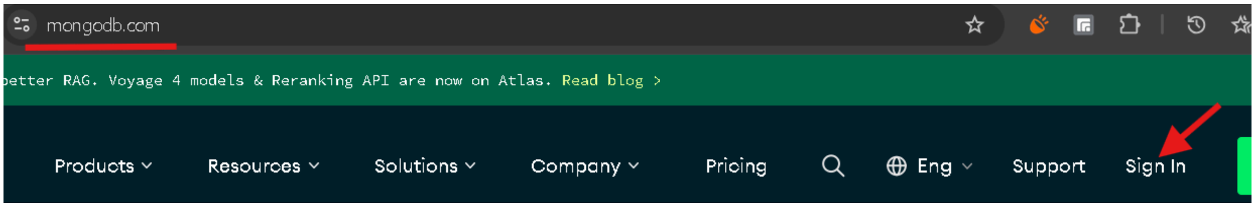
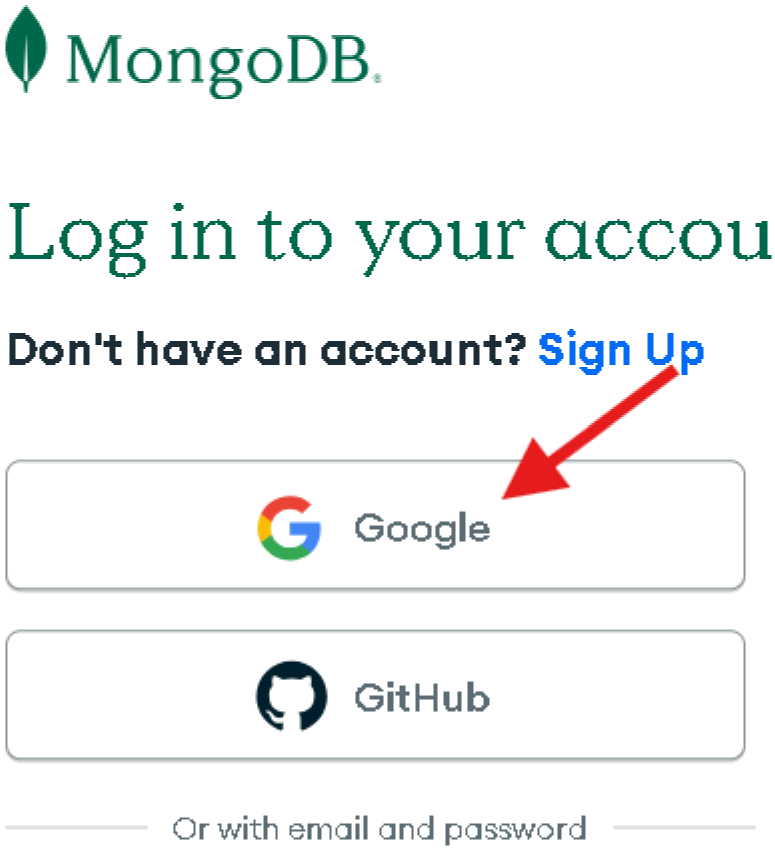
Visit: https://www.mongodb.com/
Click: Sign in

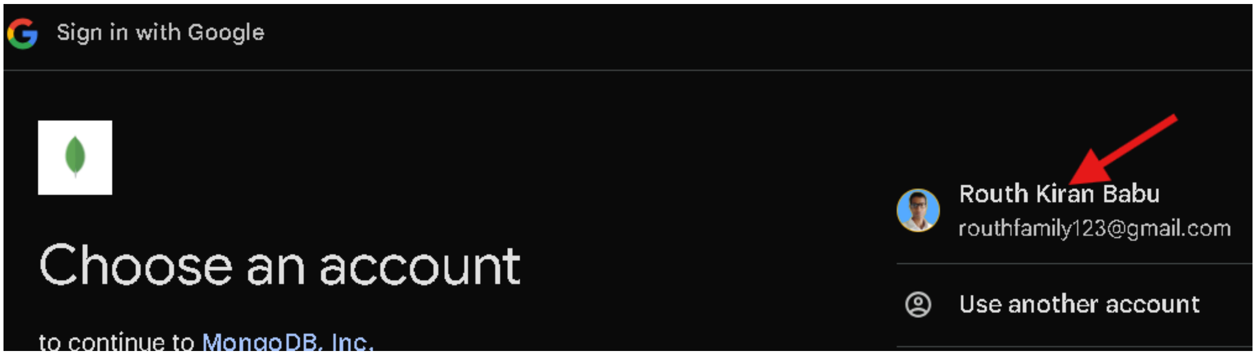

mongodb.com
etter RAG. Voyage 4 models & Reranking API are now on Atlas. Read blog >
Products
Resources
Solutions
Company
Pricing
Eng
Support
Sign In

Login with google account:


MongoDB

Log in to your account

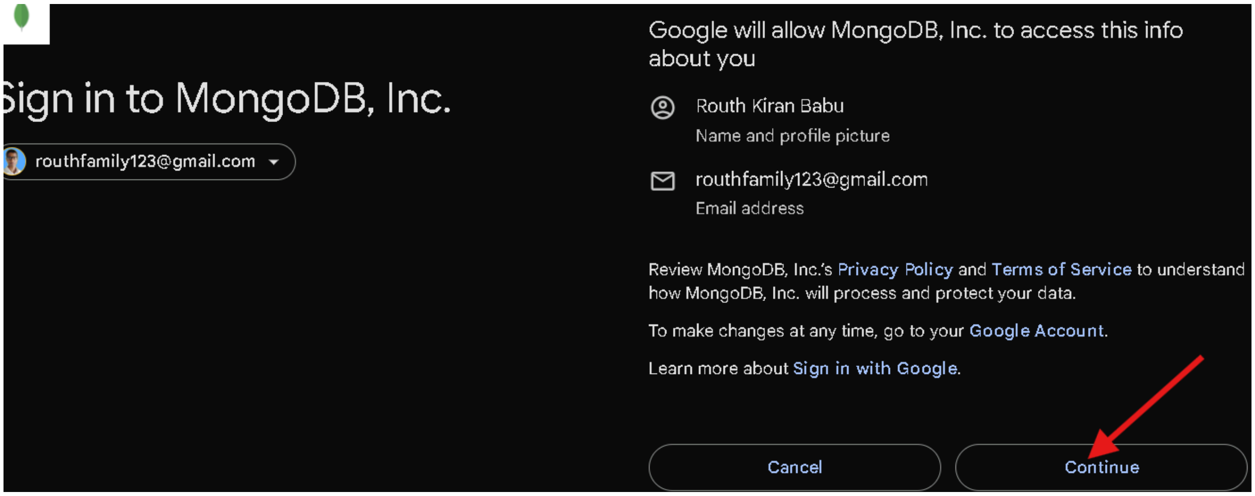
Don't have an account? Sign Up



Choose an account

to continue to MongoDB, Inc.

Routh Kiran Babu
routhfamily123@gmail.com



Sign in to MongoDB, Inc.

routhfamily123@gmail.com

Google will allow MongoDB, Inc. to access this info about you

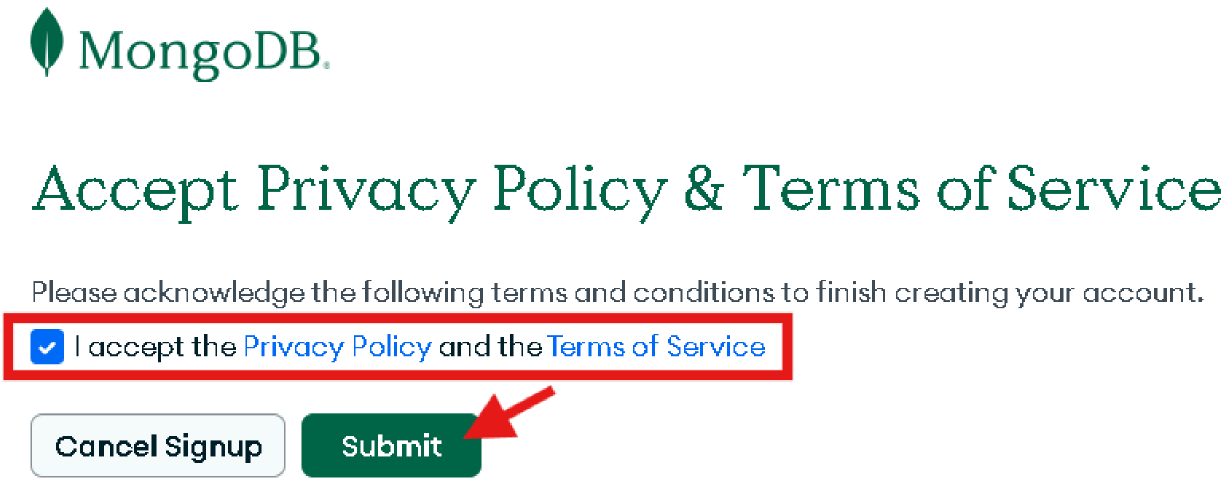
Routh Kiran Babu
Name and profile picture

routhfamily123@gmail.com
Email address

Review MongoDB, Inc.'s Privacy Policy and Terms of Service to understand how MongoDB, Inc. will process and protect your data.

To make changes at any time, go to your Google Account.

Learn more about Sign in with Google.

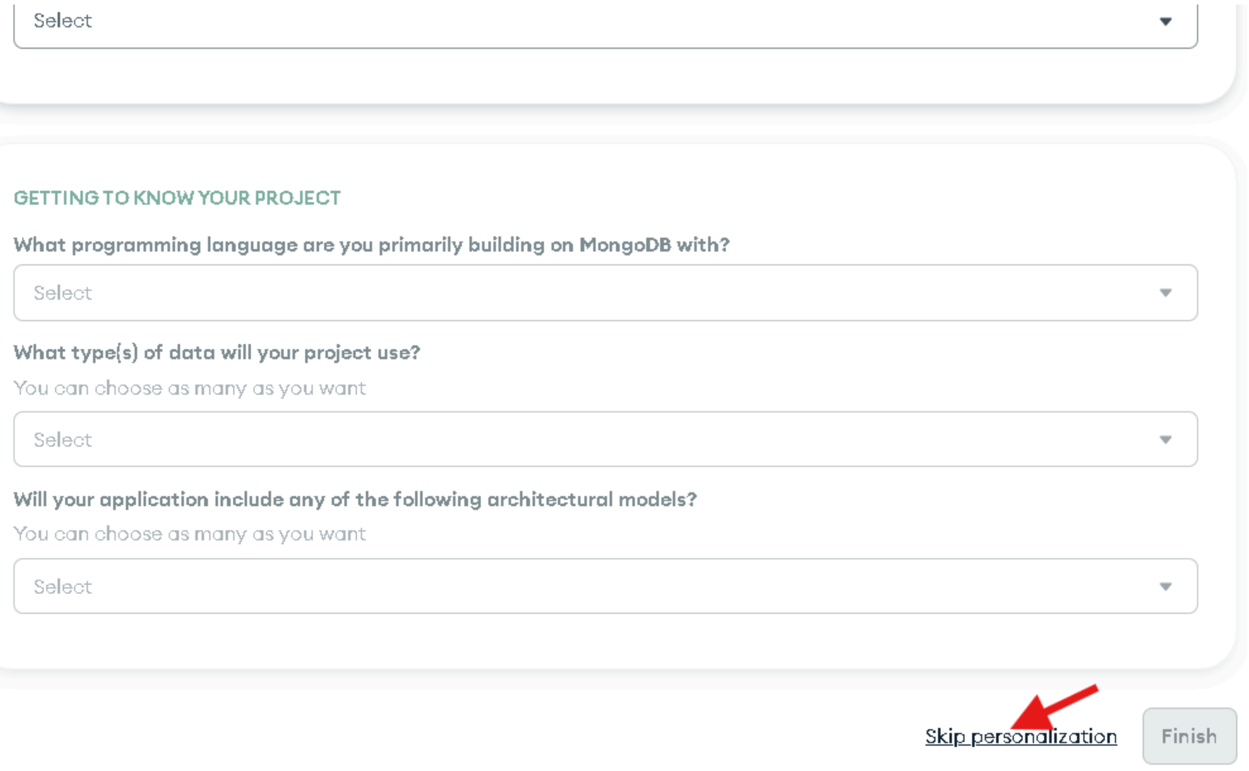


MongoDB.

Accept Privacy Policy & Terms of Service

Please acknowledge the following terms and conditions to finish creating your account.

☑ I accept the Privacy Policy and the Terms of Service



Select ▾

GETTING TO KNOW YOUR PROJECT

What programming language are you primarily building on MongoDB with?

Select ▾

What type(s) of data will your project use?

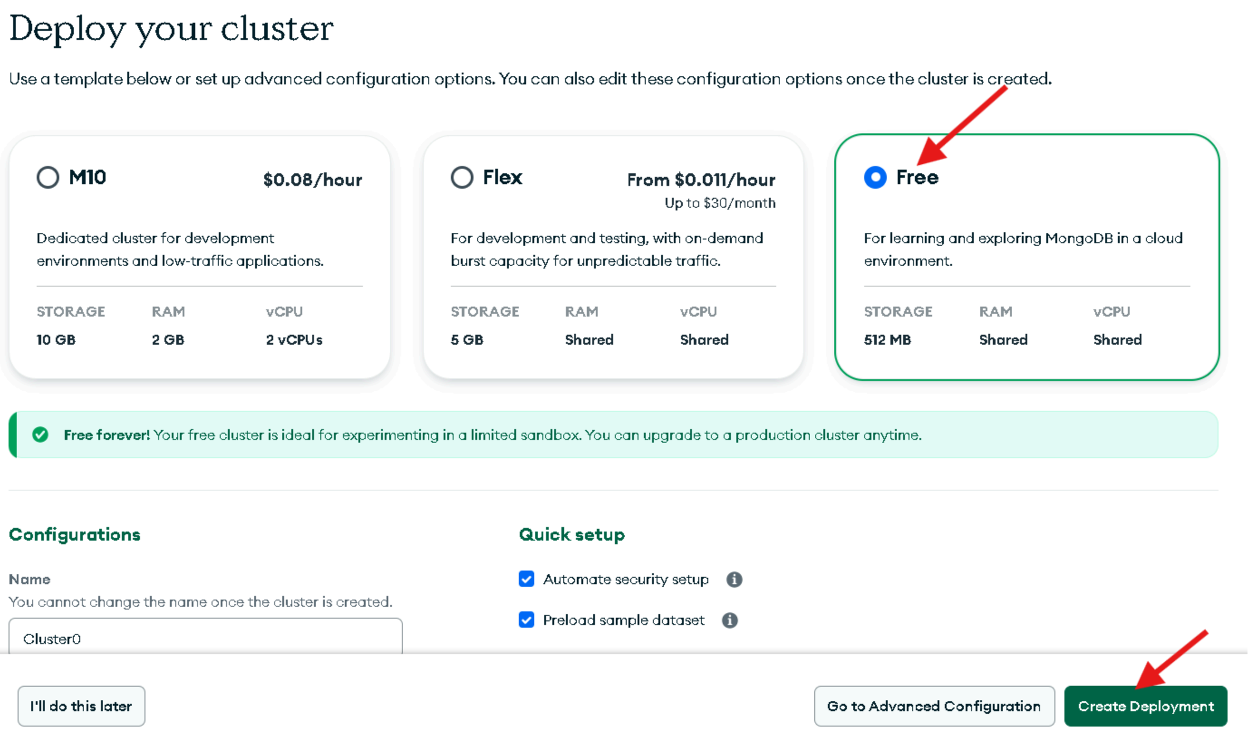
You can choose as many as you want

Select ▾

Will your application include any of the following architectural models?

You can choose as many as you want

Select ▾



Deploy your cluster

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**

Dedicated cluster for development environments and low-traffic applications.

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

○ **Flex** **From $0.011/hour**
Up to $30/month

For development and testing, with on-demand burst capacity for unpredictable traffic.

STORAGE	RAM	vCPU
5 GB	Shared	Shared

● **Free**

For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM	vCPU
512 MB	Shared	Shared

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

Configurations

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

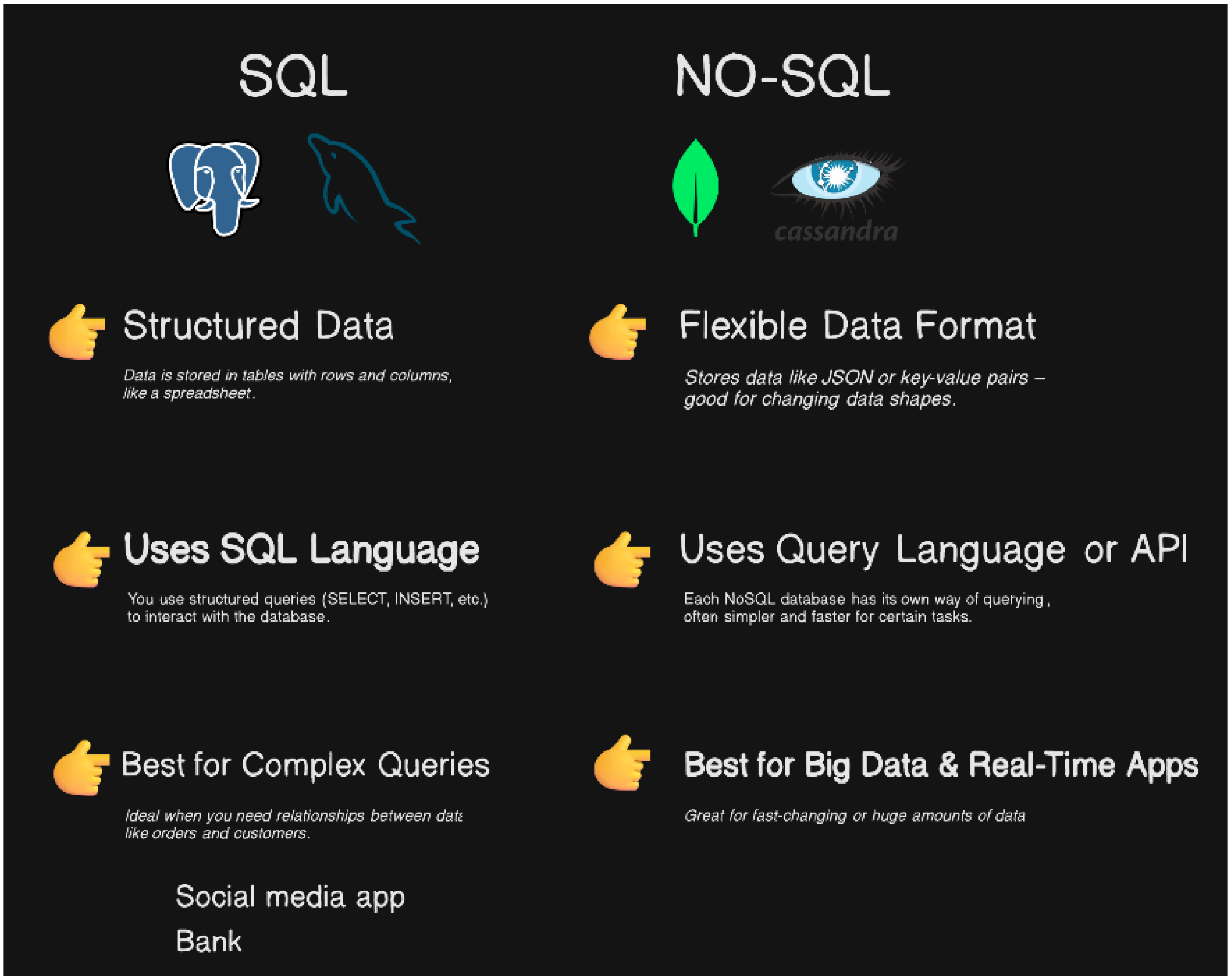
Cluster0

Quick setup

☑ Automate security setup ⓘ

☑ Preload sample dataset ⓘ

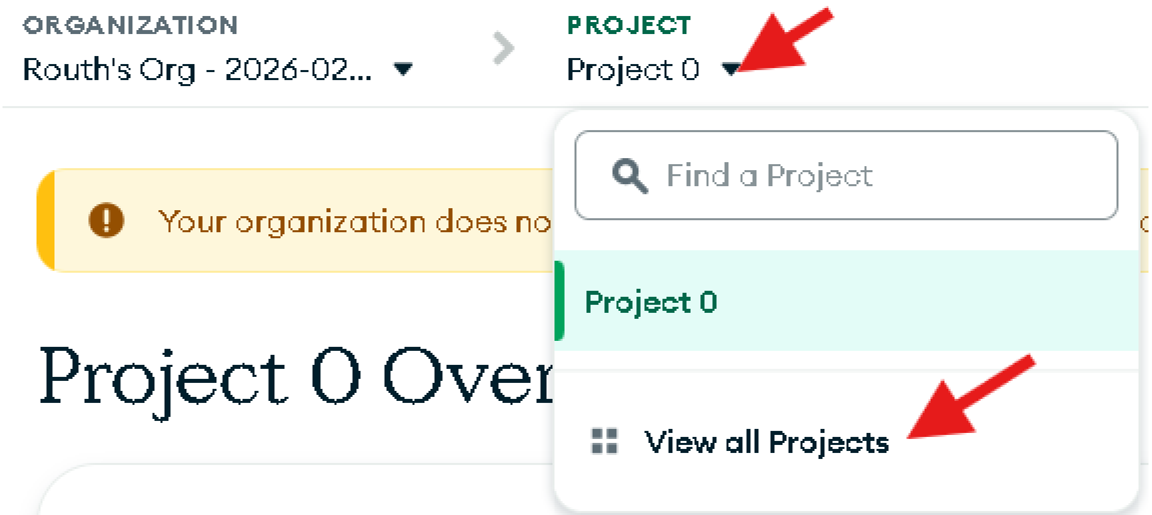
Difference between SQL and NoSQL:



SQL	NO-SQL
Structured Data *Data is stored in tables with rows and columns, like a spreadsheet.*	**Flexible Data Format** *Stores data like JSON or key-value pairs – good for changing data shapes.*
Uses SQL Language You use structured queries (SELECT, INSERT, etc.) to interact with the database.	**Uses Query Language or API** Each NoSQL database has its own way of querying, often simpler and faster for certain tasks.
Best for Complex Queries *Ideal when you need relationships between dat like orders and customers.*	**Best for Big Data & Real-Time Apps** *Great for fast-changing or huge amounts of data*

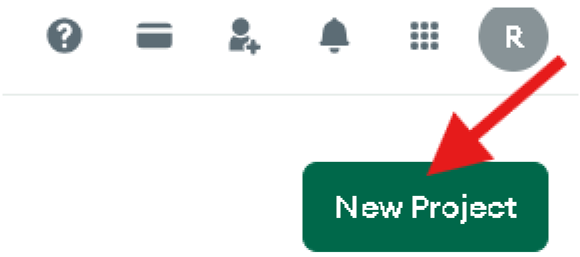
Social media app
Bank

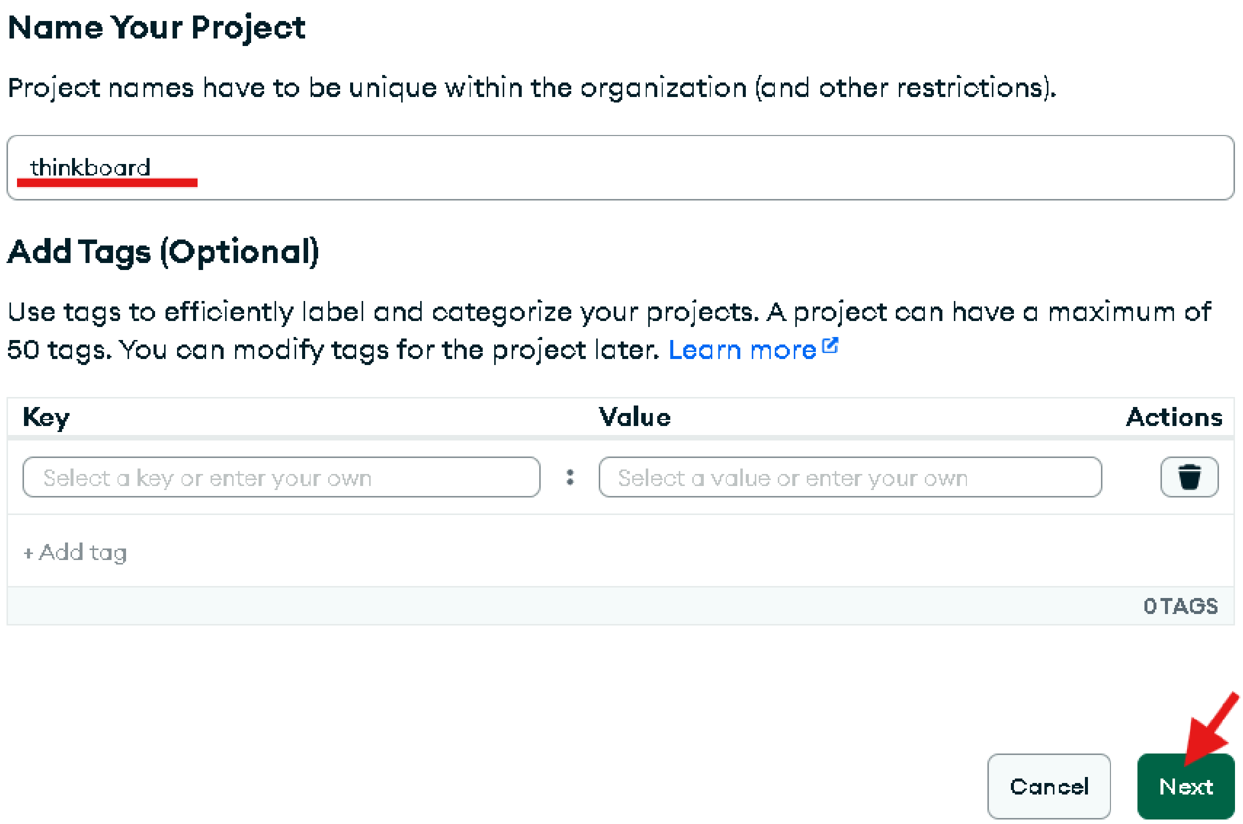
Setting up MongoDB:



Your organization does no

Project 0 Over





Name Your Project

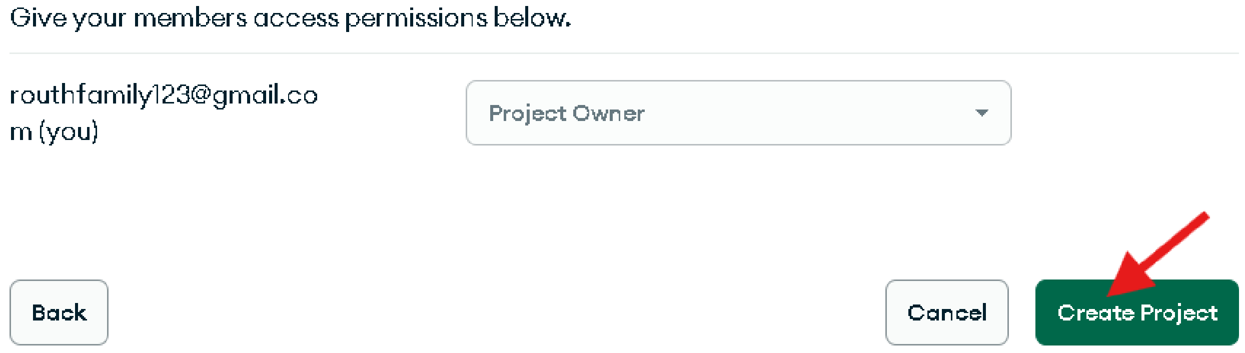
Project names have to be unique within the organization (and other restrictions).

thinkboard

Add Tags (Optional)

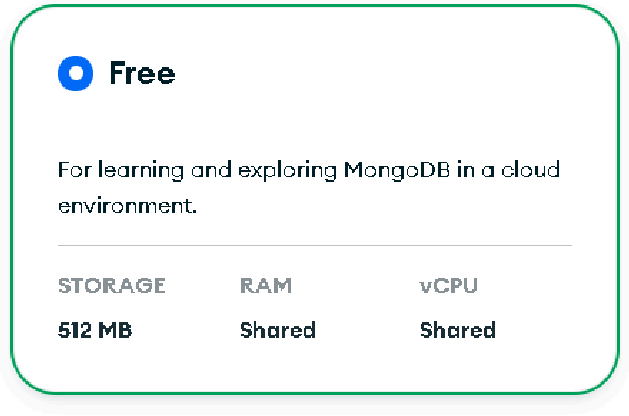
Use tags to efficiently label and categorize your projects. A project can have a maximum of 50 tags. You can modify tags for the project later. Learn more

Key		Value	Actions
Select a key or enter your own	:	Select a value or enter your own	



Give your members access permissions below.

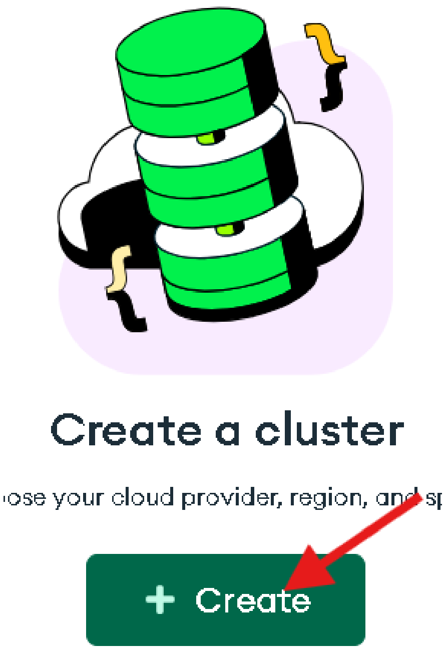
routhfamily123@gmail.com (you) — Project Owner



○ Free

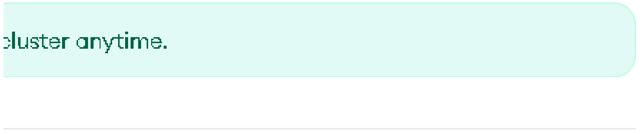
For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM	vCPU
512 MB	Shared	Shared

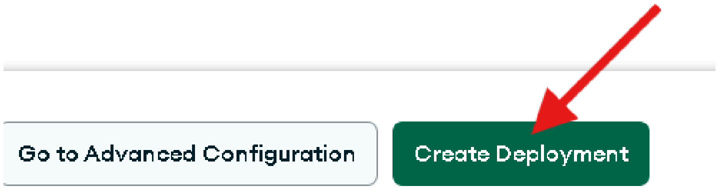


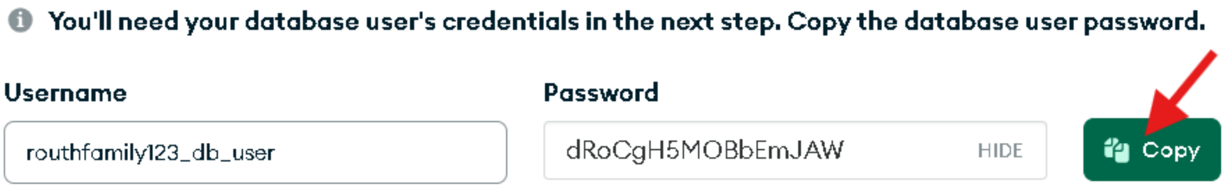
Create a cluster

ose your cloud provider, region, and sp



cluster anytime.

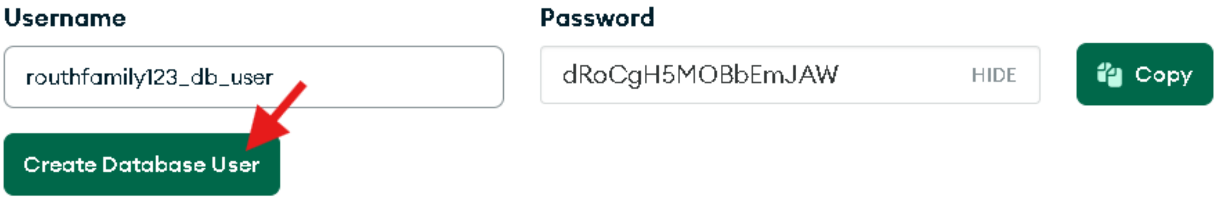




You'll need your database user's credentials in the next step. Copy the database user password.

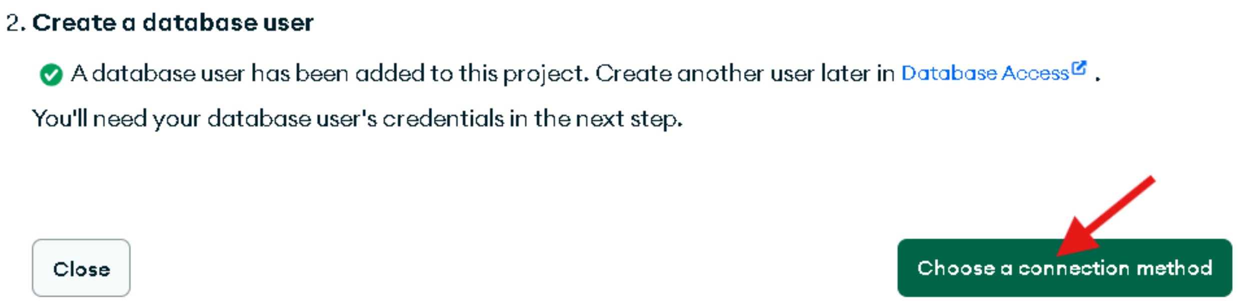
Username: routhfamily123_db_user

Password: dRoCgH5MOBbEmJAW



Username: routhfamily123_db_user

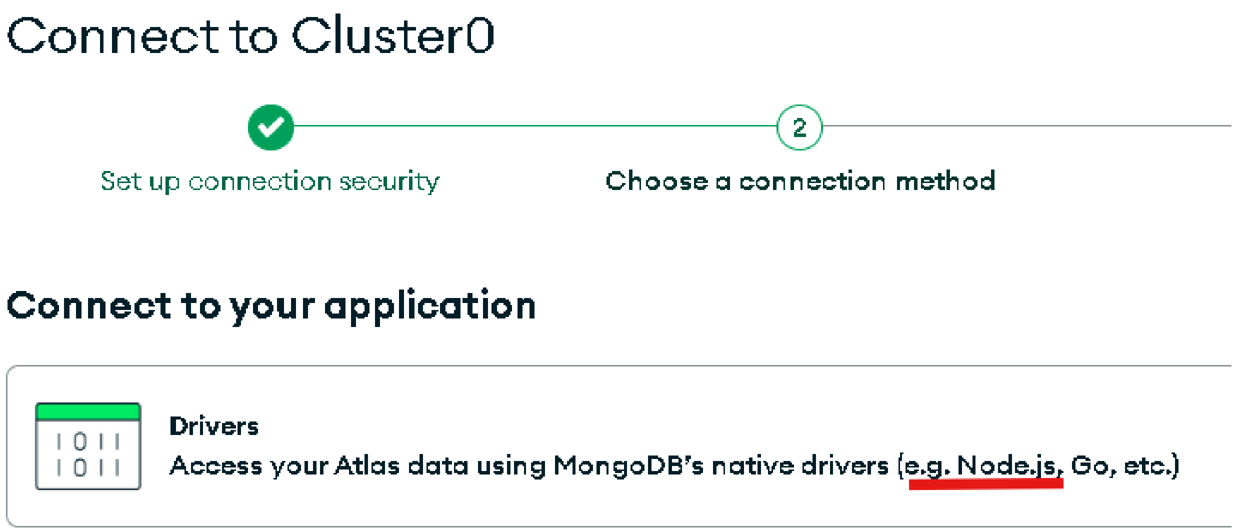
Password: dRoCgH5MOBbEmJAW



2. **Create a database user**

✓ A database user has been added to this project. Create another user later in Database Access .

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

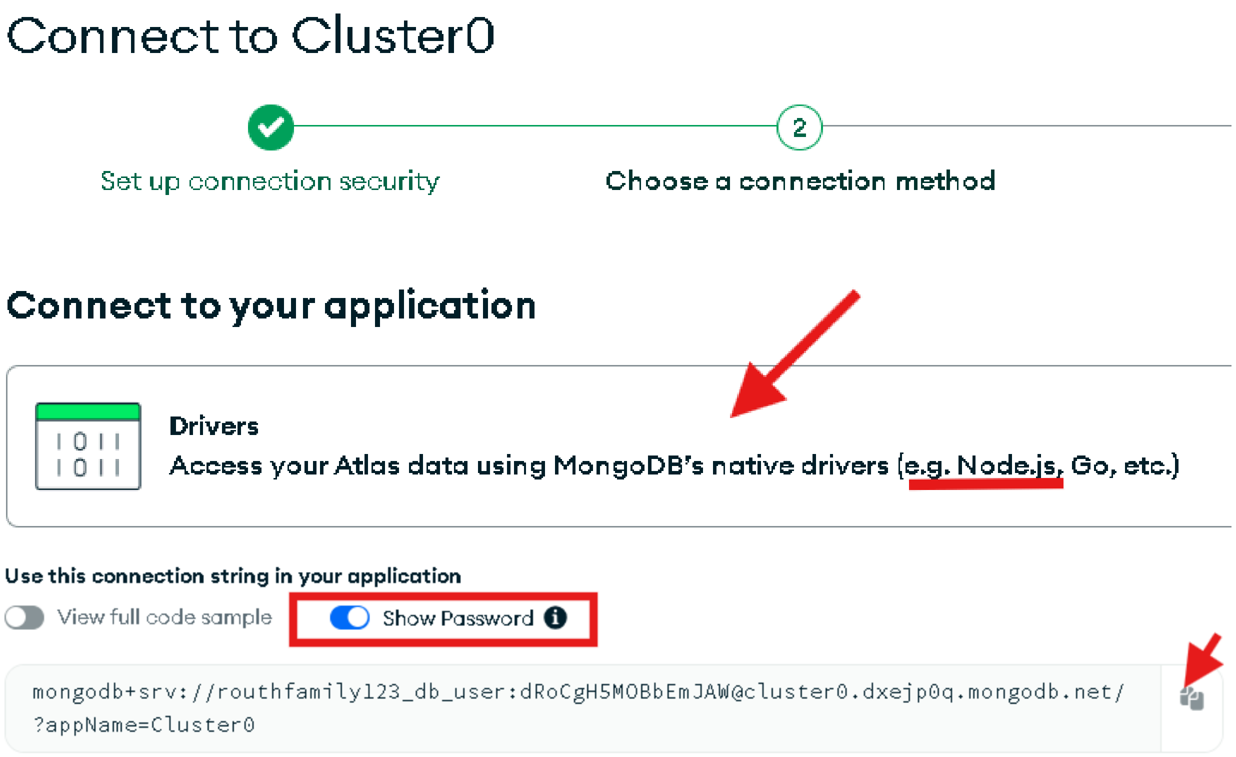


Connect to Cluster0

Connect to your application

Drivers

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



Connect to Cluster0

✓ Set up connection security
2 Choose a connection method

Connect to your application

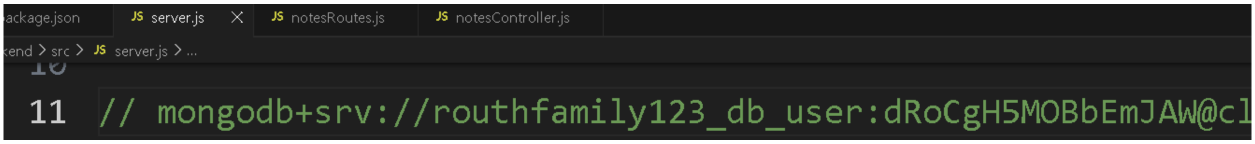
Drivers

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

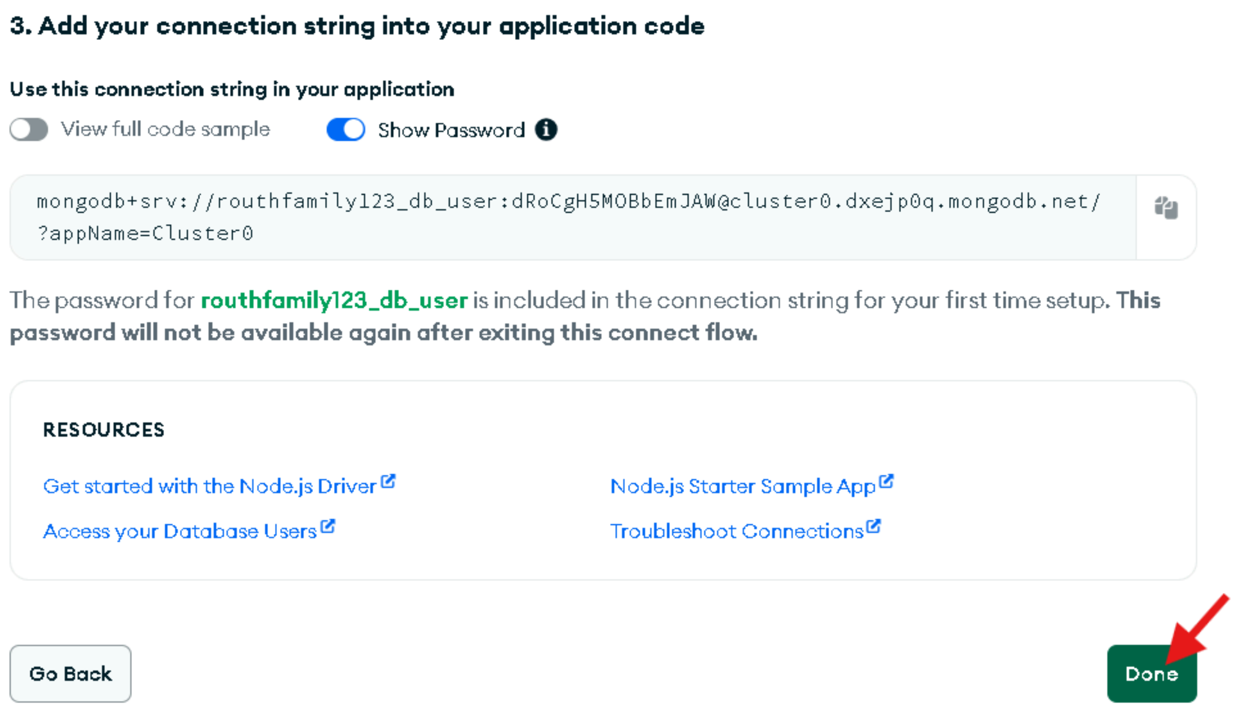
Use this connection string in your application

```
mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/
?appName=Cluster0
```

Save it somewhere like:



```
// mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cl
```



3. Add your connection string into your application code

Use this connection string in your application

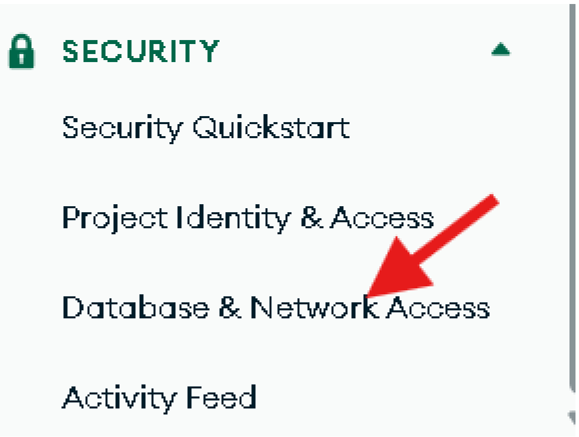
```
mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/
?appName=Cluster0
```

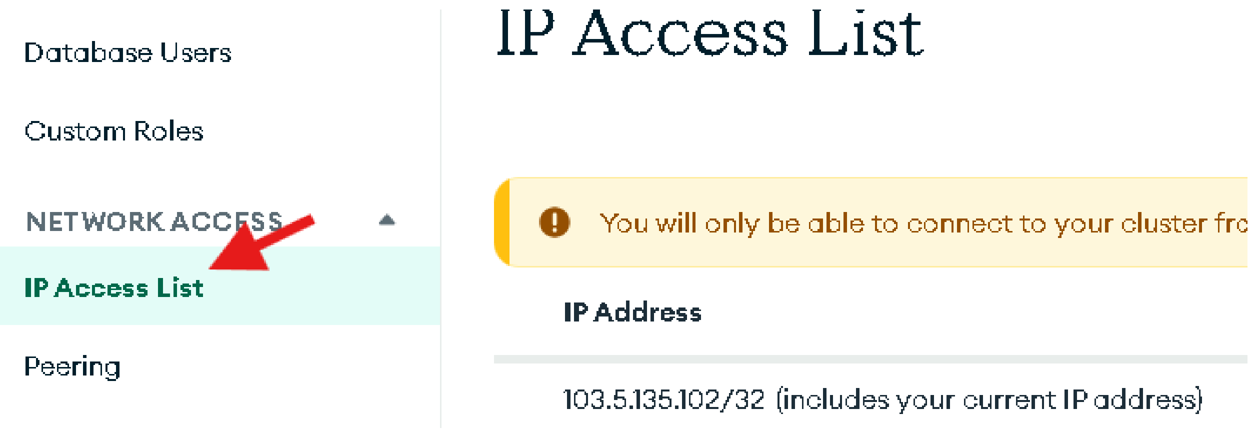
The password for **routhfamily123_db_user** is included in the connection string for your first time setup. **This password will not be available again after exiting this connect flow.**

RESOURCES

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

Before connecting to the database:
Under network access:



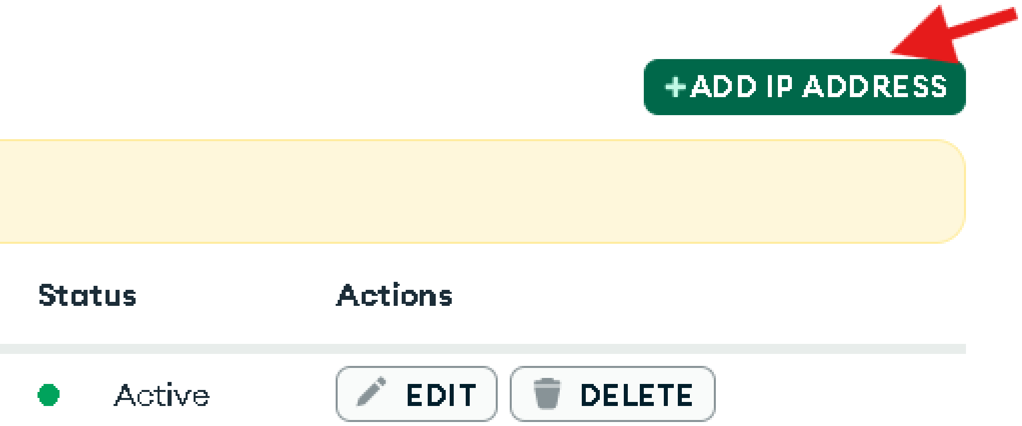


IP Access List

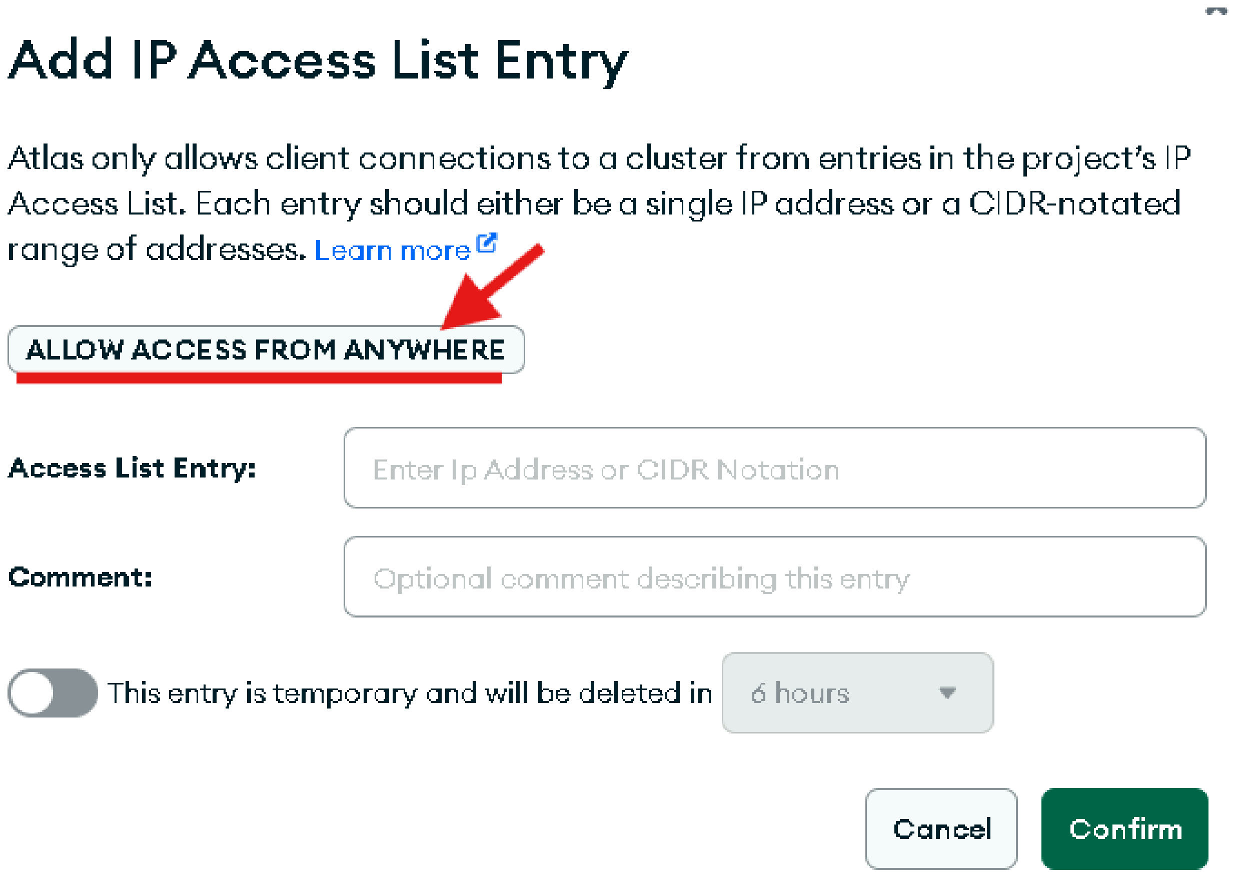
You will only be able to connect to your cluster fro

IP Address
103.5.135.102/32 (includes your current IP address)

To not get any errors:



Status	Actions
● Active	EDIT DELETE



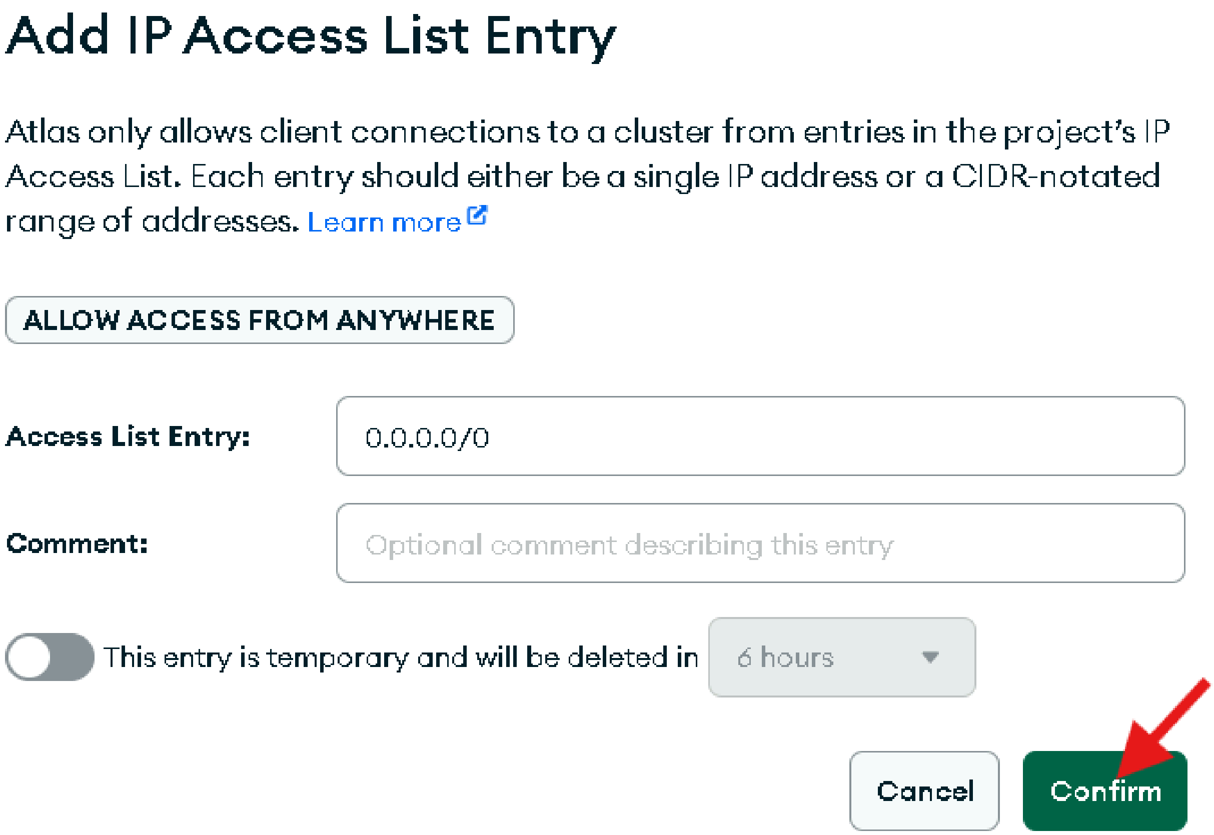
Add IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more

Access List Entry: Enter Ip Address or CIDR Notation

Comment: Optional comment describing this entry

This entry is temporary and will be deleted in 6 hours



Add IP Access List Entry

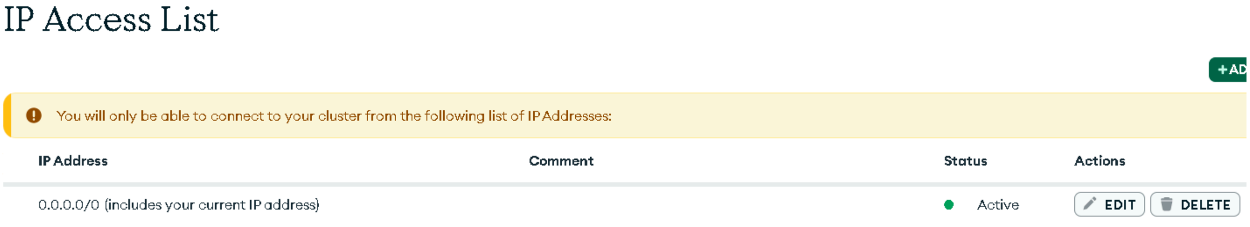
Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. Learn more

Access List Entry: 0.0.0.0/0

Comment: Optional comment describing this entry

This entry is temporary and will be deleted in 6 hours

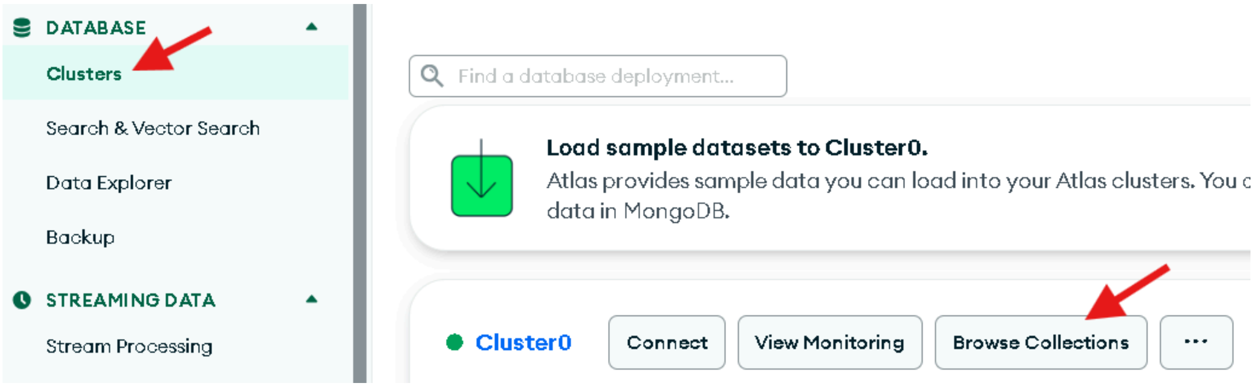
Wait for 10 to 15 seconds.
Created, may get Gmail message.



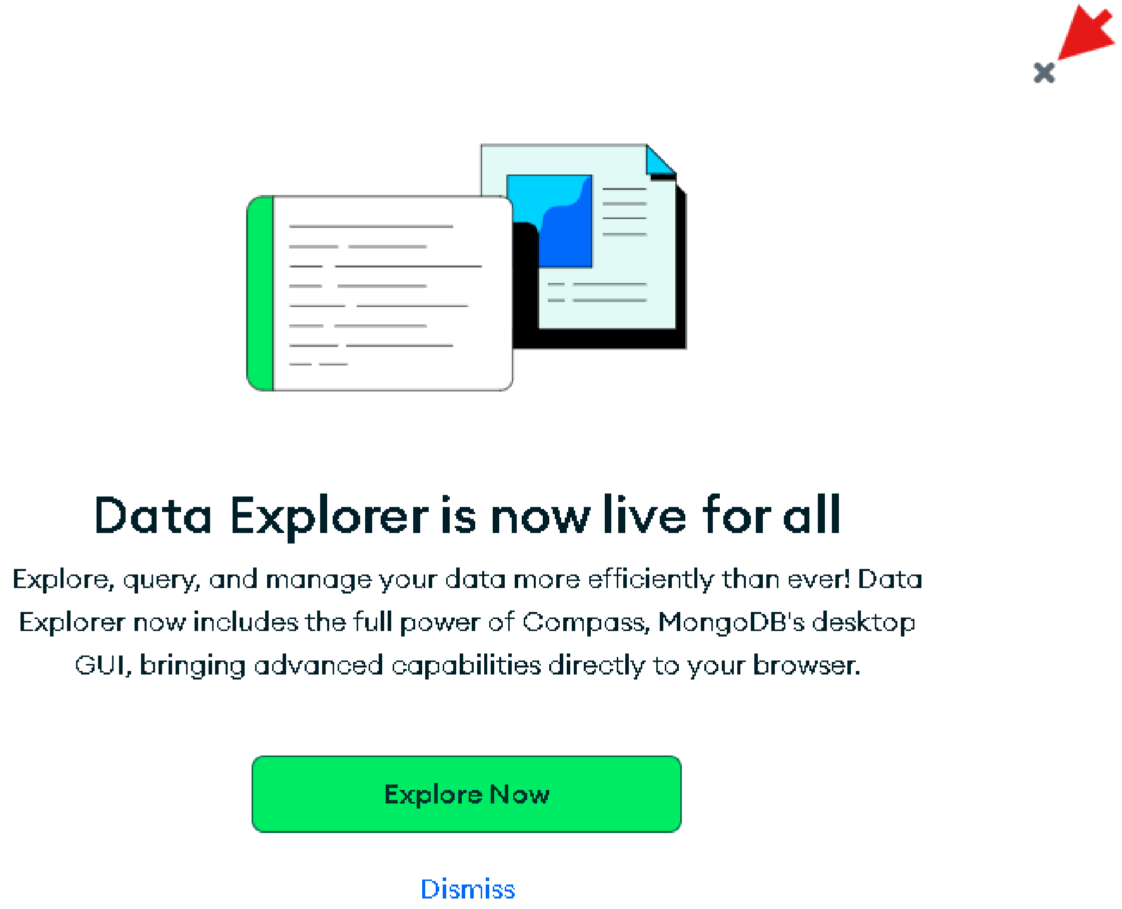
IP Access List

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

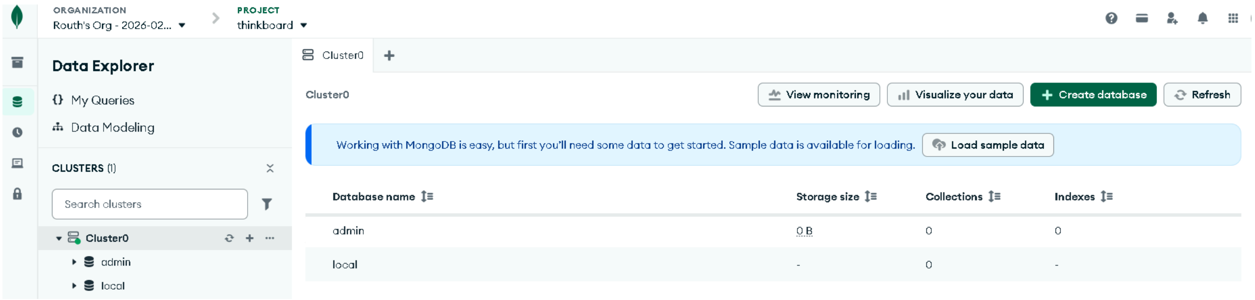


Load sample datasets to Cluster0.
Atlas provides sample data you can load into your Atlas clusters. You c
data in MongoDB.



Data Explorer is now live for all

Explore, query, and manage your data more efficiently than ever! Data Explorer now includes the full power of Compass, MongoDB's desktop GUI, bringing advanced capabilities directly to your browser.



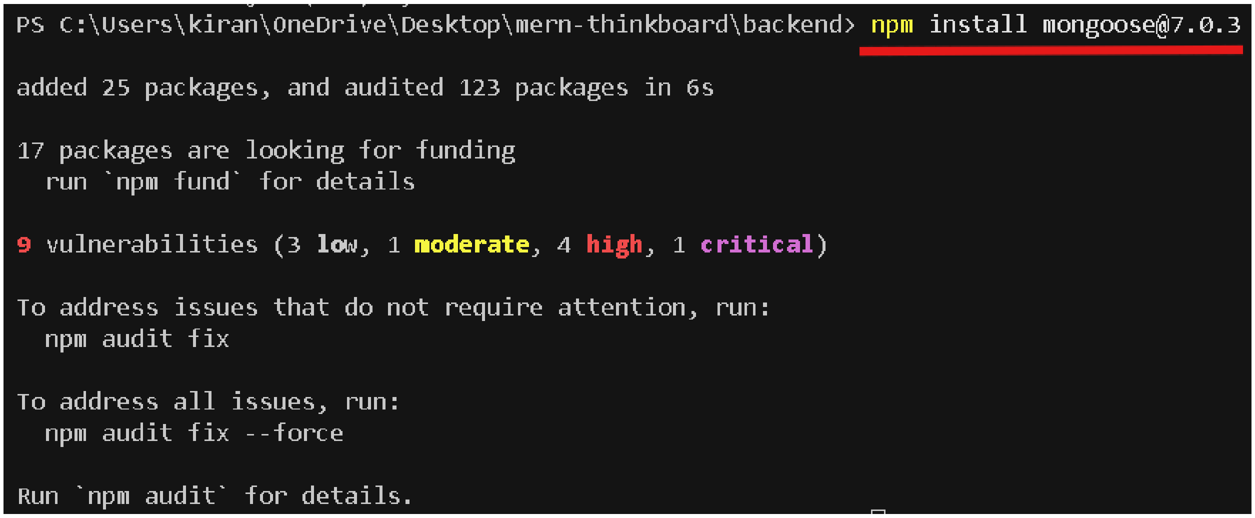
Working with MongoDB is easy, but first you'll need some data to get started. Sample data is available for loading.

Database name	Storage size	Collections	Indexes
admin	0 B	0	0
local	-	0	-

Stop running the server.

Installing dependencies[mongoose]:

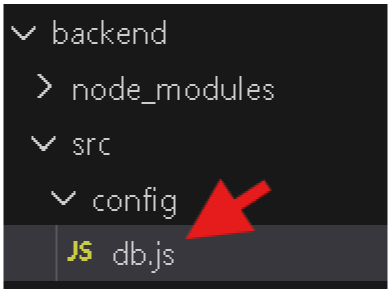
Make sure you are in the backend folder.
Command(mongoose of version 7.0.3): **npm install mongoose@7.0.3**



```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm install mongoose@7.0.3

added 25 packages, and audited 123 packages in 6s

17 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (3 low, 1 moderate, 4 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.
```

To Connect to the Database:

In src > Create config folder > create db.js file.



Code in db.js:

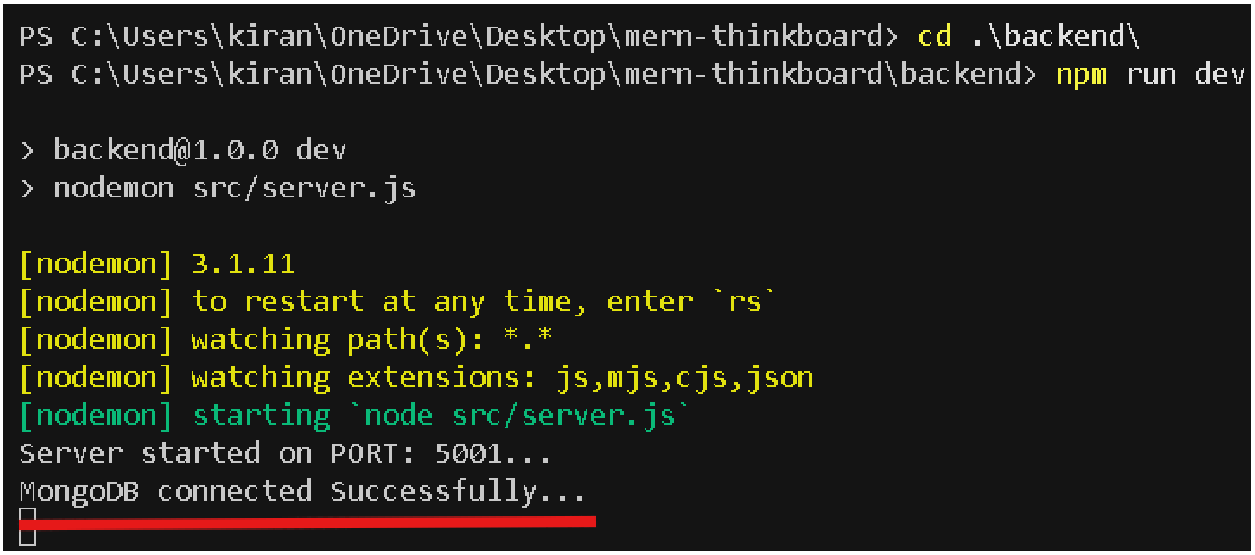
```
import mongoose from "mongoose"

// Creating a function that connects to the database
// Export to use it in server.js
export const connectDB = async () => {
   try {
      // connect by string we got earlier from MongoDB
      await mongoose.connect("mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/?appName=Cluster0")
      console.log("MongoDB connected Successfully...")
   } catch (error) {
      console.log("Error connecting to MongoDB:", error)
      // 1 = exit with failure, if there's an error
      process.exit(1) // 0 = success
   }
```

}

Code in server.js:

```
import express from "express"
import notesRoutes from "./routes/notesRoutes.js"
// Importing the connection function
import { connectDB } from "./config/db.js"
const app = express()

app.use("/api/notes", notesRoutes)
// Calling the connection function to connect
connectDB()

app.listen(5001, () => {
   console.log("Server started on PORT: 5001...")
})

//
mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/?appName=Cluster0
```

Run the code in terminal:



```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> cd .\backend\
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev

> backend@1.0.0 dev
> nodemon src/server.js

[nodemon] 3.1.11
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node src/server.js`
Server started on PORT: 5001...
MongoDB connected Successfully...
```

For the MongoDB String:

mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/?appName=Cluster0

By default database will be named as: **test**

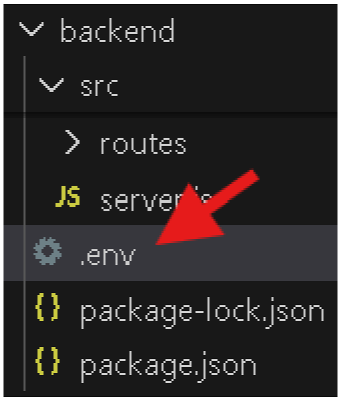
To provide the name to the database, before the question mark[?], provide the name: **notes_db**

mongodb+srv://**routhfamily123_db_user**:**dRoCgH5MOBbEmJAW**@cluster0.dxejp0q.mongodb.net/**notes_db**?appName=Cluster0

Now the database named as: notes_db

Need to hide this string, after deployment connection string is visible in the code, others can connect to the database, becouse they can see your **username: routhfamily123_db_user** and **password: dRoCgH5MOBbEmJAW** from connection string. And we can save into **.env** file. That is the use case of using **.env** file.

Create .env file under backend:



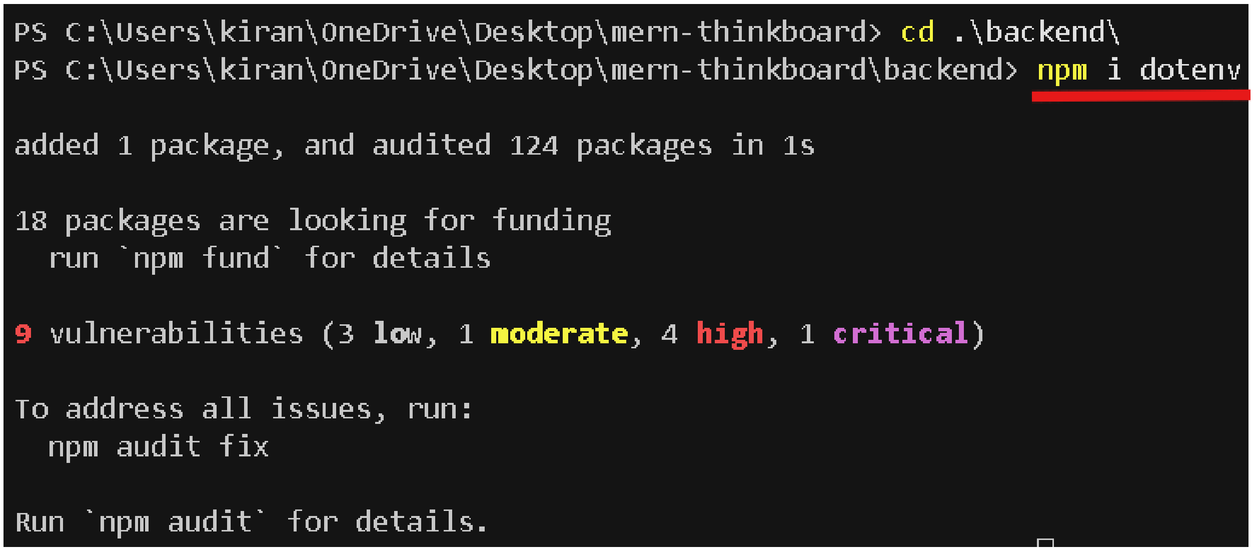
```
backend
  src
    routes
    server.js
  .env
  package-lock.json
  package.json
```

Code in .env:

MONGO_URI=mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/notes_db?appName=Cluster0

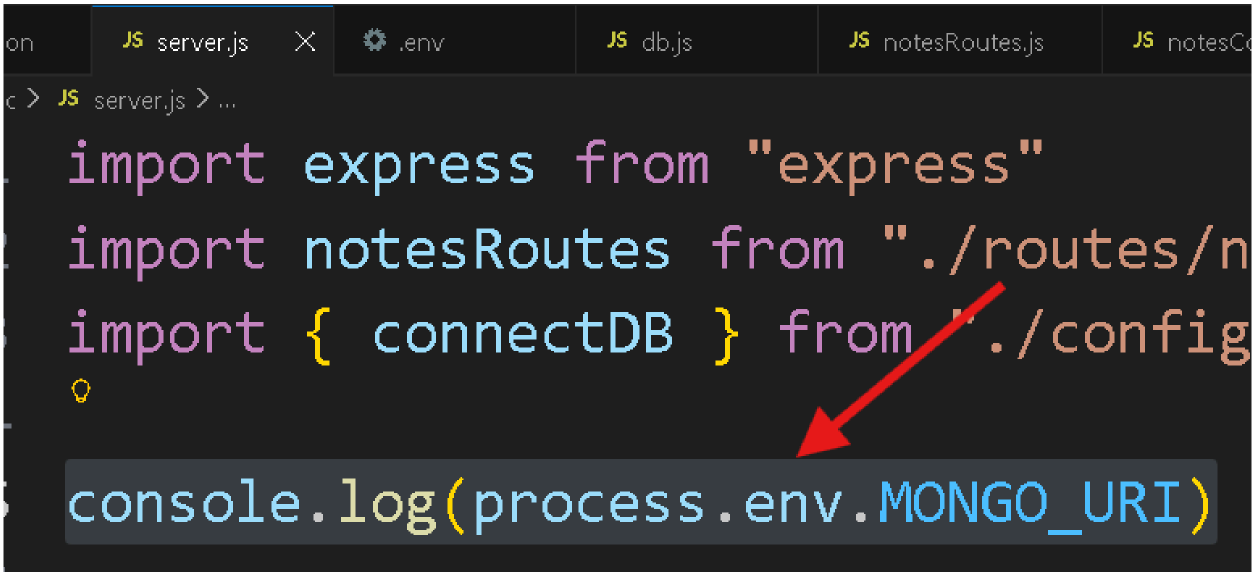
Installing a Package[dotenv]:

Command used: **npm i dotenv**



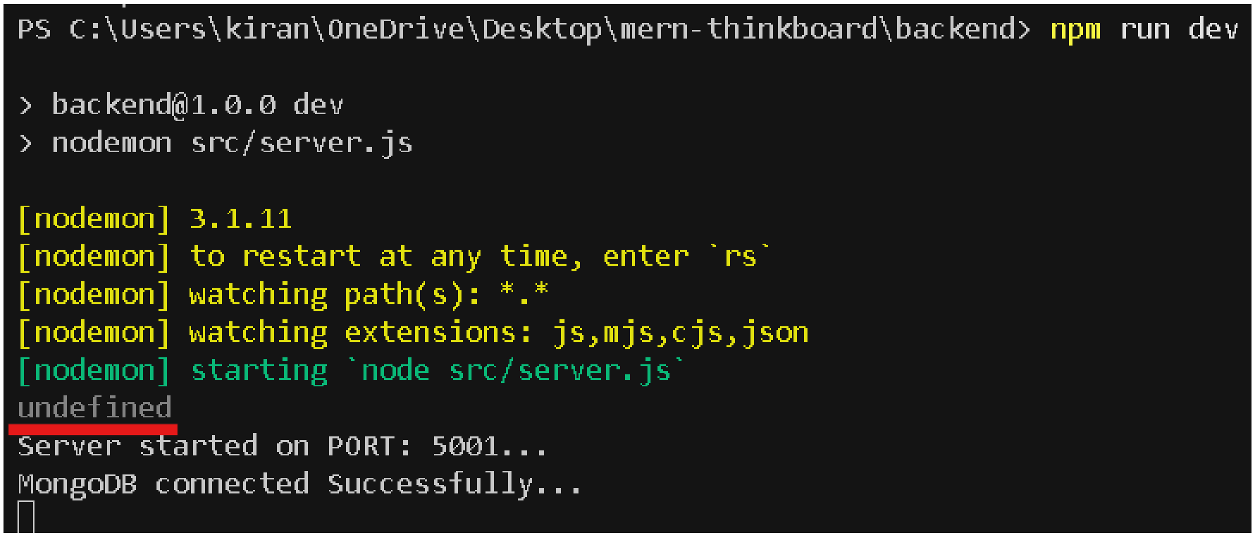
```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> cd .\backend\
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm i dotenv

added 1 package, and audited 124 packages in 1s

18 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (3 low, 1 moderate, 4 high, 1 critical)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

In server.js:
If you add: **console.log(process.env.MONGO_URI)**



```
import express from "express"
import notesRoutes from "./routes/n
import { connectDB } from "./config

console.log(process.env.MONGO_URI)
```

And run the code: shows undefined

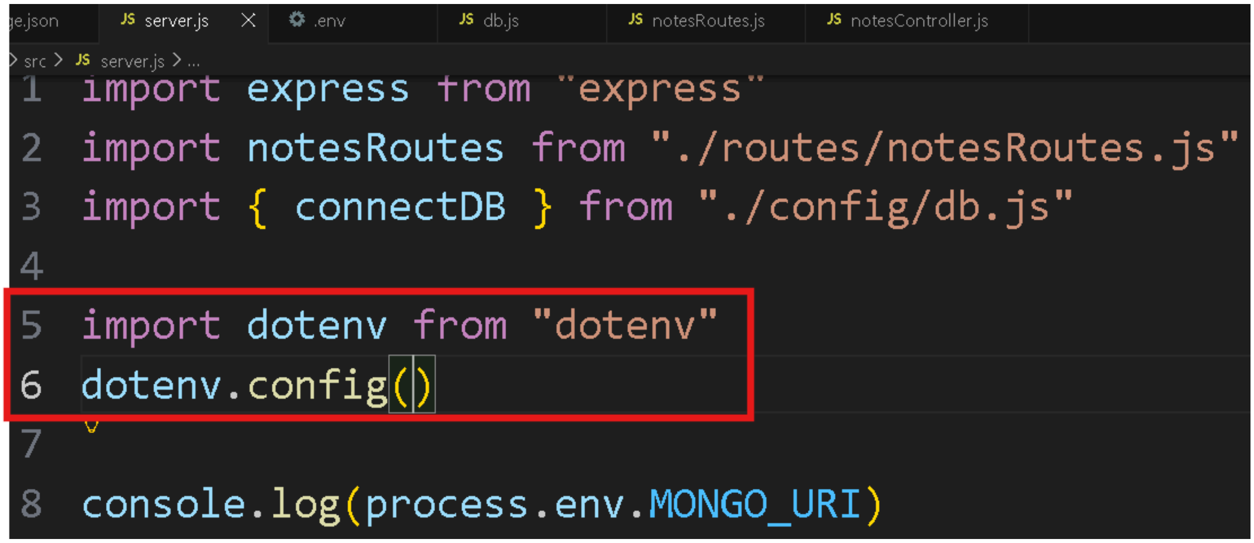


```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev

> backend@1.0.0 dev
> nodemon src/server.js

[nodemon] 3.1.11
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node src/server.js`
undefined
Server started on PORT: 5001...
MongoDB connected Successfully...
```

So to get the text:

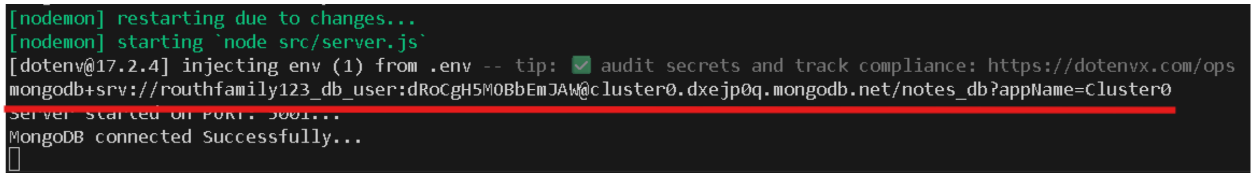
Add the code:

import dotenv from "dotenv"
dotenv.config()



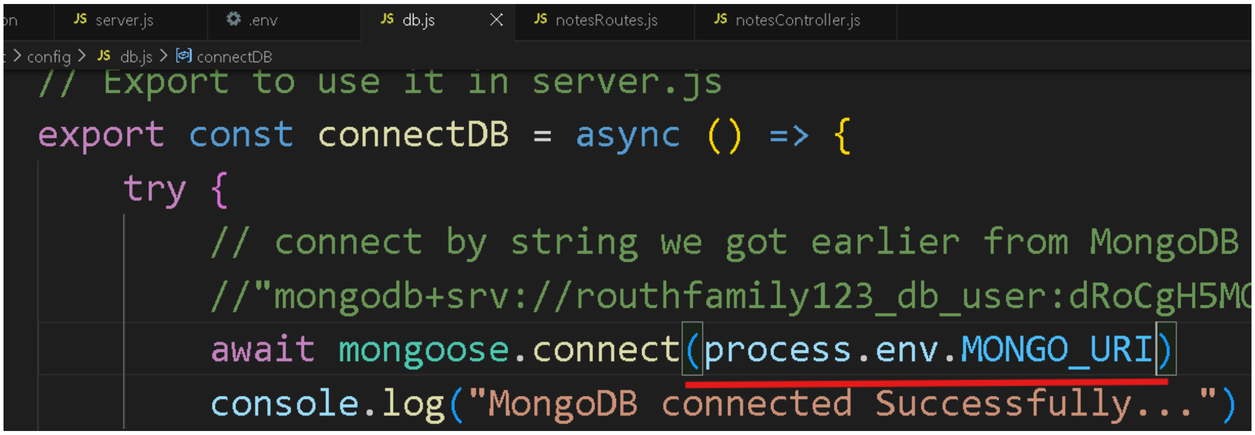
```js
import express from "express"
import notesRoutes from "./routes/notesRoutes.js"
import { connectDB } from "./config/db.js"

import dotenv from "dotenv"
dotenv.config()

console.log(process.env.MONGO_URI)
```

Got the string in terminal:



```
[nodemon] restarting due to changes...
[nodemon] starting `node src/server.js`
[dotenv@17.2.4] injecting env (1) from .env -- tip: ✓ audit secrets and track compliance: https://dotenvx.com/ops
mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/notes_db?appName=Cluster0
Server started on PORT: 5001...
MongoDB connected Successfully...
```

To get the environment variable:
Code in db.js:



```js
// Export to use it in server.js
export const connectDB = async () => {
    try {
        // connect by string we got earlier from MongoDB
        //"mongodb+srv://routhfamily123_db_user:dRoCgH5M
        await mongoose.connect(process.env.MONGO_URI)
        console.log("MongoDB connected Successfully...")
```

Code in server.js:

```
import express from "express"
import notesRoutes from "./routes/notesRoutes.js"
import { connectDB } from "./config/db.js"

import dotenv from "dotenv"
dotenv.config()

//console.log(process.env.MONGO_URI)
```

```
const app = express()

app.use("/api/notes", notesRoutes)
connectDB()

app.listen(5001, () => {
    console.log("Server started on PORT: 5001...")
})
```

Code in db.js:

```
import mongoose from "mongoose"

// Creating a function that connects to the database
// Export to use it in server.js
export const connectDB = async () => {
    try {
        // connect by string we got earlier from MongoDB

//"mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/notes_db?appName=Cluster0"
        await mongoose.connect(process.env.MONGO_URI)
        console.log("MongoDB connected Successfully...")
    } catch (error) {
        console.log("Error connecting to MongoDB:", error)
        // 1 = exit with failure, if there's an error
        process.exit(1) // 0 = success
    }
}
```

So .env will be hidden to hide secret values.
Similarly, do that for PORT.

Code in .env:

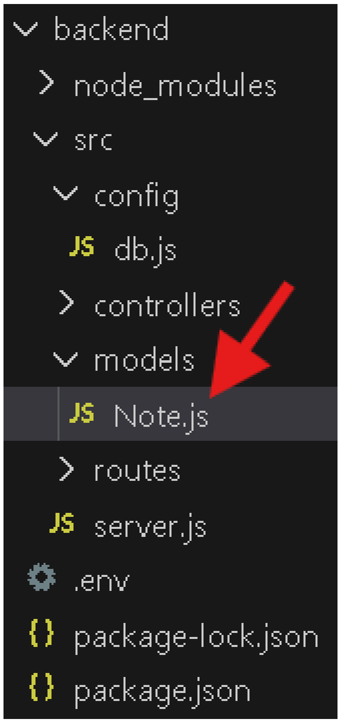
```
MONGO_URI=mongodb+srv://routhfamily123_db_user:dRoCgH5MOBbEmJAW@cluster0.dxejp0q.mongodb.net/notes_db?appName=Cluster0
PORT=5001
```

Code in server.js:

```
import express from "express"
import notesRoutes from "./routes/notesRoutes.js"
import { connectDB } from "./config/db.js"
import dotenv from "dotenv"
dotenv.config()

//console.log(process.env.MONGO_URI)

const app = express()
// if process.env.PORT is undefined then PORT = 5001(by default value)
const PORT = process.env.PORT || 5001

app.use("/api/notes", notesRoutes)
connectDB()

app.listen(PORT, () => {
  console.log("Server started on PORT:", PORT)
})
```

Create models:

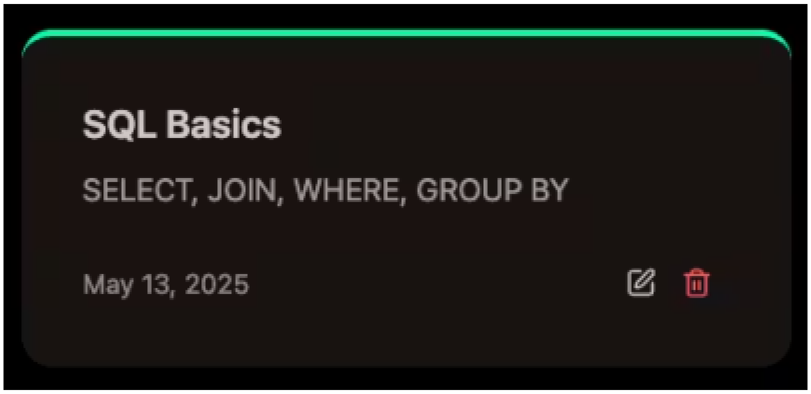
backend > src > models > Note.js



```
backend
  node_modules
  src
    config
      db.js
    controllers
    models
      Note.js
    routes
    server.js
  .env
  package-lock.json
  package.json
```

Make sure **first letter N is capital** for note.js, and **Note is singular** not like Notes.js.

Every single note is shown as:



SQL Basics

SELECT, JOIN, WHERE, GROUP BY

May 13, 2025

which has title=SQL Basics, content=SELECT, JOIN, WHERE, GROUP BY, date=May 13, 2025 etc

Code in Note.js:

```
import mongoose from "mongoose";

// 1st step: you need to create the schema
// 2nd step: you would create a model based off that schema

const noteSchema = new mongoose.Schema({
    title: {
      type:String,
      required: true
    },
    content: {
      type:String,
      required: true
    },
  },
  // for createdAt, updatedAt
  {timestamps: true}
)

// Create a Model based on the schema
// N is capital
const Note = mongoose.model("Note", noteSchema)

export default Note
```