



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



Create your account

Have an account? [Log in now](#)



Or with email and password

Email Address

We recommend using your work email

First Name

Last Name

Password

Must be at least 8 characters

Does not contain your email address

Company Name

 Optional

I accept the [Privacy Policy](#) and
the [Terms of Service](#)

I'm not a robot

[Sign up](#)



MongoDB Atlas Setup - Create Cluster



Overview

DATABASE

Clusters

SERVICES

Atlas Search

Stream Processing

Triggers

Migration

Data Federation

Data API

SECURITY

Backup

Database Access

Network Access

Advanced

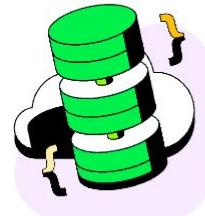
New On Atlas 8

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

Support

Support Plan: Basic Plan

CHAT WITH SUPPORT**VIEW SUPPORT OPTIONS**

Project 0

Data Services

Charts



Overview

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

Clusters

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



Create a cluster

Choose your cloud provider, region, and specs.

Build a Cluster

2. Build 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

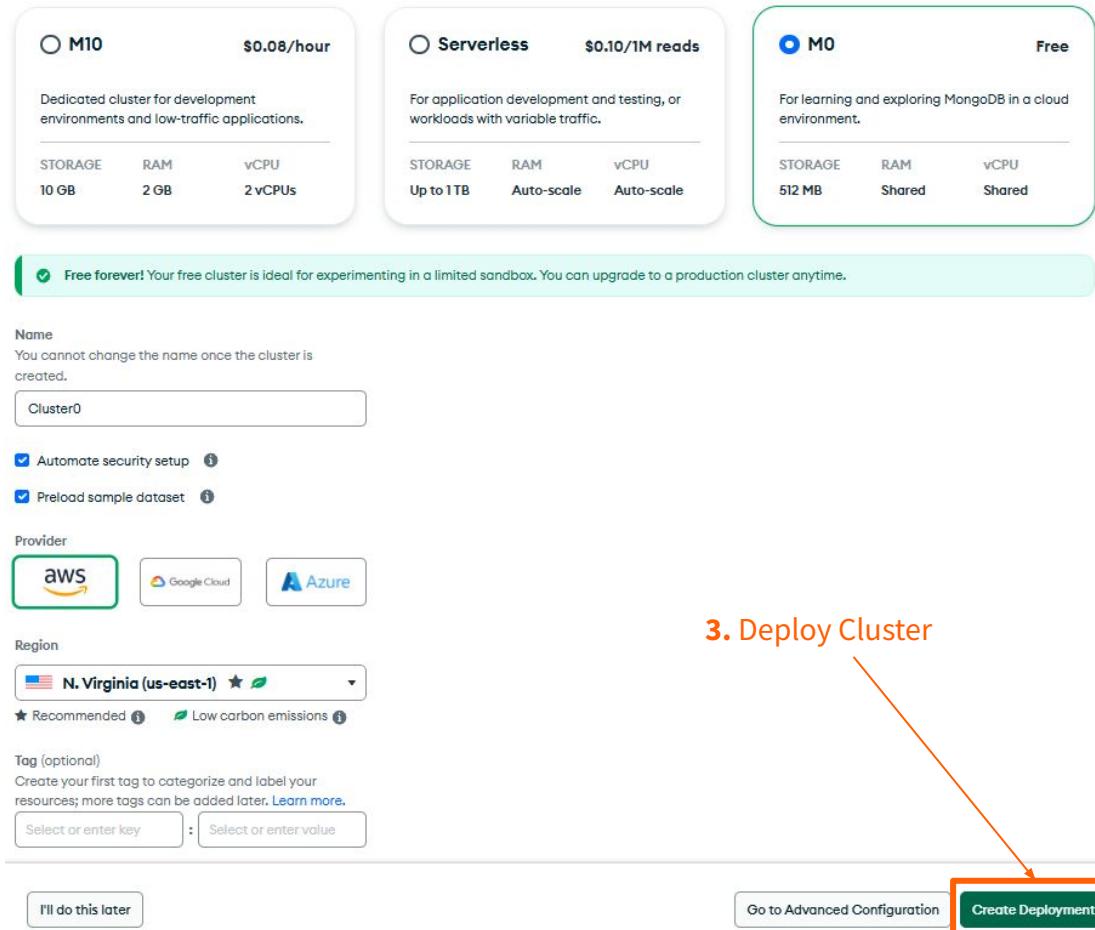
©2024 MongoDB, Inc. [Status](#) [Terms](#) [Privacy](#) [Atlas Blog](#) [Contact Sales](#)

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.

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



M10 \$0.08/hour
Dedicated cluster for development environments and low-traffic applications.
STORAGE RAM vCPU
10 GB 2 GB 2 vCPUs

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

M0 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.

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

Automate security setup ⓘ
 Preload sample dataset ⓘ

Provider
 aws Google Cloud Azure

Region

★ Recommended ⓘ 🌱 Low carbon emissions ⓘ

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

I'll do this later

2. Ignore the following buttons

- “I’ll do this later”
- “Go to Advanced Configuration”

3. Finally click “Create Deployment”

3. Deploy Cluster

Project 0

Data Services

Charts



Overview

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

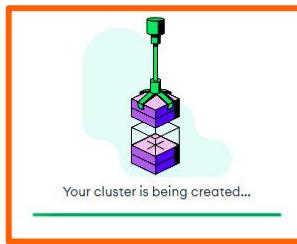
Your organization does not have a designated security contact. Add an Atlas Security Contact in [Organization Settings](#) to receive security-related notifications.



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

Overview

Clusters

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

Your cluster is being created...

Create cluster

...

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

Project 0

Data Services

Charts



Overview

DATABASE

Clusters

SERVICES

Atlas Search

Stream Processing

Triggers

Migration

Data Federation

Data API

SECURITY

Quickstart

Backup

Database Access

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

Database Access

Database Users

Custom Roles

User	Description	Authentication Method	MongoDB Roles	Resources	Actions
abadia2		SCRAM	readWriteAnyDatabase@admin	All Resources	<button>EDIT</button> <button>DELETE</button>

+ ADD NEW DATABASE USER

1. Create a new database User

System Status: All Good

©2024 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

New On Atlas 8



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!

Add New Database User

Create a database user to grant an application or user access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding [Access Manager](#).

Authentication Method

MongoDB uses **SCRAM** as its default authentication method.

Password Authentication

chicken
little HIDE

User Description
Add an optional description to your user.

Database User Privileges
Configure role based access control by assigning database user a mix of one built-in role, multiple custom roles, and multiple specific privileges. A user will gain access to all actions within the roles assigned to them, not just the actions those roles share in common. You must choose at least one role or privilege. [Learn more about roles](#).

Built-in Role 1 SELECTED

Select one [built-in role](#) for this user.

Custom Roles
Select your [pre-defined custom role\(s\)](#). Create a custom role in the [Custom Roles](#) tab.

Specific Privileges
Select multiple privileges and what database and collection they are associated with. Leaving collection blank will grant this role for all collections in the database.

Restrict Access to Specific Clusters/Federated Database Instances/Stream Processing Instances
Enable to specify the resources this user can access. By default, all resources in this project are accessible.

Temporary User
This user is temporary and will be deleted after your specified duration of 6 hours, 1 day, or 1 week.

2. Add the new database user



MongoDB Atlas Setup - View Connection String Syntax

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

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN > PROJECT 0

Clusters

 Find a database deployment...

1. View Connection String Syntax

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

REGION

CLUSTER TIER

TYPE

BACKUPS

LINKED APP SERVICES

ATLAS SQL

ATLAS SEARCH

7.0.15

AWS / N. Virginia (us-east-1)

M0 Sandbox (General)

Replica Set - 3 nodes

Inactive

None Linked

Connect

Create Index

+ Add Tag

New On Atlas 8



Connect to Cluster0



Set up connection security



Choose a connection method



Connect



Connect to your application



Drivers

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



2. Click That



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.dtvb**
 - d.mongodb.net/?retryWrites=true&w=majority
 - &appName=Cluster0

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

Connect to Cluster0



Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

Driver Version

Node.js

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

Use this connection string in your application

View full code sample

```
mongodb+srv://<db_username>:<db_password>@cluster0.dtvb.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0
```

Replace `<db_password>` with the password for the `<db_username>` database user. Ensure any option params are [URL encoded](#).

RESOURCES

[Get started with the Node.js Driver](#)

[Access your Database Users](#)

[Node.js Starter Sample App](#)

[Troubleshoot Connections](#)

Go Back

Done



MP3 Express.js Setup

Environment Variables

```
 1. Paste in the mongo connection string  
1 # Uncomment .env from .gitignore to prevent it from being committed to version control  
2 TOKEN=secret-starter-mern  
3 MONGODB_URI=<your_mongodb_uri mongodb+srv://username:password@your-cluster.mongodb.net/database>
```

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)
1 // Get the packages we need
2 var express = require('express'),
3     router = express.Router(),
4     mongoose = require('mongoose'),
5     bodyParser = require('body-parser');
6
7 // Read .env file
8 require('dotenv').config();
9
10 // Create our Express application
11 var app = express();
12
13 // Use environment defined port or 3000
14 var port = process.env.PORT || 3000;
15
16 // Connect to a MongoDB --> Uncomment this once you have a connection string!!
17 //mongoose.connect(process.env.MONGODB_URI, { useNewUrlParser: true });
18
19 // Allow CORS so that backend and frontend could be put on different servers
20 var allowCrossDomain = function (req, res, next) {
21     res.header("Access-Control-Allow-Origin", "*");
22     res.header("Access-Control-Allow-Headers", "X-Requested-With, X-HTTP-Method-Override, Content-Type, Accept");
23     res.header("Access-Control-Allow-Methods", "POST, GET, PUT, DELETE, OPTIONS");
24     next();
25 };
26 app.use(allowCrossDomain);
27
28 // Use the body-parser package in our application
29 app.use(bodyParser.urlencoded({
30     extended: true
31 }));
32 app.use(bodyParser.json());
33
34 // Use routes as a module (see index.js)
35 require('./routes')(app, router);
36
37 // Start the server
38 app.listen(port);
39 console.log('Server running on port ' + port);
40 |
```

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
 - o Preferably using **GitHub**

Render

Create an account

GitHub	GitLab
Bitbucket	Google

or

Email

Password

By signing up you agree to our [terms of service](#) and [privacy policy](#).

Create Account

Already have an account? [Sign in](#)

This site is protected by hCaptcha. Its [Privacy Policy](#) and [Terms of Service](#) apply.

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.**

Overview

Projects

The screenshot shows the Render interface. On the left, there's a large callout text: "Get organized with Projects" with a subtext "An easier way to organize your resources and collaborate with team members." Below it is a button "+ Create your first project". To the right is a sidebar titled "Overview" with a purple "Invite your team" button and a purple "+ Add new ^" button. A dropdown menu is open, listing several project types: "Static Site", "Web Service" (which is highlighted with a red border), "Private Service", "Background Worker", "Cron Job", "Postgres", "Key Value", "Project", and "Blueprint". An orange arrow points from the third item in the list ("Web Service") to the corresponding item in the dropdown menu.

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.

New Web Service

Source Code



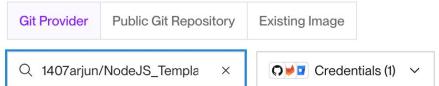
Connect Git provider

Connect your Git provider to deploy from your existing repositories.



New Web Service

Source Code



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)

New Web Service

Source Code

1407arjun / NodeJS_Template • Sep 27, 2021 Edit

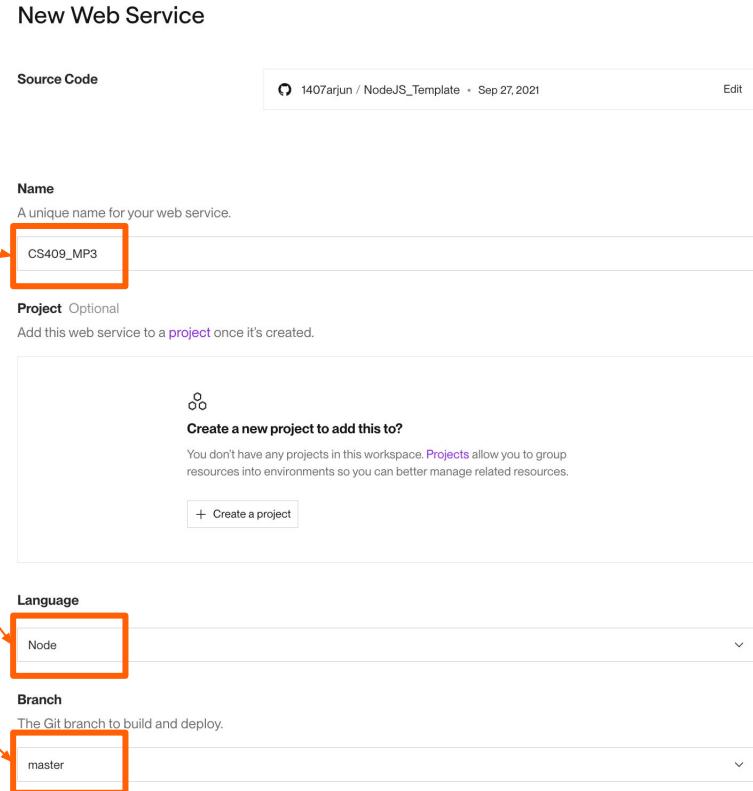
Name
A unique name for your web service.

Project Optional
Add this web service to a [project](#) once it's created.

Create a new project to add this to?
You don't have any projects in this workspace. [Projects](#) allow you to group resources into environments so you can better manage related resources.
+ Create a project

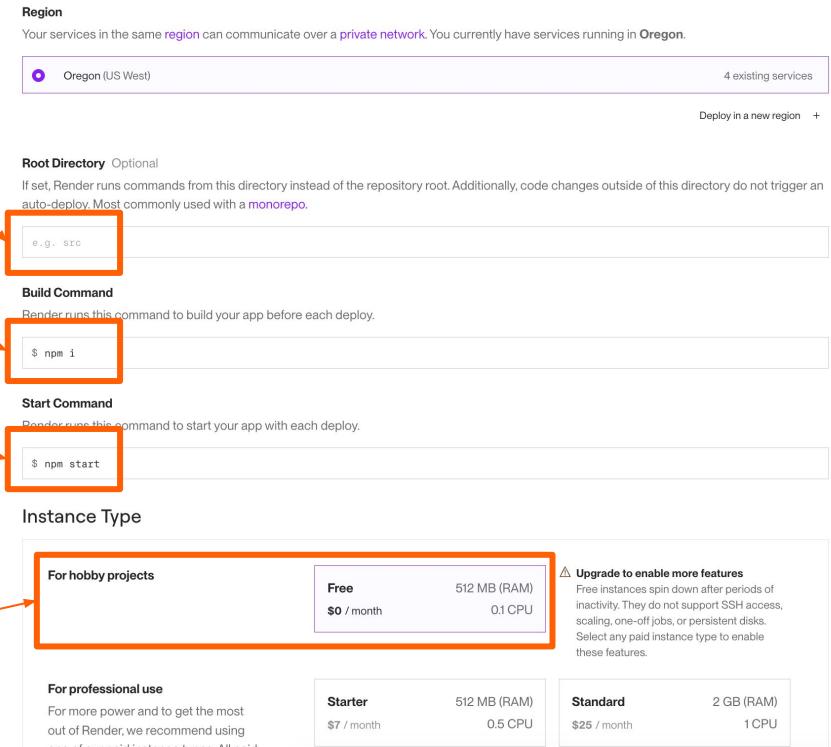
Language

Branch
The Git branch to build and deploy.

A screenshot of the Heroku 'New Web Service' setup page. The 'Name' field contains 'CS409_MP3'. The 'Language' dropdown is set to 'Node'. The 'Branch' dropdown is set to 'master'. A callout arrow from the first bullet point in the list points to the 'Name' field. Another callout arrow from the third bullet point points to the 'Language' and 'Branch' sections.

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**”.
 - o “**yarn**” if using Yarn as the package manager.
3. Change the start command to “**npm start**”.
 - o Should be present in your package.json.
 - o “**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.

The screenshot shows a deployment interface with three stacked environment variable input fields, each with a red arrow pointing downwards to the next field. The first field has 'NAME_OF_VARIABLE' and 'value' input boxes. The second field has 'MONGODB_URI' and a placeholder ' <paste connection string from Altas here>'. The third field has 'MONGODB_URI' and a redacted value. Below these is an 'Advanced' section with a 'Deploy Web Service' button, which is also highlighted with a red box and an arrow pointing to it.

Environment Variables
Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more.](#)

NAME_OF_VARIABLE value [Generate](#) [Delete](#)

+ Add Environment Variable [Add from.env](#)

Environment Variables
Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more.](#)

MONGODB_URI <paste connection string from Altas here> [Delete](#)

+ Add Environment Variable [Add from.env](#)

Environment Variables
Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more.](#)

MONGODB_URI [Delete](#)

+ Add Environment Variable [Add from.env](#)

> Advanced

Deploy Web Service

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.

WEB SERVICE
CS409_MP3

Node Free Upgrade your instance →

Service ID: srv-d3lmaepr0fn873e45nc0 🔍

1407arjun / NodeJS_Template master

<https://cs409-mp3.onrender.com> 🔍

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.
[Upgrade now](#)

October 12, 2025 at 3:22 AM
aa02d86 Docs: Add instructions [Rollback](#)

All logs [Live tail](#) CDT ⌂ ...

```
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ added 290 packages, and audited 291 packages in 3s
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ 53 packages are looking for funding
Oct 12 03:22:30 AM ⓘ run 'npm fund' for details
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ 17 vulnerabilities (4 low, 2 moderate, 0 high, 2 critical)
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ To address all issues, run:
Oct 12 03:22:30 AM ⓘ npm audit fix
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ Run 'npm audit' for details.
Oct 12 03:22:31 AM ⓘ =>> Uploading build...
Oct 12 03:22:36 AM ⓘ =>> Uploaded in 3.4s. Compression took 1.2s
Oct 12 03:22:36 AM ⓘ =>> Build successful 🎉
Oct 12 03:22:37 AM ⓘ =>> Deploying...
Oct 12 03:22:50 AM ⓘ =>> Running 'npm start'
```

Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#)

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.

WEB SERVICE
CS409_MP3

Node Free Upgrade your instance →

Service ID: srv-d3lmaepr0fn873e45nc0 ⓘ

1407arjun / NodeJS_Template master ↗ master

<https://cs409-mp3.onrender.com> ⓘ

Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.
[Upgrade now](#)

October 12, 2025 at 3:22 AM
aa02d86 Docs: Add instructions

Rollback

All logs Search Live tail CDT ...

```
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ added 290 packages, and audited 291 packages in 3s
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ 53 packages are looking for funding
Oct 12 03:22:30 AM ⓘ run 'npm fund' for details
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ 17 vulnerabilities (4 low, 2 moderate, 0 high, 2 critical)
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ To address all issues, run:
Oct 12 03:22:30 AM ⓘ npm audit fix
Oct 12 03:22:30 AM ⓘ
Oct 12 03:22:30 AM ⓘ Run 'npm audit' for details.
Oct 12 03:22:31 AM ⓘ =>> Uploading build...
Oct 12 03:22:36 AM ⓘ =>> Uploaded in 3.4s. Compression took 1.2s
Oct 12 03:22:36 AM ⓘ =>> Build successful 🎉
Oct 12 03:22:37 AM ⓘ =>> Deploying...
Oct 12 03:22:50 AM ⓘ =>> Running 'npm start'
```

Need better ways to work with logs? Try the [Render CLI](#), [Render MCP Server](#), or set up a [log stream integration](#)





MP3 Tips & Tricks



Multiple Files For Each Endpoint

The screenshot shows a VS Code interface with the following details:

- Explorer View:** Shows a tree structure of files and folders:
 - MP3**: config, database_scripts, models, node_modules, routes (selected)
 - routes**: home.js (selected), index.js (highlighted), users.js
 - Others: .gitignore, FAQ.md, LICENSE, package-lock.json, package.json, README.md, server.js
- Editor View:** The **index.js** file in the **routes** folder is open. The code is as follows:

```
/*
 * Connect all of your endpoints together here.
 */
module.exports = function (app, router) {
  ...
  app.use('/api', require('./home.js')(router));
  app.use('/api/users', require('./users.js')(router));
};
```



Endpoint Multiple Routers

```
JS users.js 8, U X
routes > JS users.js > ⚡ <unknown> > ⚡ exports
1   module.exports = function (router) {
2
3     const usersRoute = router.route("/users");
4     const usersIdRoute = router.route("/users/:id");
5
6     usersRoute.post(...);
7     usersRoute.get(...);
8     usersRoute.put(...);
9     usersRoute.delete(...);
10
11    usersIdRoute.post(...);
12    usersIdRoute.get(...);
13    usersIdRoute.put(...);
14    usersIdRoute.delete(...);
15
16    return router;
17
18  }
```



Mongoose Schema Validation

```
JS user.js M X
models > JS user.js > ...
1 // Load required packages
2 var mongoose = require('mongoose');
3
4 // Define our user schema (https://mongoosejs.com/docs/api/schema.html)
5 var UserSchema = new mongoose.Schema({
6   name: { type: String, required: [true, "name is required"], unique: true }
7 });
8
9 // Export the Mongoose model
10 module.exports = mongoose.model('User', UserSchema);
11
```



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 | });


```



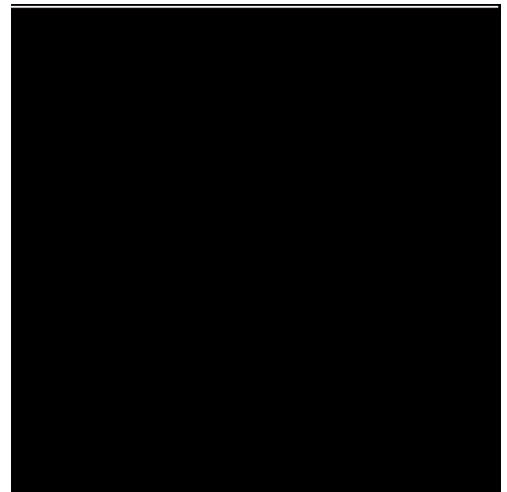
Mongoose Query Builder & Reading URL Params

```
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  })
```



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/>