MP3 Devlab

CS 409: Fall 2025

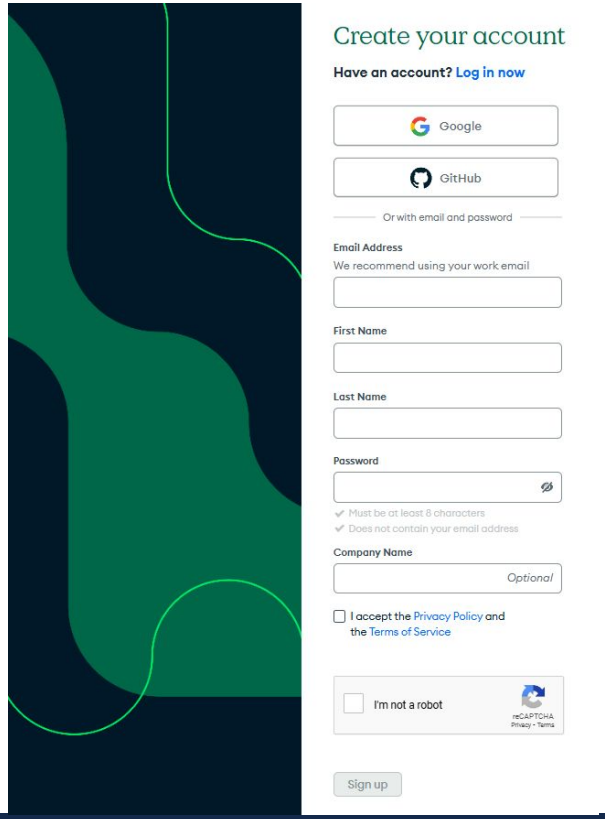
OVERVIEW

Your goal is to **create a database-backed RESTful API** using Express.js and MongoDB Atlas and deploy it on Render to make the API **publicly accessible**:

1. MongoDB Atlas
    a. Create Account
    b. Create Cluster and Database Users
    c. Get Connection String
2. Connect Express.js to MongoDB Atlas Cluster
    a. Add Connection String to **.env**
    b. Uncomment mongoose connection code in **server.js**
3. Setup deployment on Render
4. MP 3 Tips & Tricks

MongoDB Atlas Setup

Make an Account



MongoDB Atlas Setup - Create Cluster

Atlas | University of... | Access Manager | Billing | All Clusters | Get Help | Alex

Project 0 | Data Services | Charts

Overview

DATABASE

Clusters

SERVICES

Atlas Search

Stream Processing

Triggers

Migration

Data Federation

Data API

SECURITY

Backup

Database Access

Network Access

Advanced

New On Atlas 8

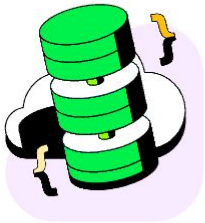
Goto

Your organization does not have a designated security contact. Add an Atlas Security Contact in Organization Settings to receive security-related notifications.

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

Overview

1. Create Cluster



Create a cluster

Choose your cloud provider, region, and specs.

+ Create

Toolbar

Featured Resources

GENERAL

Get Started with Atlas

Reference MongoDB Documentation

Develop Applications with the Developer Center

Ask the MongoDB Community

Support Plan

You are on the Basic Plan. You can view the Support page to learn more.

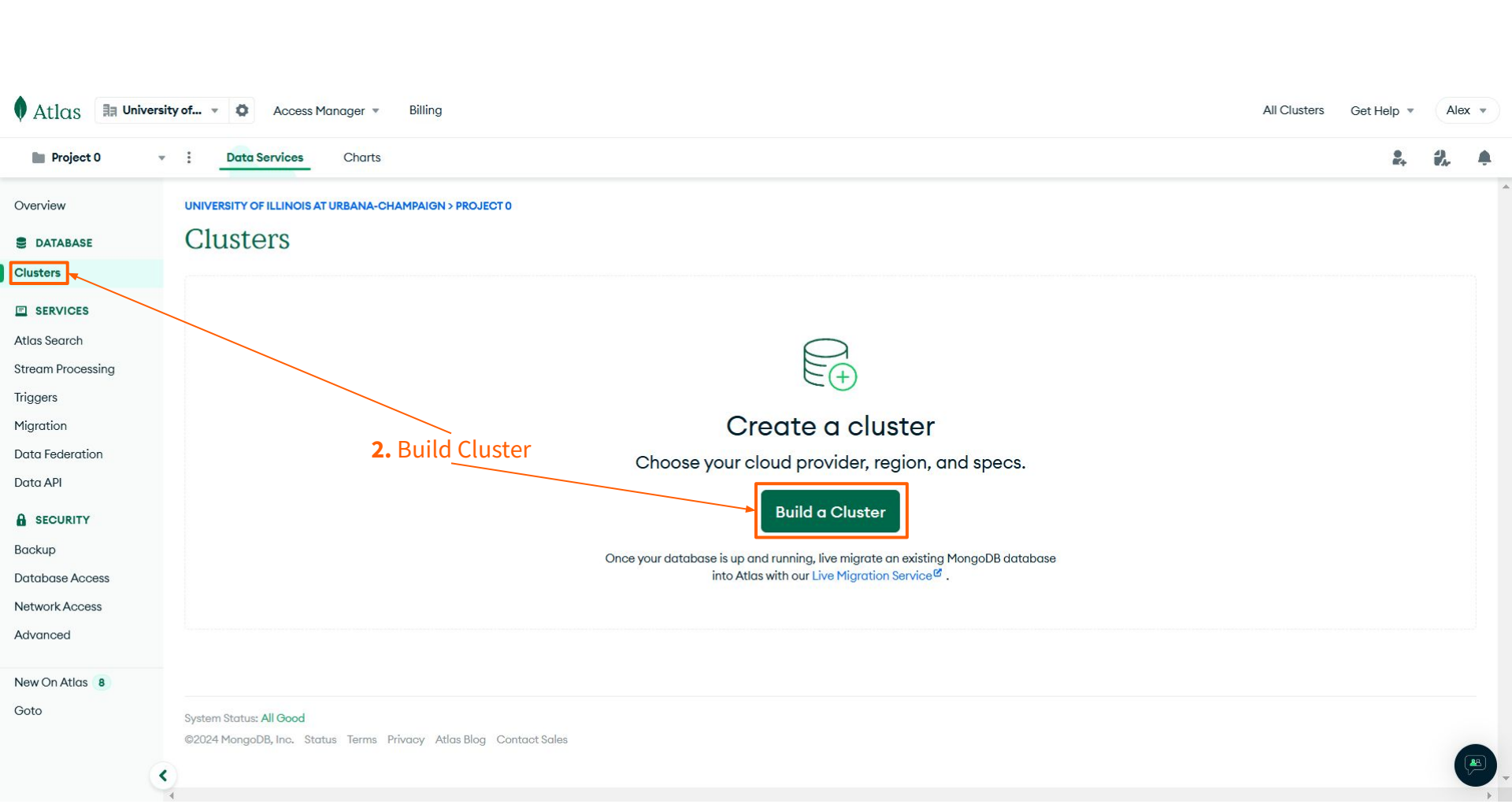
Got it

...nts on Atlas.

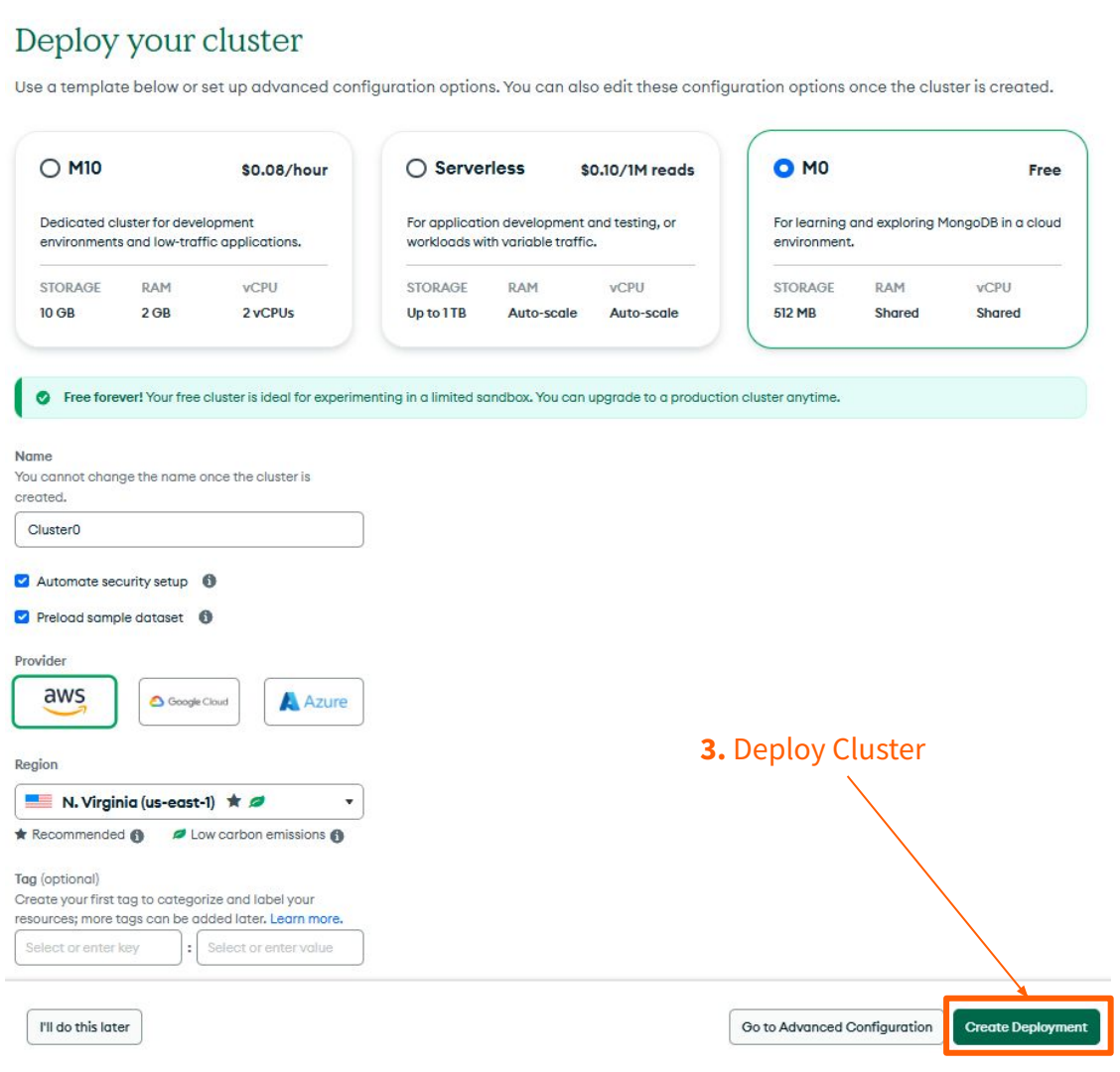
Support

Support Plan: Basic Plan

CHAT WITH SUPPORT | VIEW SUPPORT OPTIONS


Atlas
University of...
Access Manager
Billing
All Clusters
Get Help
Alex
Project 0
Data Services
Charts
Overview
DATABASE
Clusters
SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
Data API
SECURITY
Backup
Database Access
Network Access
Advanced
New On Atlas 8
Goto
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0
Clusters
2. Build Cluster
Create a cluster
Choose your cloud provider, region, and specs.
Build a Cluster
Once your database is up and running, live migrate an existing MongoDB database into Atlas with our Live Migration Service .
System Status: All Good
©2024 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

1. Basic Cluster Configuration
    - **Free** tier is sufficient
    - **Name** doesn't matter for MP3
    - **Automate Security Setup** to help protect cluster from public access
    - **Preload Sample Dataset** is optional
    - **Provider** doesn't matter for MP3
    - **Region** doesn't matter for MP3
    - **Tag** doesn't matter for MP3
2. **Ignore** the following buttons
    - "I'll do this later"
    - "Go to Advanced Configuration"
3. Finally click "**Create Deployment**"



Atlas | University of... | Access Manager | Billing | All Clusters | Get Help | Alex

Project 0 | Data Services | Charts

Overview

DATABASE
Clusters

SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
Data API

SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced

New On Atlas 8
Goto

We are deploying your changes (current action: creating a plan)

Your organization does not have a designated security contact. Add an Atlas Security Contact in Organization Settings to receive security-related notifications.

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

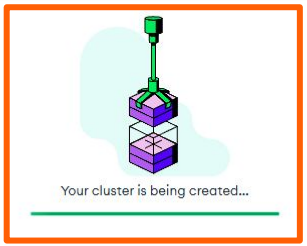
Overview

Clusters

4. Wait (This can take a few minutes…)

Create cluster

Cluster0



+ Add Tag

Toolbar

Featured Resources

JAVASCRIPT / NODE.JS
Connect to your data with Node.js / JavaScript
Query your data: CRUD with Node.js
Mongoose Quickstart
Integrate with Next.js and Vercel
More JavaScript Content

LEARN
Sample Apps Repo
Relevance-based apps with Search
Semantic Search with Vector Search

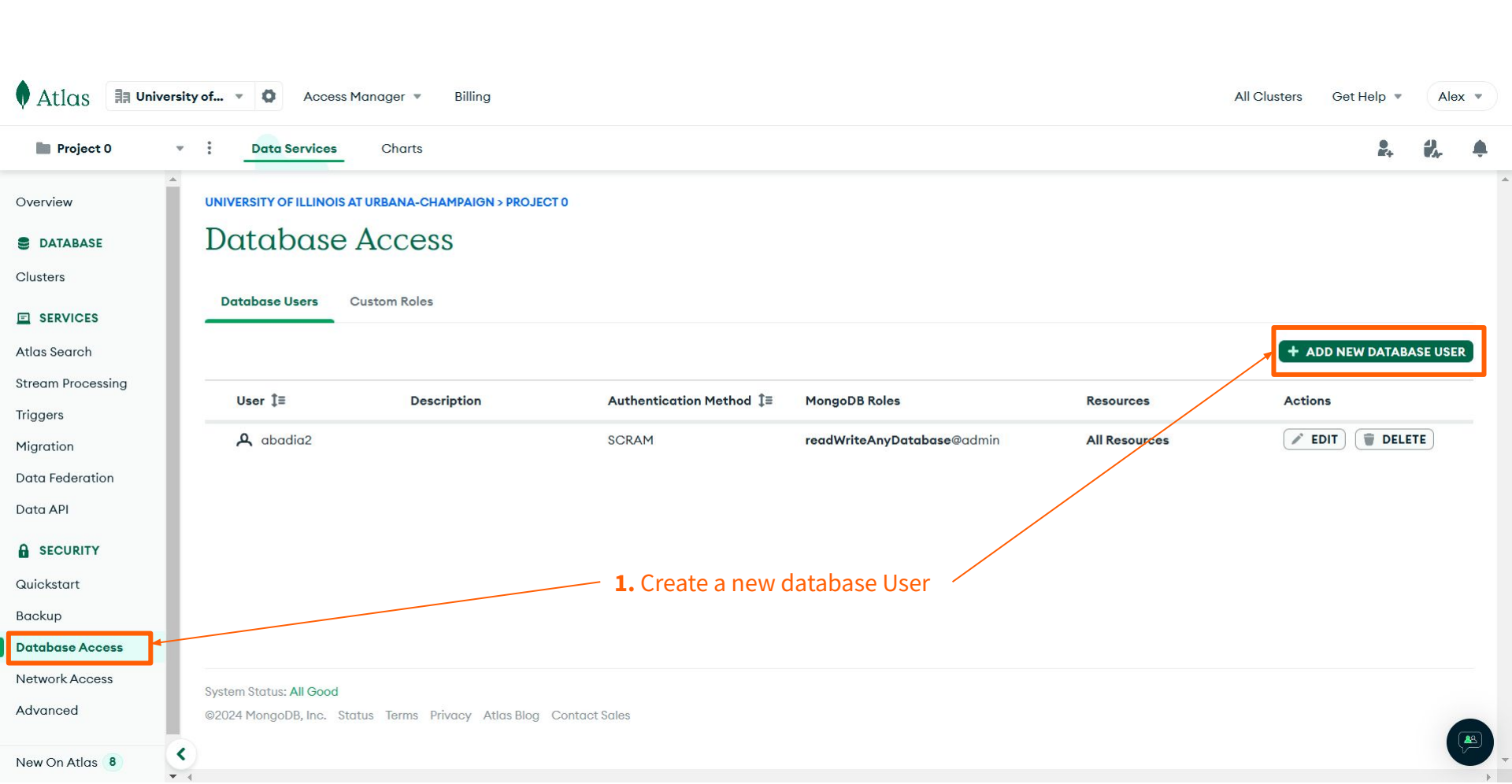
Sample Apps

JAVASCRIPT / NODE.JS
MERN Stack
MEAN Stack
Remix Stack
Search in JavaScript
Change Stream Publishing

New On Atlas

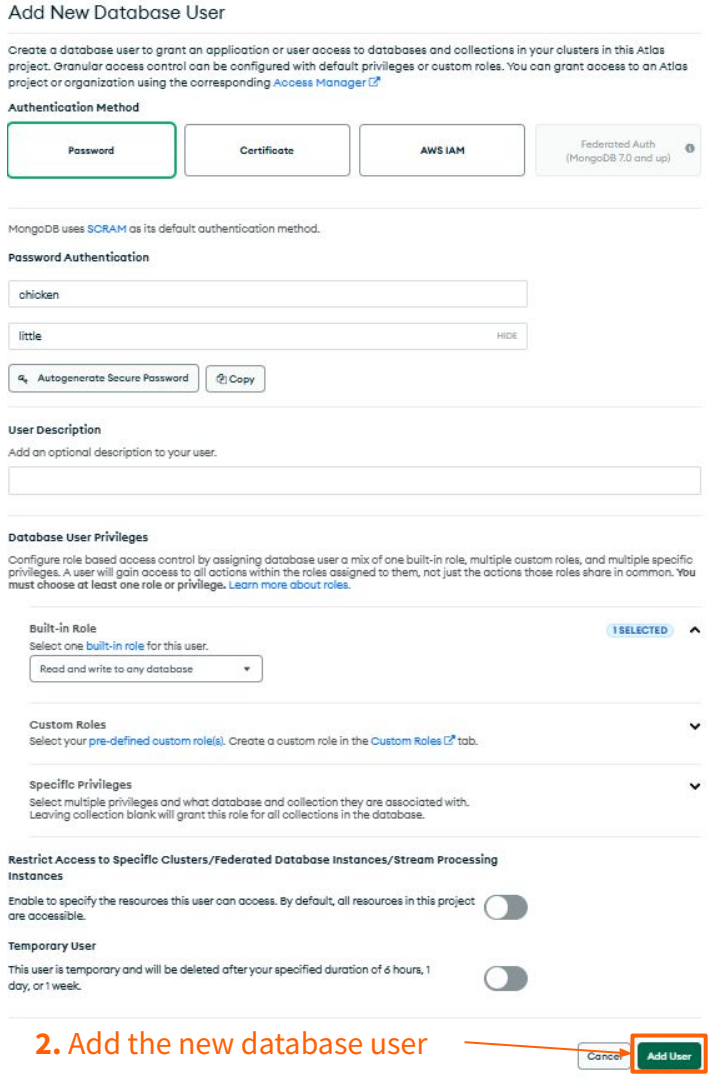
8 NEW
Learn about the latest feature enhancements on Atlas.

MongoDB Atlas Setup - Create User

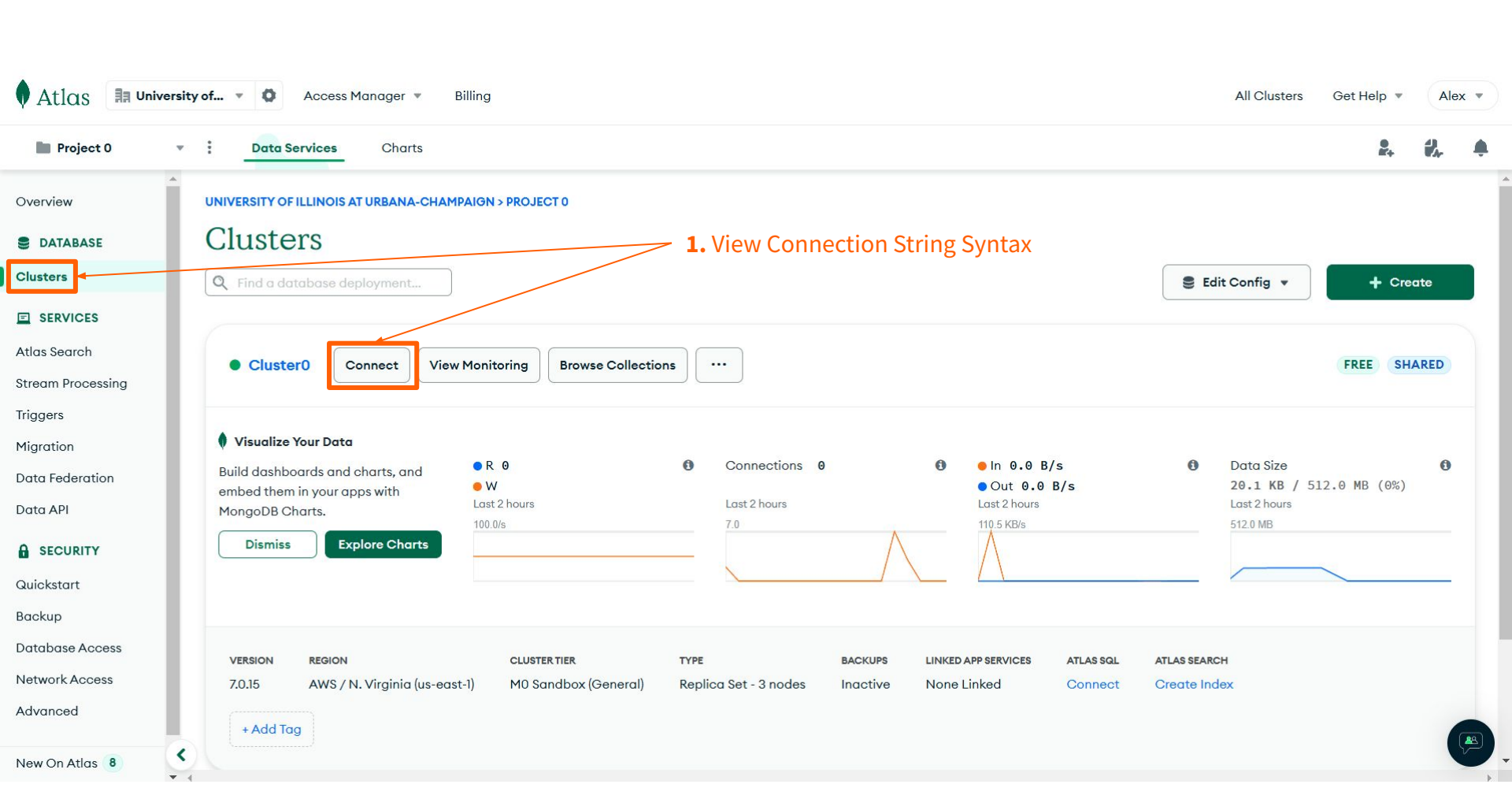

Atlas
University of...
Access Manager
Billing
All Clusters
Get Help
Alex
Project 0
Data Services
Charts
Overview
DATABASE
Clusters
SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
Data API
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
New On Atlas 8
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0
Database Access
Database Users
Custom Roles
+ ADD NEW DATABASE USER
User
Description
Authentication Method
MongoDB Roles
Resources
Actions
abadia2
SCRAM
readWriteAnyDatabase@admin
All Resources
EDIT
DELETE
1. Create a new database User
System Status: All Good
©2024 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

1. New Database User Configuration
    - **Password**-based **authentication method** is sufficient
    - **Remember** the **username** and **password** for later:
        i. Username: chicken
        ii. Password: little
    - **Built-in-role** is “read and write to any database”
2. **Ignore** the rest to keep things simple
3. Finally click “**Add user**”

Note: Useful if you forgot the username and password of the original user created when making a MongoDB Atlas account!



MongoDB Atlas Setup - View Connection String Syntax


Atlas
University of...
Access Manager
Billing
All Clusters
Get Help
Alex
Project 0
Data Services
Charts
Overview
DATABASE
Clusters
SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
Data API
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
New On Atlas 8
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0
Clusters
1. View Connection String Syntax
Find a database deployment...
Edit Config
Create
Cluster0
Connect
View Monitoring
Browse Collections
FREE
SHARED
Visualize Your Data
Build dashboards and charts, and embed them in your apps with MongoDB Charts.
Dismiss
Explore Charts
R 0
W
Last 2 hours
100.0/s
Connections 0
Last 2 hours
7.0
In 0.0 B/s
Out 0.0 B/s
Last 2 hours
110.5 KB/s
Data Size
20.1 KB / 512.0 MB (0%)
Last 2 hours
512.0 MB
VERSION
7.0.15
REGION
AWS / N. Virginia (us-east-1)
CLUSTER TIER
M0 Sandbox (General)
TYPE
Replica Set - 3 nodes
BACKUPS
Inactive
LINKED APP SERVICES
None Linked
ATLAS SQL
Connect
ATLAS SEARCH
Create Index
+ Add Tag

Connect to Cluster0

Set up connection security — 2 Choose a connection method — 3 Connect

2. Click That →

Connect to your application

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

Access your data through tools

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

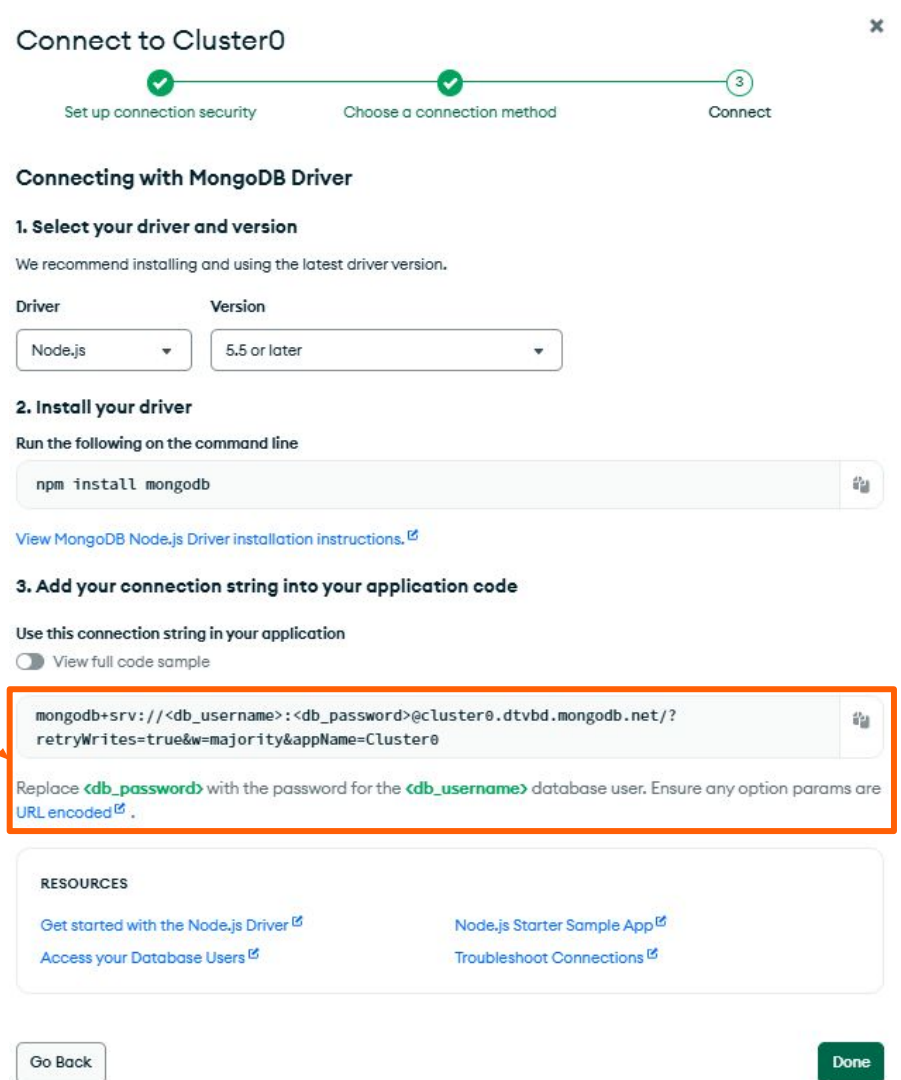
Atlas SQL
Easily connect SQL tools to Atlas for data analysis and visualization

Go Back

Close

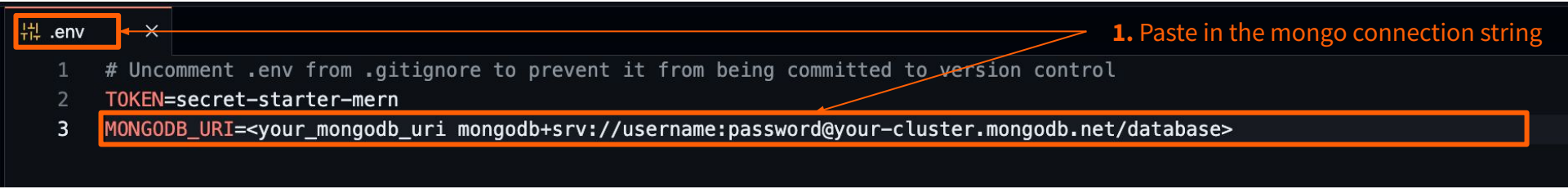
3. Remember Connection String Syntax
    - **<db_username>** is a database user's username
    - **<db_password>** is a database user's password
    - Notice that the **cluster name** is included
    - Notice **url parameters** can be added set connection configurations/behaviors
    - For example
        i. mongodb+srv://**chicken**:**little**@**cluster0**.dtvbd.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0

Note: This will be used for configuring the Express.js application and deployment on Render



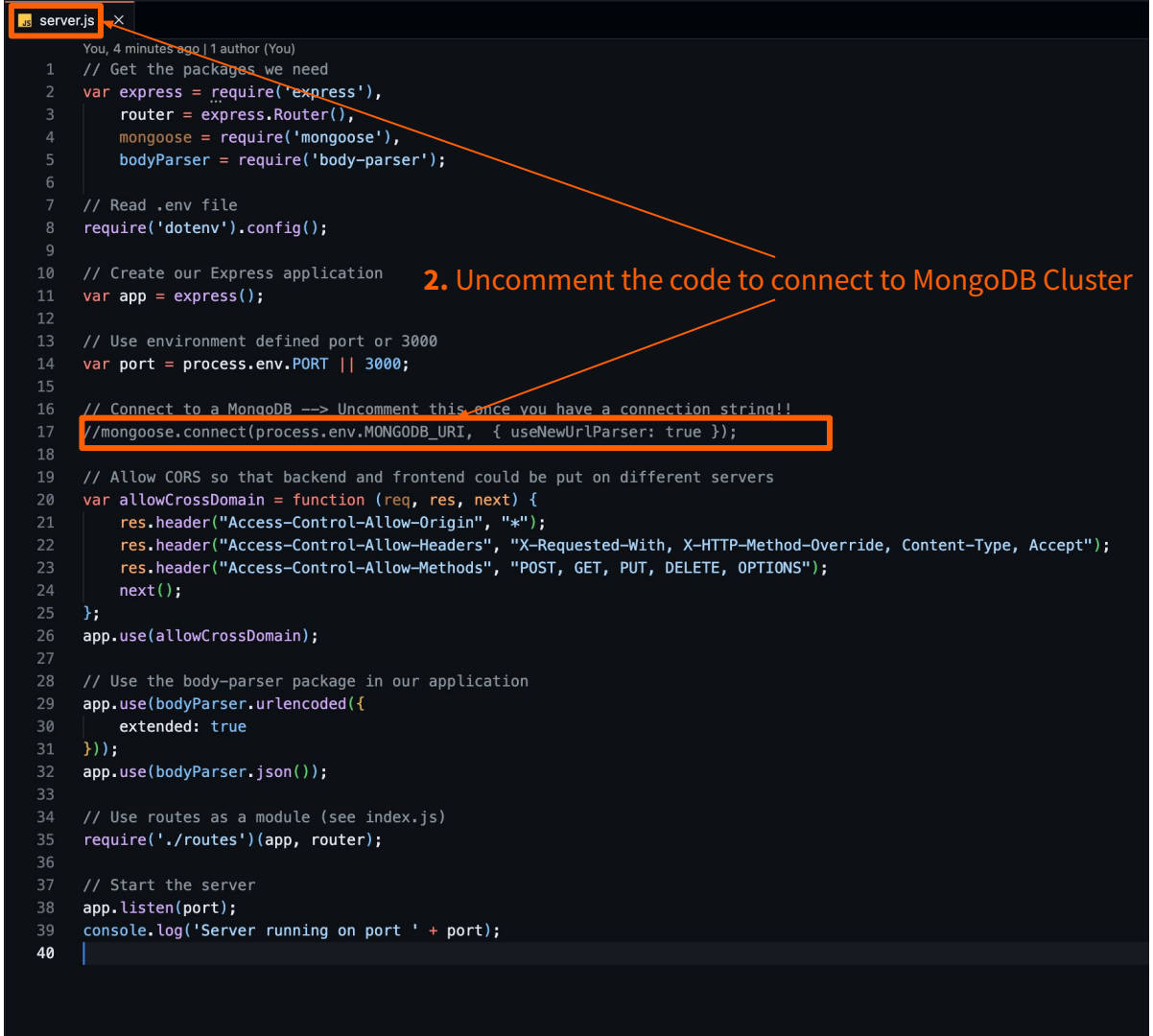
MP3 Express.js Setup

Environment Variables



Environment Variables are usually configuration variables or secrets like API keys or database strings.

You should never expose ENVs by committing them to your repository.


server.js
You, 4 minutes ago | 1 author (You)
// Get the packages we need
var express = require('express'),
router = express.Router(),
mongoose = require('mongoose'),
bodyParser = require('body-parser');
// Read .env file
require('dotenv').config();
// Create our Express application
var app = express();
// Use environment defined port or 3000
var port = process.env.PORT || 3000;
// Connect to a MongoDB --> Uncomment this once you have a connection string!!
//mongoose.connect(process.env.MONGODB_URI, { useNewUrlParser: true });
// Allow CORS so that backend and frontend could be put on different servers
var allowCrossDomain = function (req, res, next) {
res.header("Access-Control-Allow-Origin", "*");
res.header("Access-Control-Allow-Headers", "X-Requested-With, X-HTTP-Method-Override, Content-Type, Accept");
res.header("Access-Control-Allow-Methods", "POST, GET, PUT, DELETE, OPTIONS");
next();
};
app.use(allowCrossDomain);
// Use the body-parser package in our application
app.use(bodyParser.urlencoded({
extended: true
}));
app.use(bodyParser.json());
// Use routes as a module (see index.js)
require('./routes')(app, router);
// Start the server
app.listen(port);
console.log('Server running on port ' + port);
2. Uncomment the code to connect to MongoDB Cluster


.gitignore M
You, 1 second ago | 1 author (You)
3. Uncomment to add .env to .gitignore
1 .DS_Store
2 .idea/
3 .idea/*
4 .vs_code/
5 .vs_code/*
6 node_modules
7 node_modules/*
8 public/
9 public/*
10 # .env
11

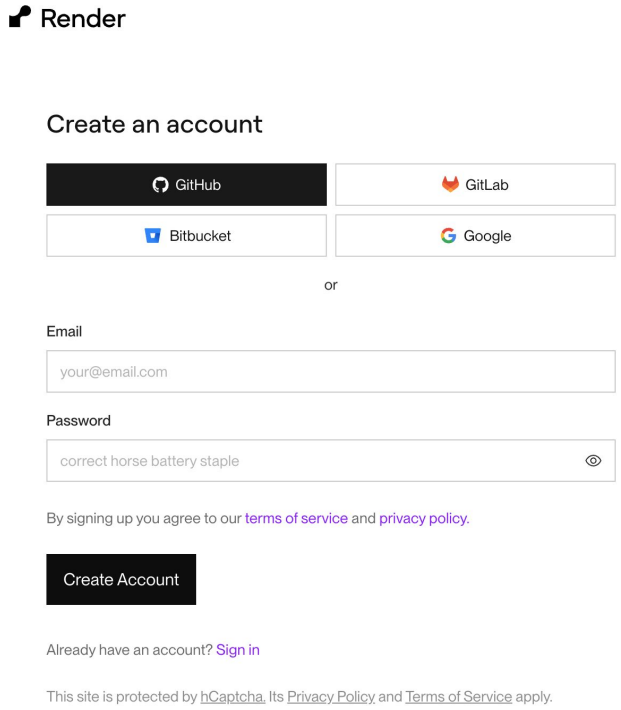
Deployment on Render

Make an Account

Render.com is a cloud platform that simplifies building, deploying, and scaling web applications, static sites and databases.

It offers automated builds and deployments on push by connecting to your GitHub/GitLab/BitBucket repository.

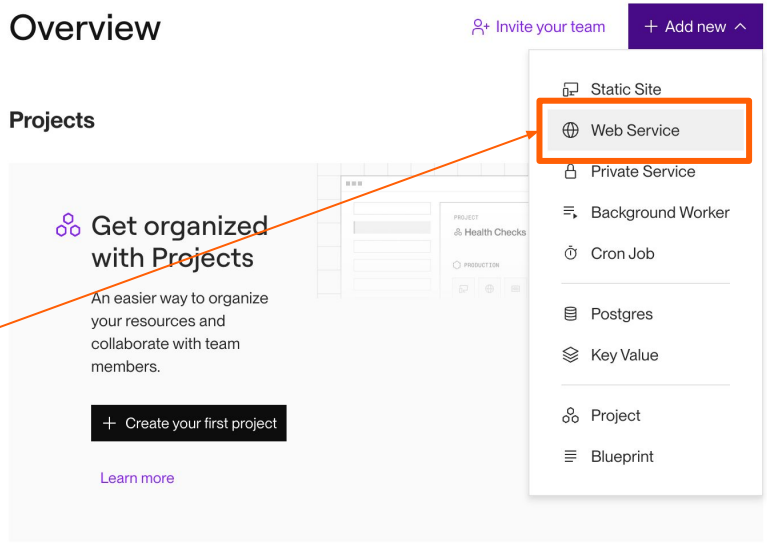
1. Navigate to render.com
2. Create an account
    - Preferably using **GitHub**



Add a new project

We will be using Render to deploy our Express.js server as a publicly accessible API.

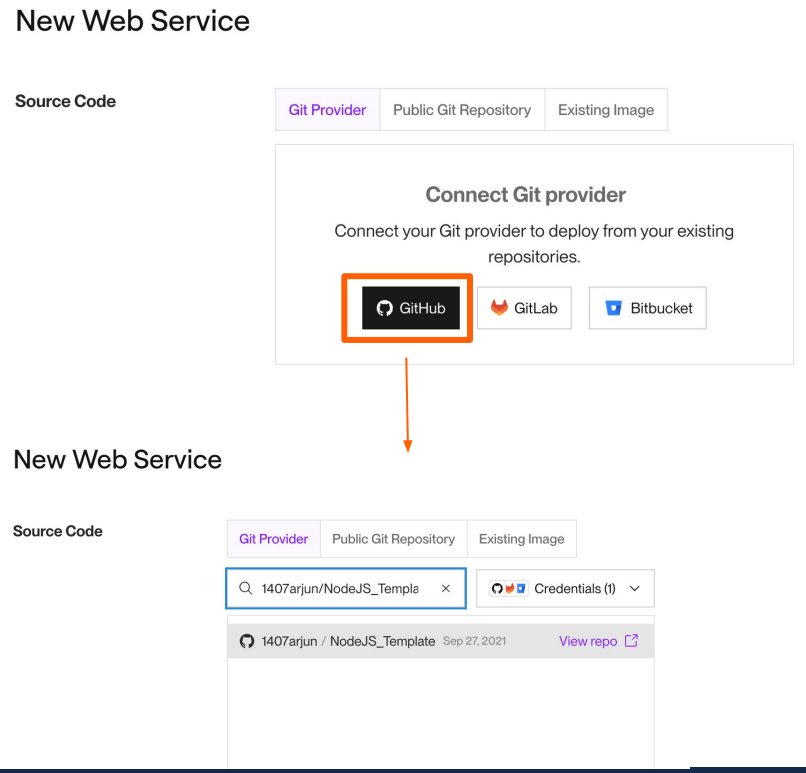
1. Choose **Static Site** for frontend applications, similar to GitHub pages.
    - React, Vue, plain HTML/CSS/JS, etc.
2. Choose **Web Service** for backend or server-side applications.
    - Node.js, Express.js, Django, etc.
3. **Click on web service for MP3.**



Add a new project

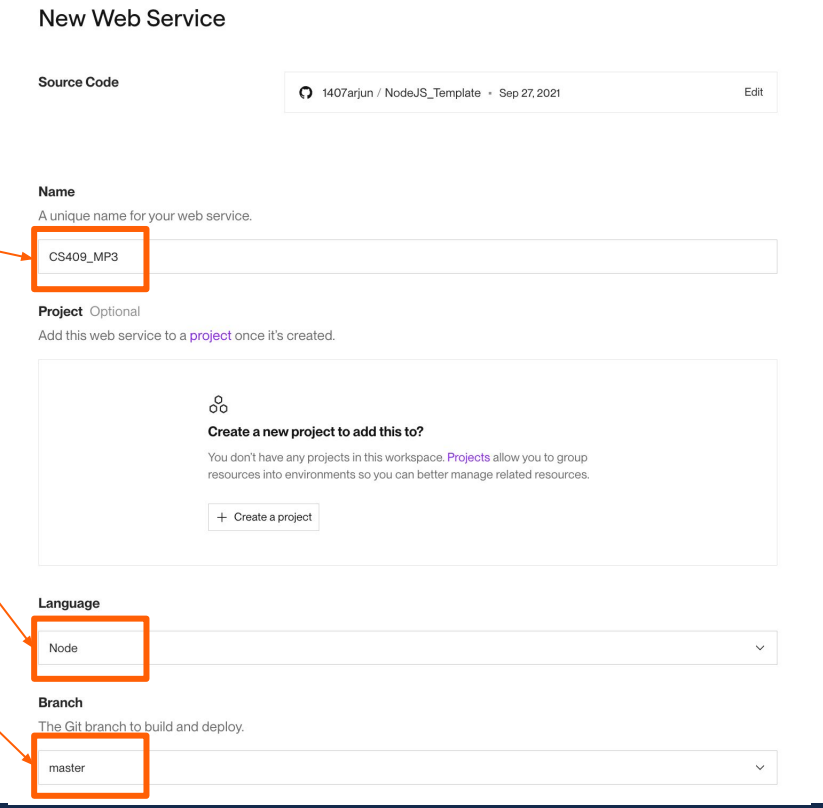
1. Choose the provider where your repository is located (**GitHub** in our case).
2. In the popup that follows, provide permissions to all the repositories that you'd want to deploy OR provide complete access.
3. Once connected, choose the repository to deploy from the dropdown.

Note: This step enables Render to automatically build and deploy your application whenever there is push to your repository.



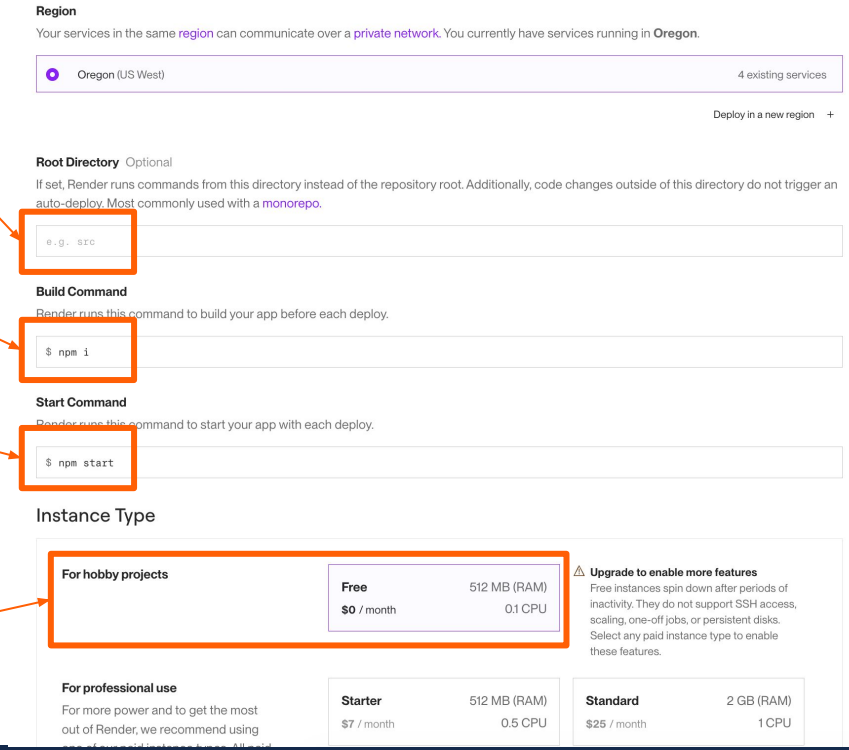
Setup your project

1. Provide a name for your deployment
   - This can be anything, not necessarily the name of the repository.
2. Choose the language of the framework that you are using, Node in our case.
3. Select the branch you would want to deploy.
   - It automatically selects the default branch.
   - main or master (for older repositories)



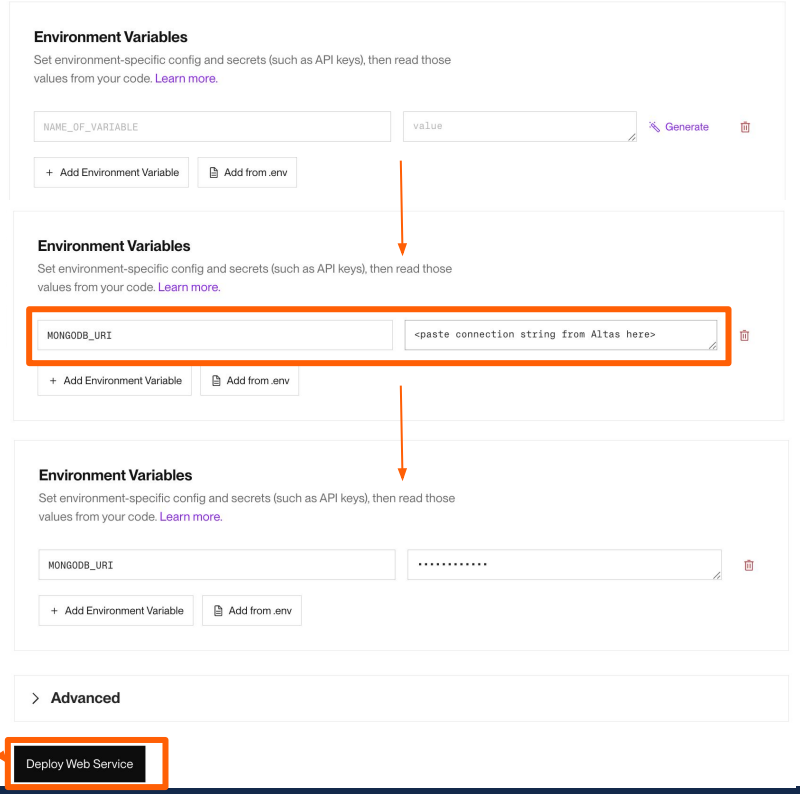
Setup your Node project

1. Root directory should be set to the location of the server.js/index.js and package.json, if not in the root.
2. Change the build command to “**npm i**”.
   - “**yarn**” if using Yarn as the package manager.
3. Change the start command to “**npm start**”.
   - Should be present in your package.json.
   - “**yarn start**” if using Yarn as the package manager.
4. Select the Free instance type, if not already selected



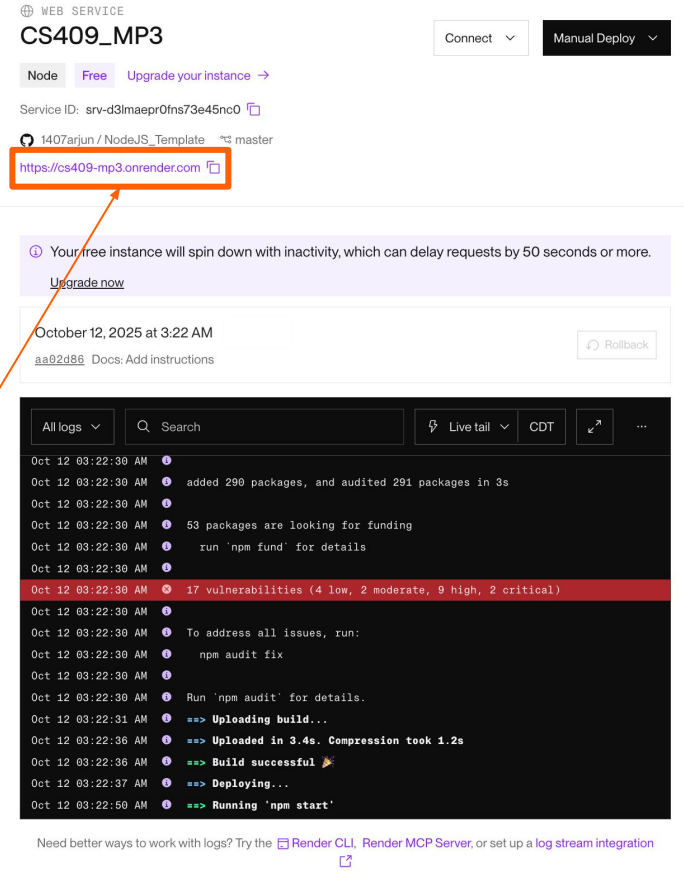
Environment Variables

1. Environment Variables are usually configuration variables or secrets like API keys or database strings.
    - **You should never expose ENVs by committing them to your repository.**
2. Assign a name to the env for the MongoDB connection string and paste the connection string from Atlas as its value.
3. Finally click **deploy** to complete the process.
    - You can ignore the advanced options for now to keep it simple.

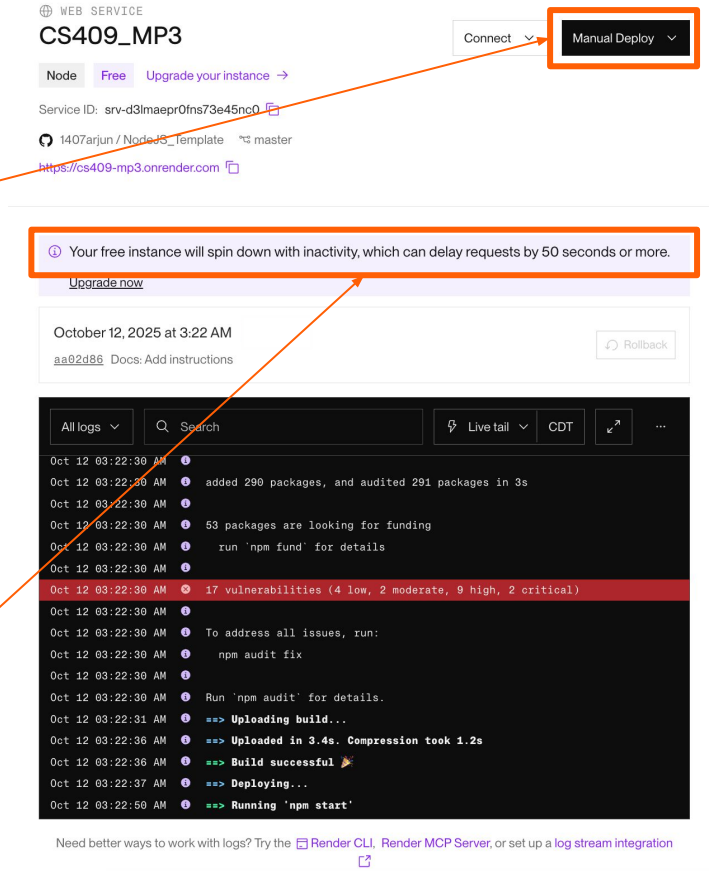


View deployed service

1. You will be redirect to your project homepage on clicking deploy on the previous step.
2. You will be able to view your deployed service on this link.
    - You might have to wait, while the service is deployed completely.
    - **This is the link to be submitted for the MP3 deployment.**
3. If everything is successful, the API would be accessible on the above link without throwing a 404 status.



View deployed service

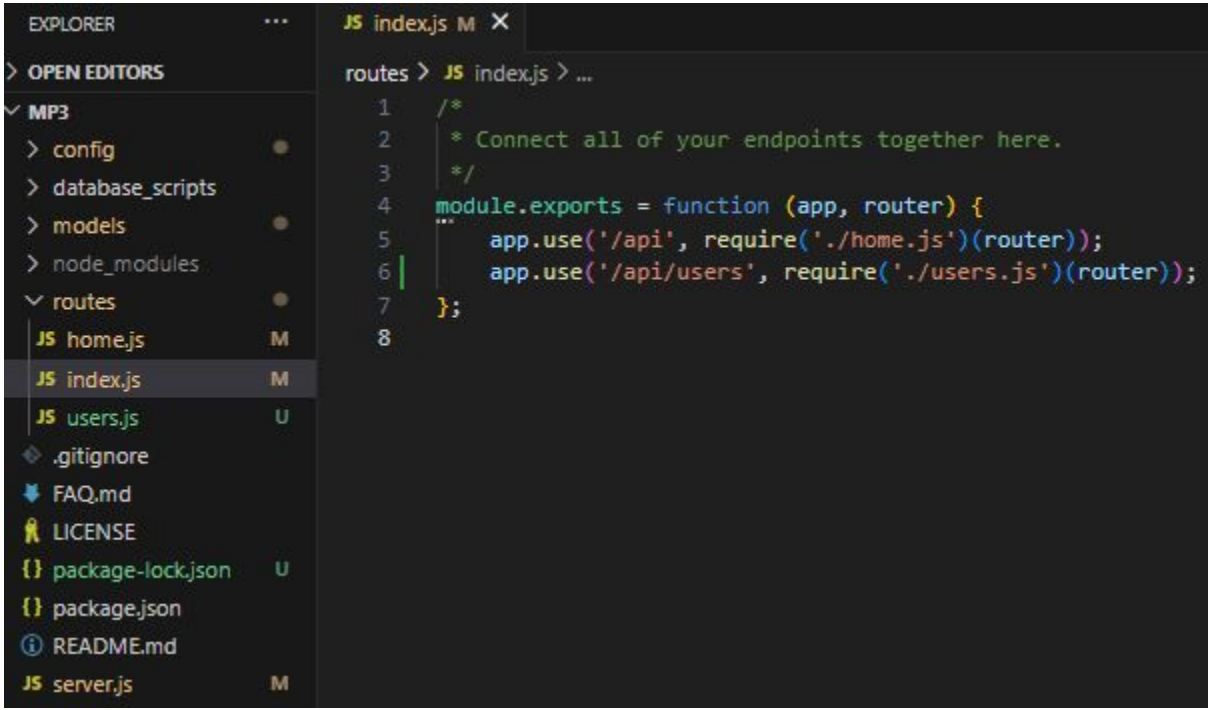


Notes:

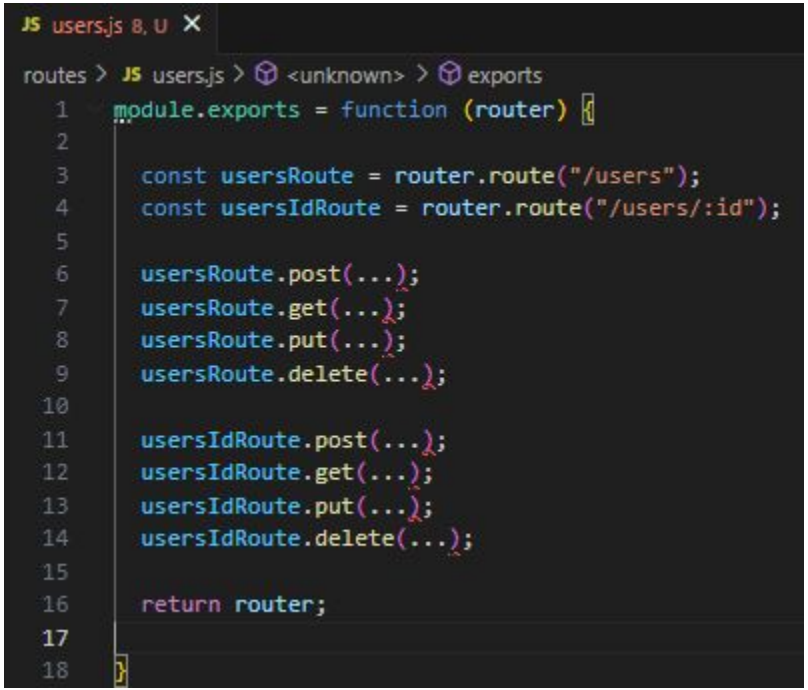
1. Be sure to **redeploy** your service on Render, whenever you **make a change to the environment variables (ENVs)**.
    - Choose to deploy the latest commit.
2. Render automatically suspends the instance, if requests are not made for a while. **This is normal**, and would just increase the API response time for the first time when used after inactivity.

MP3 Tips & Tricks

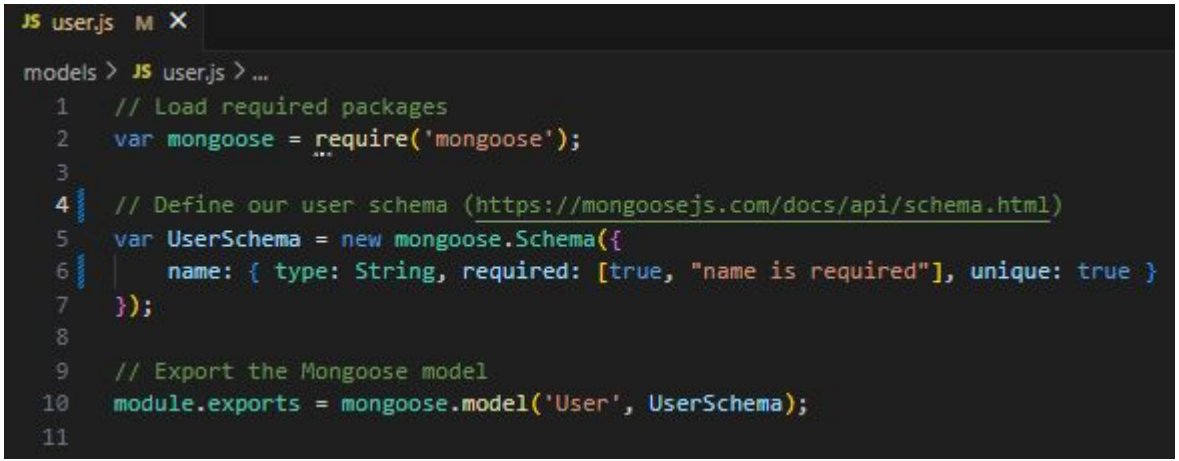
Multiple Files For Each Endpoint



Endpoint Multiple Routers



Mongoose Schema Validation



Concurrency & Transactions

```
9   usersRoute.post(async function (req, res) {
10
11    // Use mongoose for schema validation
12    const newUser = new User(req.body);
13    const err = newUser.validateSync();
14
15    if (err) {
16        // TODO: Handle Errors and responses...
17        return;
18    }
19
20    try {
21      // Mongoose will commit the transaction if the async function
22      // executes successfully and attempt to retry if there was a retriable error.
23      await User.db.transaction(async (session) => {
24
25        // Multiple db queries (i.e re-assigning tasks) can have concurrency issues
26        //   Lost updates, Inconsistent retrieval, etc
27        const savedUser = await newUser.save();
28      });
29    } catch (e) {
30      // ...
31      return;
32    }
33    // ...
34  });
```

Mongoose Query Builder

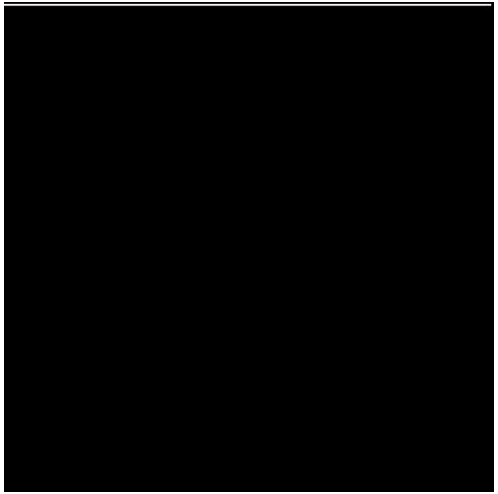
```
36  usersRoute.get(async function (req, res) {
37
38    // mongoose query builder
39    const query = User.find();
40    query.collection(User.collection);
41
42    // TODO: Add query conditions (where, limit, etc)
43    //   if (req.query["where"]) { query.where(...); }
44
45    try {
46      const result = await query.exec();
47      res.status(200).json({data: result});
48    } catch (err) {
49      res.status(500).json({data: err});
50    }
51  });
```

Mongoose Query Builder & Reading URL Params

```
53  usersIdRoute.get(async function (req, res) {
54
55    const userId = req.params["id"];
56    const query = User.findById(userId);
57    query.collection(User.collection);
58
59    // TODO: Add query conditions (where, limit, etc)
60    //   if (req.query["where"]) { query.where(...); }
61
62    try {
63      const result = await query.exec();
64      res.status(200).json({data: result});
65    } catch (err) {
66      res.status(500).json({data: err});
67    }
68
69  })
```

On Artificial Intelligence

- Artificial intelligence can make you feel like superman
  - Seek to understand the output otherwise you rack up…
  - **Knowledge Debt**
    - All the “unimportant” details that are overlooked
    - “*I’ll look into that later*”
      - *proceeds to copy paste*
    - Amount varies based on your background knowledge
- Knowledge Debt Consequences
  - All that knowledge debt catches up when…
    - Something breaks
    - You have a *midterm*…
- Don’t forget to submit your LLM logs!



RESOURCES

- https://expressjs.com/
- https://www.mongodb.com/
- https://mongoosejs.com/docs/index.html
- https://render.com/