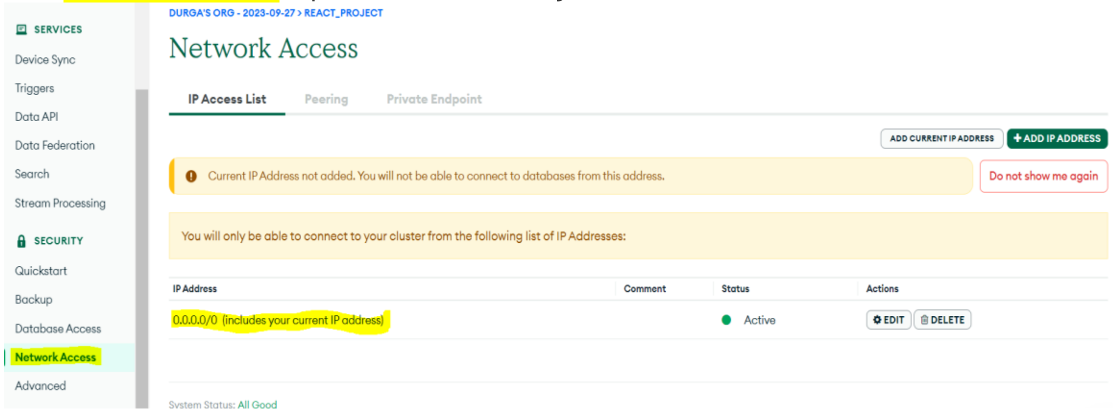
Only an IP address you add to your Access List will be able to connect to your project's clusters. You can manage existing IP entries via the Network Access Page.

IP Address	Description	
0.0.0.0/0	Enter description	Add My Current IP Address

Add Entry

IP Access List	Description	
152.58.233.102/32	My IP Address	EDIT REMOVE

x. You can check whether that IP address is properly added or not in the **Network Access** option in Security Section.



DURGA'S ORG - 2023-09-27 > REACT_PROJECT

Network Access

IP Access List | Peering | Private Endpoint

ADD CURRENT IP ADDRESS | + ADD IP ADDRESS

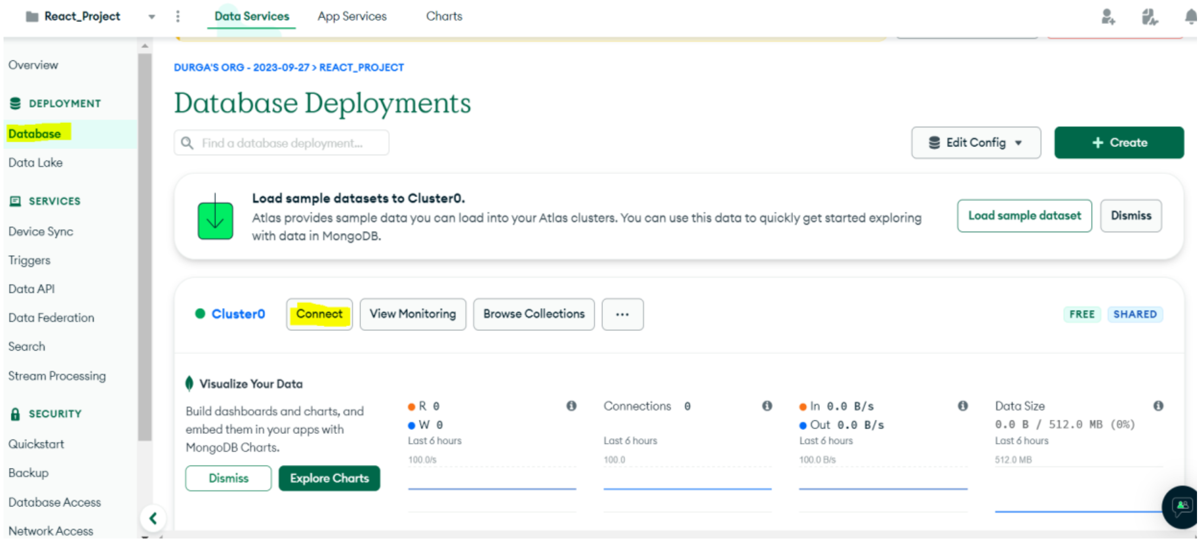
Current IP Address not added. You will not be able to connect to databases from this address. Do not show me again

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
0.0.0.0/0 (includes your current IP address)		● Active	EDIT DELETE

System Status: All Good

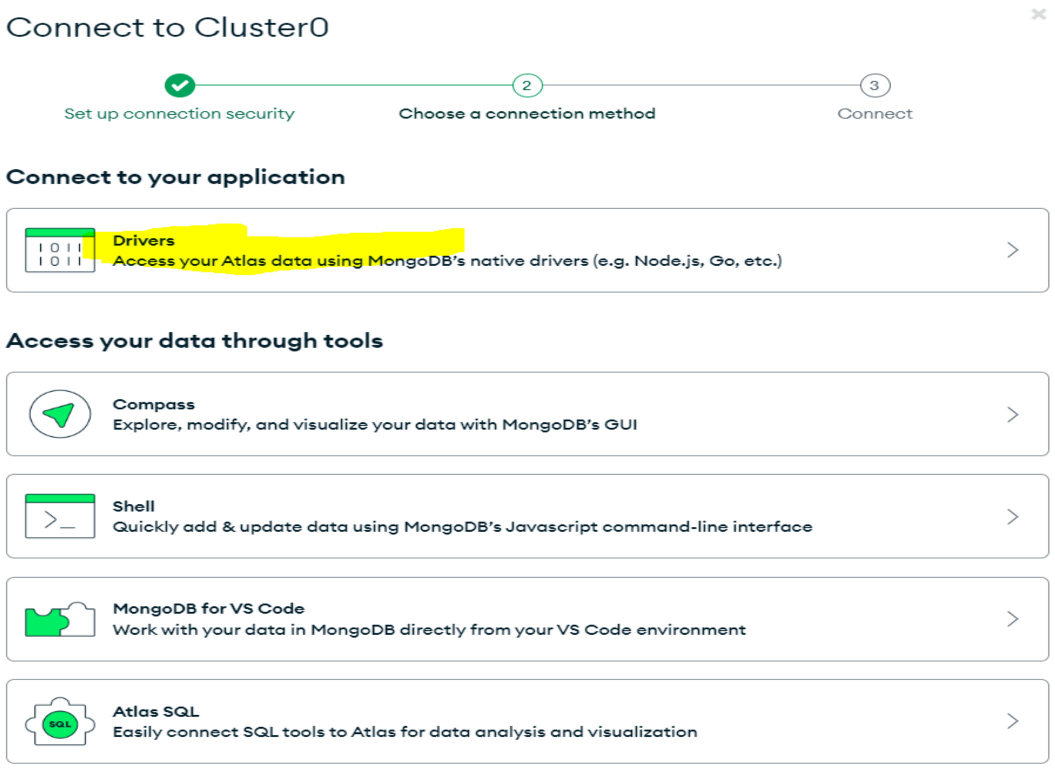
y. After that click on Database option to see the created database. Click on **Connect** option in the database



Load sample datasets to Cluster0.
Atlas provides sample data you can load into your Atlas clusters. You can use this data to quickly get started exploring with data in MongoDB.

Build dashboards and charts, and embed them in your apps with MongoDB Charts.

z. After Clicking on Connect you will get the pop-up like below. Now Click on **Drivers** option.


Connect to Cluster0
Set up connection security
Choose a connection method
Connect
Connect to your application
Access your data through tools

Drivers
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

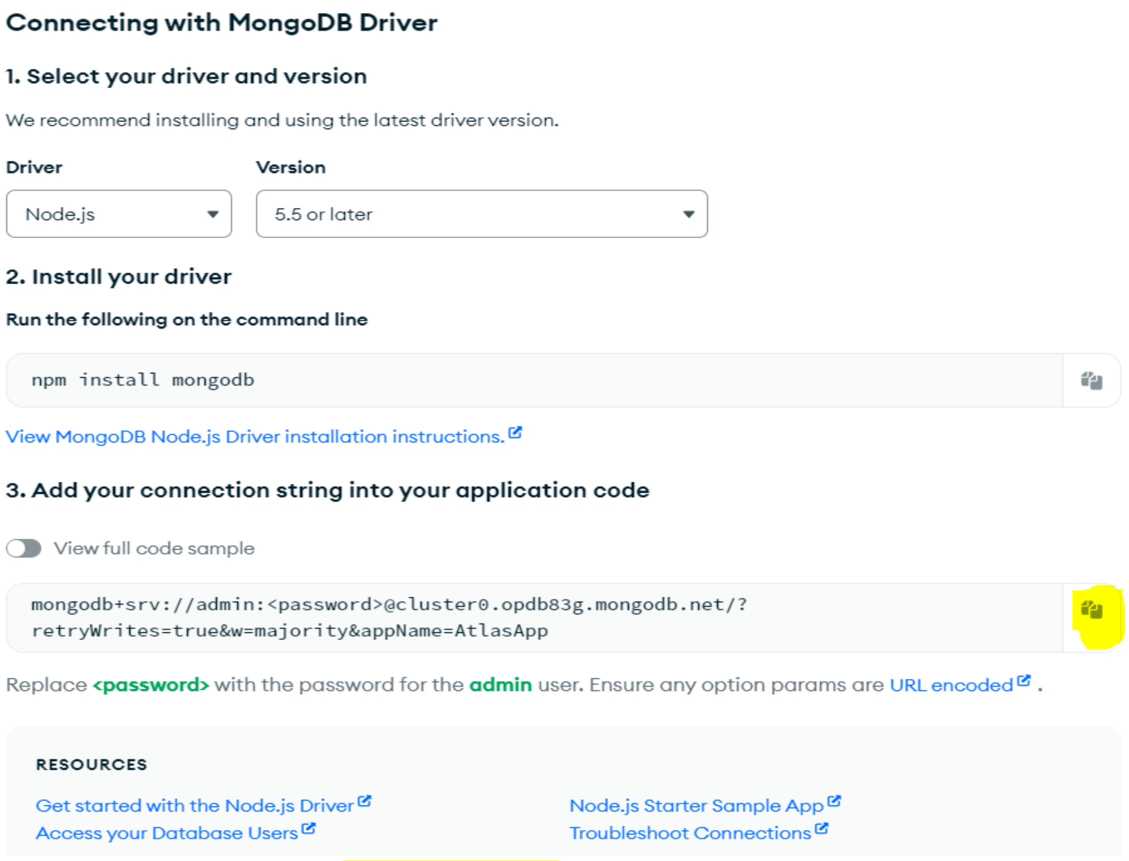
Compass
Explore, modify, and visualize your data with MongoDB's GUI

Shell
Quickly add & update data using MongoDB's Javascript command-line interface

MongoDB for VS Code
Work with your data in MongoDB directly from your VS Code environment

Atlas SQL
Easily connect SQL tools to Atlas for data analysis and visualization

aa. After Clicking on Drivers option, you will get the page like below. Now copy the code from 3 section by clicking on **copy** option in the right side.



Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

Driver: Node.js

Version: 5.5 or later

2. Install your driver

Run the following on the command line

```
npm install mongodb
```

View MongoDB Node.js Driver installation instructions.

3. Add your connection string into your application code

View full code sample

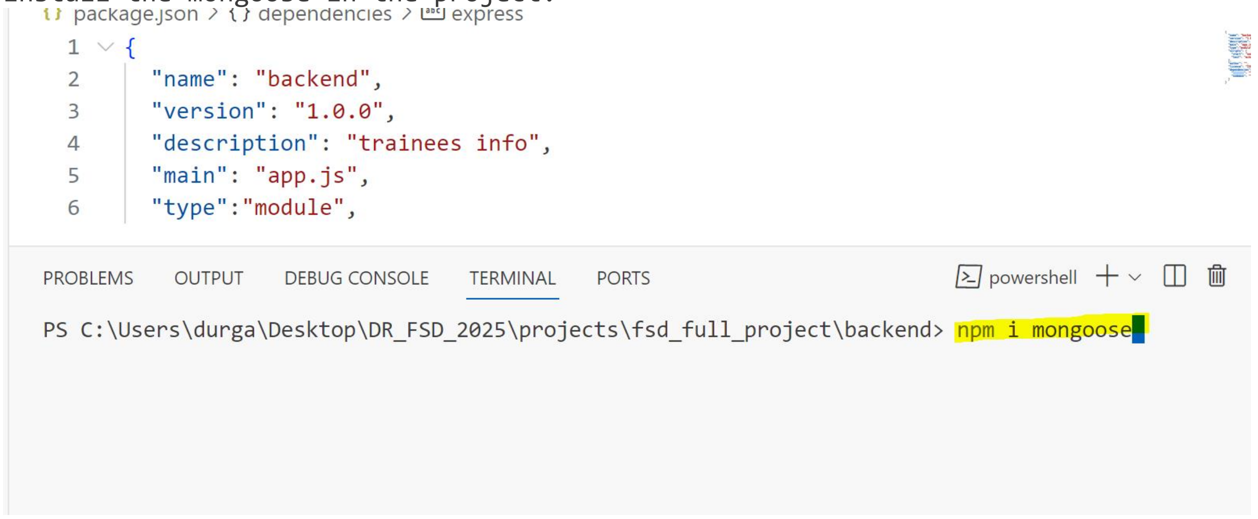
```
mongodb+srv://admin:<password>@cluster0.opdb83g.mongodb.net/?
retryWrites=true&w=majority&appName=AtlasApp
```

Replace **<password>** with the password for the **admin** user. Ensure any option params are URL encoded.

RESOURCES

Get started with the Node.js Driver | Node.js Starter Sample App
Access your Database Users | Troubleshoot Connections

3. Now go to VS Code and run **npm i mongoose** command in the terminal to install the mongoose in the project.

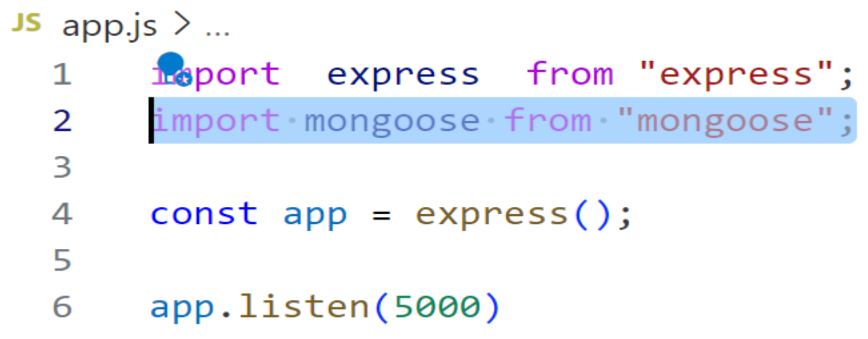


package.json > { } dependencies > express

```
{
  "name": "backend",
  "version": "1.0.0",
  "description": "trainees info",
  "main": "app.js",
  "type":"module",
```

```
PS C:\Users\durga\Desktop\DR_FSD_2025\projects\fsd_full_project\backend> npm i mongoose
```

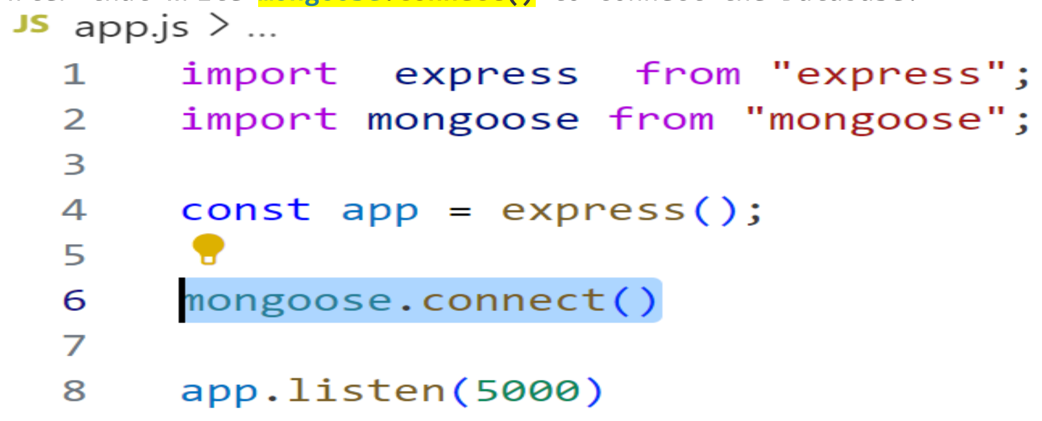
4. Now open **app.js** file and import mongoose package in the page. `import mongoose from "mongoose";` by entering the code in the file we can import the mongoose into the page.



JS app.js > ...

```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

app.listen(5000)
```

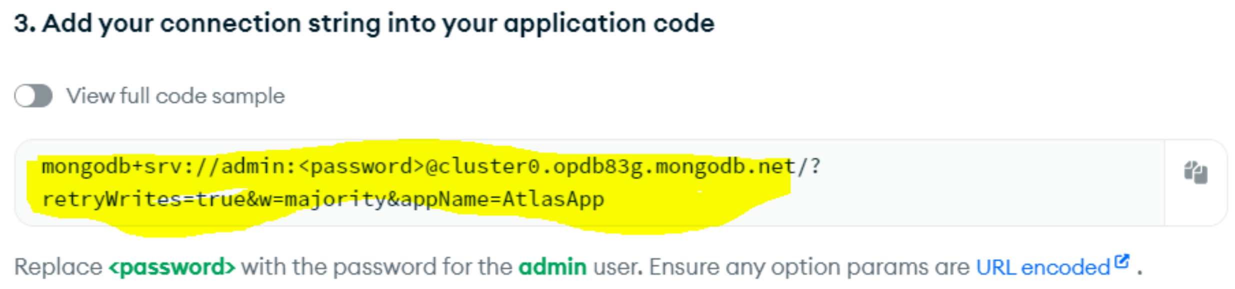
5. After that write `mongoose.connect()` to connect the Database.



JS app.js > ...

```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

mongoose.connect()

app.listen(5000)
```

6. Now copy the database connection link from Atlas and paste the copied link inside of the **mongoose.connect**() function.

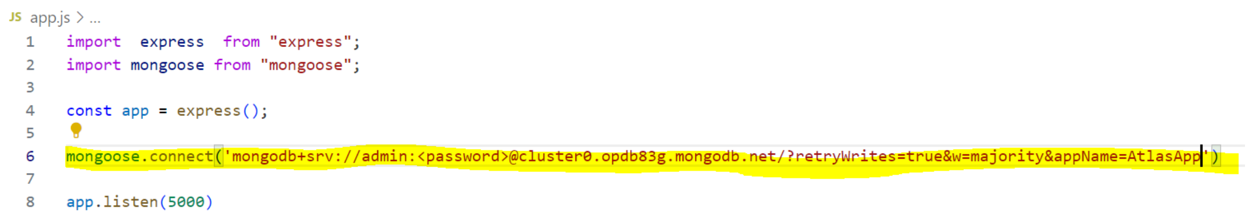


3. Add your connection string into your application code

View full code sample

```
mongodb+srv://admin:<password>@cluster0.opdb83g.mongodb.net/?retryWrites=true&w=majority&appName=AtlasApp
```

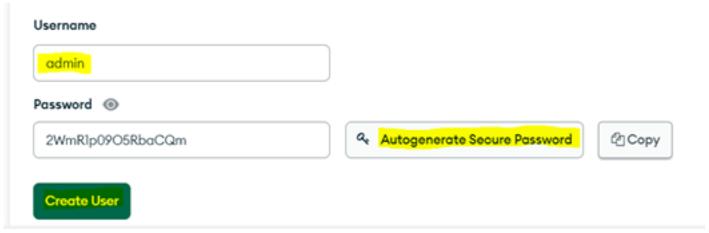
Replace **<password>** with the password for the **admin** user. Ensure any option params are URL encoded .



JS app.js > ...

```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

mongoose.connect('mongodb+srv://admin:<password>@cluster0.opdb83g.mongodb.net/?retryWrites=true&w=majority&appName=AtlasApp')

app.listen(5000)
```

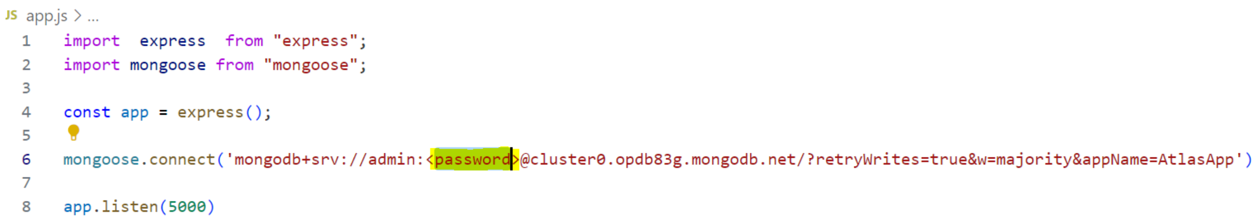
7. Now we have to replace the **password** in that connection link with the **Autogenerated password** which was previously generated.


Username

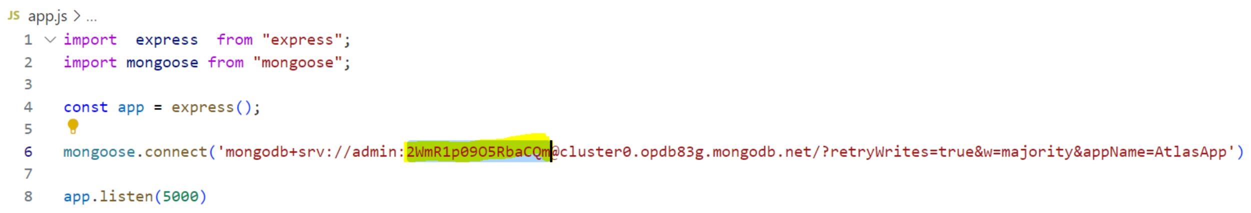
admin

Password

2WmR1p09O5RbaCQm

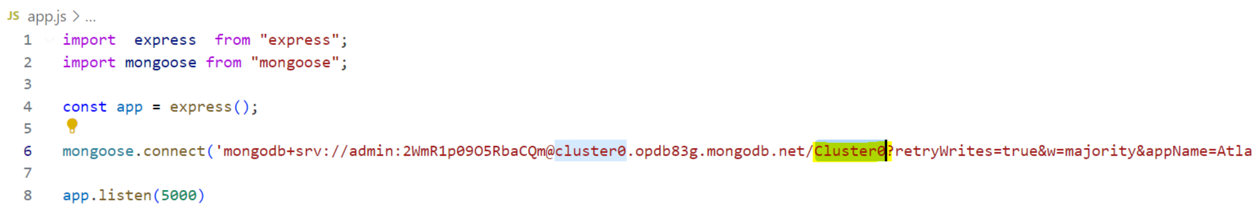

app.js > ...

```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

mongoose.connect('mongodb+srv://admin:<password>@cluster0.opdb83g.mongodb.net/?retryWrites=true&w=majority&appName=AtlasApp')

app.listen(5000)
```


app.js > ...

```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

mongoose.connect('mongodb+srv://admin:2WmR1p09O5RbaCQm@cluster0.opdb83g.mongodb.net/?retryWrites=true&w=majority&appName=AtlasApp')

app.listen(5000)
```

Add Database name shown like below. (**Cluster0** is DB name)


app.js > ...

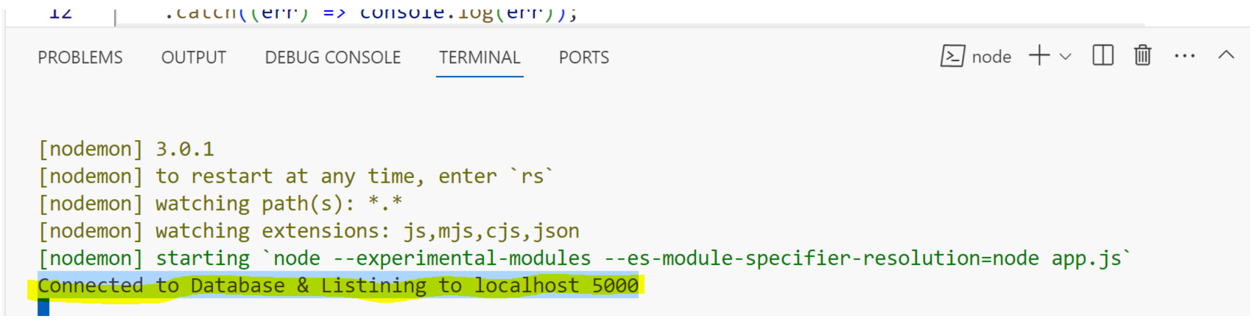
```
import  express  from "express";
import mongoose from "mongoose";

const app = express();

mongoose.connect('mongodb+srv://admin:2WmR1p09O5RbaCQm@cluster0.opdb83g.mongodb.net/Cluster0?retryWrites=true&w=majority&appName=Atla
```

```
app.listen(5000)
```

8. Now write the below code to check whether the Database connection is connected properly or not.

```
mongoose.connect('mongodb+srv://admin:2WmR1p09O5RbaCQm@cluster0.opdb83g.mo
ngodb.net/Cluster0?retryWrites=true&w=majority&appName=AtlasApp'
)
    .then(() => app.listen(5000))
    .then(() =>
        console.log("Connected to Database & Listining to localhost 5000")
    )
    .catch((err) => console.log(err));
```

9. After that run **npm start** command to check the database connection whether it is working or not
10. If the connection is established properly you will get the message like below which we written inside of console.



```
.catch((err) => console.log(err));
```

```
[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 --experimental-modules --es-module-specifier-resolution=node app.js`
Connected to Database & Listining to localhost 5000
```