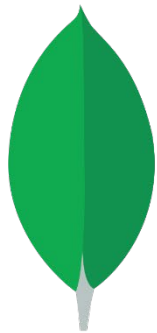


# Setting Up MongoDB: A Guide

Authored by Zander Preston



# mongoDB®

## Introduction

As your start to learn the inner machinations of the wonderful, yet confusing world of web development, you quickly learn that your options for storage are quite limited. A browser can only store so much data in cookies, and local storage can be heavy on a personal computer. Database storage is a must-have for most web server operations.

MongoDB is a cloud-based database host that simplifies the difficulties of setting up a local database server. The free tier offered by the service gives you 512 MB of storage for easy deployment to see if the software is a right fit for your application.

In this guide, we will underline the process of creating and setting up an account on MongoDB, as well as implementing the database into a basic NodeJS application.

## Requirements

- A basic knowledge of the JavaScript programming language
- Installation and basic knowledge of NodeJS
- An email available for creating an account on MongoDB

## Table of Contents

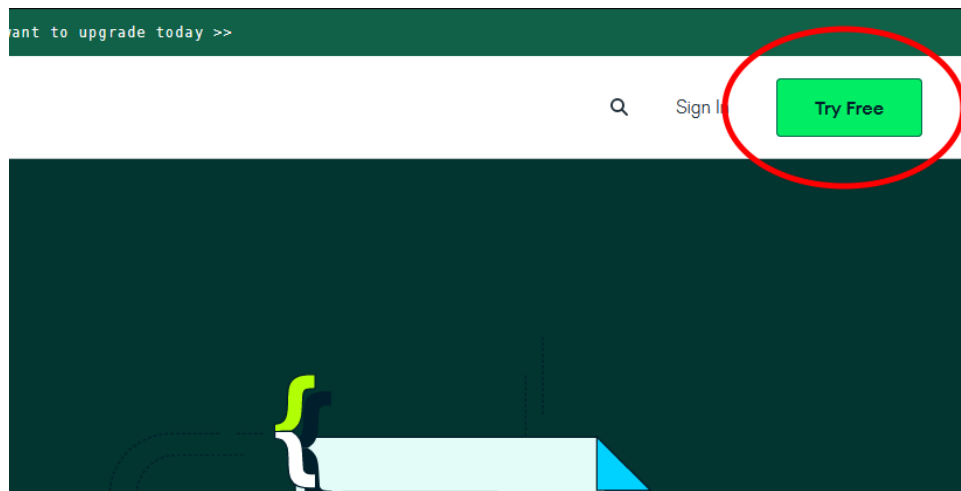
- [Setting Up a MongoDB Account](#)
  - ✓ [Creating the Account](#)
  - ✓ [Setting Up a Cluster](#)
- [Installing MongoDB and Mongoose](#)
  - ✓ [Installing the Packages Using npm](#)
- [Implementing MongoDB into an Application](#)
  - ✓ [Connecting to the Cluster](#)
- [Conclusion](#)
- [Glossary](#)

# Setting Up a MongoDB Account

To use MongoDB as a database for our web application, we must first create credentials for our database. As MongoDB is a cloud-based storage, we will be connecting to our database over the Internet, and with an account, we can modify the security of connecting to our database.

## 1. Creating the Account

Head over to <https://www.mongodb.com> in your web browser of choice and click the “Try Free” button in the top right of the homepage.

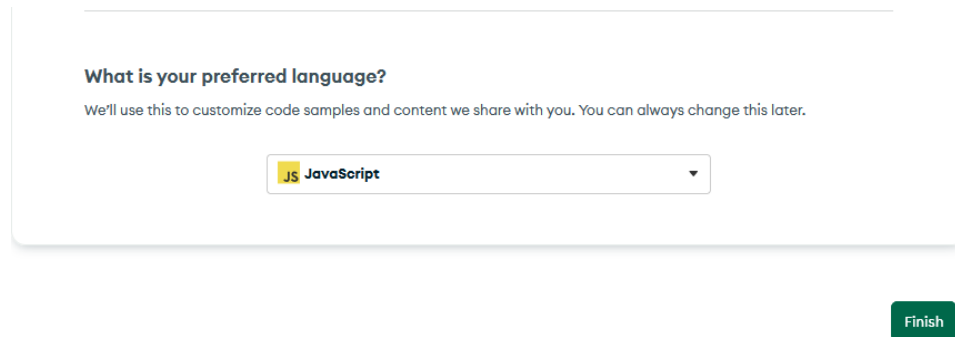


After clicking this button, you will be redirected to a form to fill out for your new account. Fill out this form and click the “Create your Atlas account” button.

A screenshot of the MongoDB Atlas Sign Up form. The form is titled "Sign Up" and includes the subtitle "See what Atlas is capable of for free". On the left side, there is a green sidebar with the heading "MongoDB Atlas" and three bullet points: "Work with your data as code", "Focus on building, not managing", and "Simplify your data dependencies". The main form area contains five input fields: "First Name\*", "Last Name\*", "Company", "Email\*", and "Password\*" (with an eye icon for toggling visibility). Below these fields is a checkbox labeled "I agree to the Terms of Service and Privacy Policy." and a green button at the bottom labeled "Create your Atlas account".

Your email address will be sent a verification email from MongoDB. Navigate to your email and verify your new account.

Afterwards, you will be sent to another form page. This form identifies why you are interested in using MongoDB. You can fill out the elements as you wish, but for the sake of following this tutorial, be sure to set your preferred language to “JavaScript”. Afterwards, click “Finish”.



**What is your preferred language?**

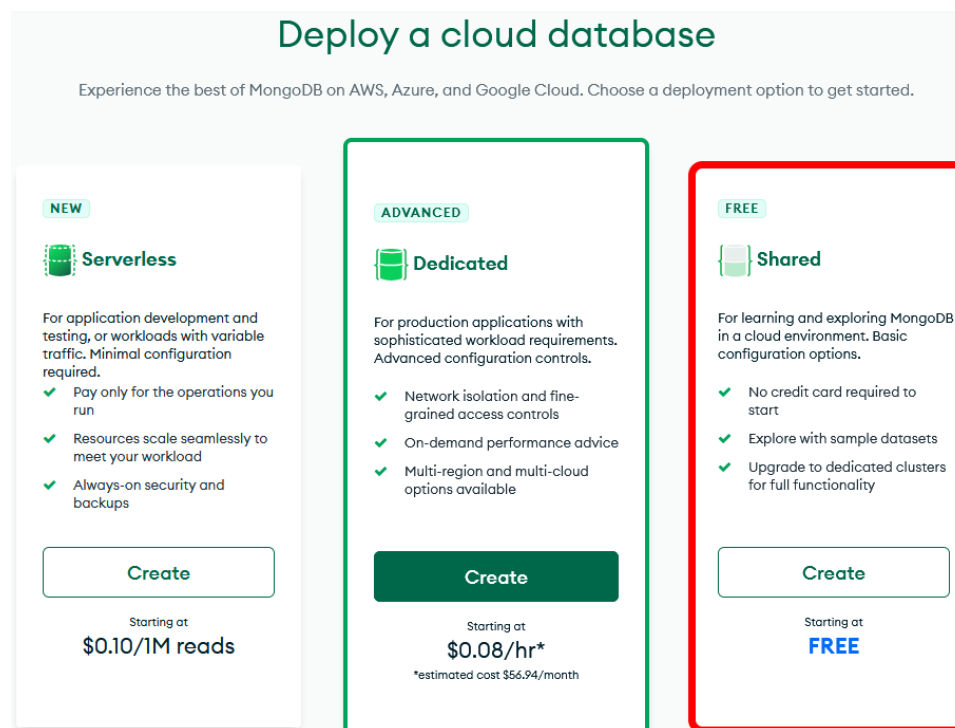
We'll use this to customize code samples and content we share with you. You can always change this later.

JS JavaScript

Finish

## 2. Setting Up a Cluster

After creating your account, you should be shown three options for deploying your database. In this tutorial, we will be selecting the Shared cluster option.



### Deploy a cloud database

Experience the best of MongoDB on AWS, Azure, and Google Cloud. Choose a deployment option to get started.

NEW	ADVANCED	FREE
<b>Serverless</b>	<b>Dedicated</b>	<b>Shared</b>
For application development and testing, or workloads with variable traffic. Minimal configuration required.	For production applications with sophisticated workload requirements. Advanced configuration controls.	For learning and exploring MongoDB in a cloud environment. Basic configuration options.
<ul style="list-style-type: none"><li>✓ Pay only for the operations you run</li><li>✓ Resources scale seamlessly to meet your workload</li><li>✓ Always-on security and backups</li></ul>	<ul style="list-style-type: none"><li>✓ Network isolation and fine-grained access controls</li><li>✓ On-demand performance advice</li><li>✓ Multi-region and multi-cloud options available</li></ul>	<ul style="list-style-type: none"><li>✓ No credit card required to start</li><li>✓ Explore with sample datasets</li><li>✓ Upgrade to dedicated clusters for full functionality</li></ul>
Create	Create	Create
Starting at \$0.10/1M reads	Starting at \$0.08/hr* <small>*estimated cost \$56.94/month</small>	Starting at <b>FREE</b>

After clicking “Create”, you will be redirected to the creation page, where the recommended settings are preselected for you. You can change the name of the cluster at the bottom and then click “Create Cluster”. *Note: you cannot change the name of the cluster after creating it.*

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

You need to create a database user to connect to the database. The username and password can be set to anything you like, and you can autogenerate a password for more security. Be sure to create this user, as we will be using it to connect to the database later.

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

Enter username

Password

Enter password

Autogenerate Secure Password

Copy

Create User

One more step before we can access the cluster. We need to add IP addresses to connect to the database. Add your current IP address. You can add “0.0.0.0” to access the database from anywhere as well.

#### 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		
Enter IP Address	Enter description	Add Entry	Add My Current IP Address

---

IP Access List	Description	
0.0.0.0/0		REMOVE
	My IP Address	REMOVE

Afterwards, click “Access and close”. You have successfully created a cloud database cluster!

# Installing MongoDB and Mongoose

Now that we have a cluster set up and created, we need to download and install MongoDB. With Node, this is a quick and painless step, as Node is packaged with NPM, or the Node Package Manager. In addition, we will install Mongoose, a Node package to make connecting to our database and accessing its data convenient.

## 1. Installing the Packages Using npm

Using your operating system's terminal, locate the folder containing your web application, or create a new folder. In this tutorial, we will be using Windows Powershell.

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

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

PS D:\Windows Files\Documents\Programs\testApplication> |
```

In the terminal, type the following command to install the MongoDB package for Node:

```
npm install mongodb
```

If you created a new folder for your web application, a successful installation would create a folder and two package files. Afterwards, enter the following to install the Mongoose package for Node:

```
npm install mongoose
```

```
PS D:\Windows Files\Documents\Programs\testApplication> npm install mongoose

added 9 packages, and audited 30 packages in 2s

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

found 0 vulnerabilities
```

To ensure the packages downloaded correctly, open the package.json file using a text editor. If you see “mongodb” and “mongoose” under dependencies, you have successfully installed both packages.

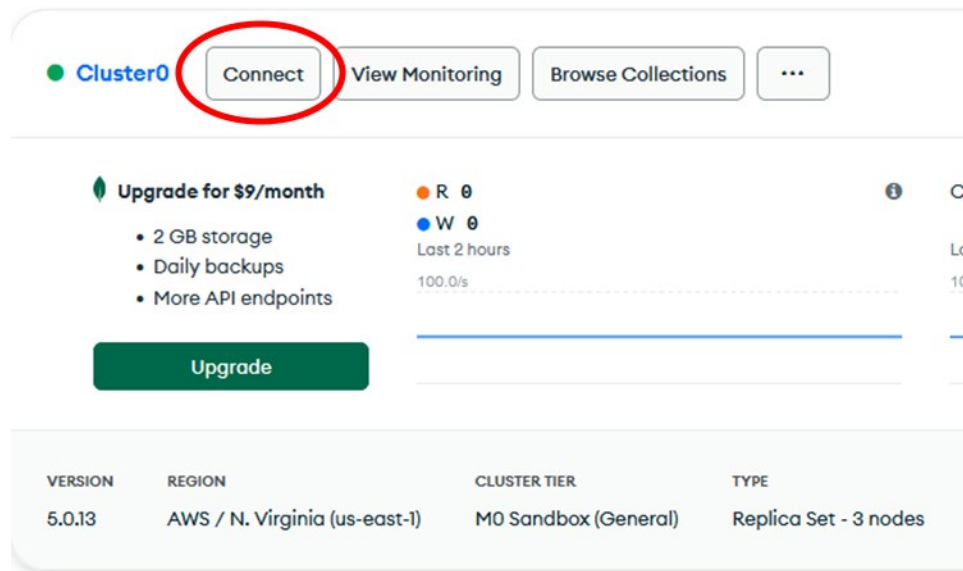
```
1  {
2    "dependencies": {
3      "mongodb": "^4.10.0",
4      "mongoose": "^6.6.5"
5    }
6  }
```

# Implementing MongoDB into an Application

Now that we have MongoDB and Mongoose installed, connecting to and utilizing our database will be a simple process.

## 1. Connecting to the Cluster

On our Database Deployments homepage, you can a lot of data about connections and operations pertaining to your cluster. There are four buttons the right of the cluster name, and we want to select “Connect”.



From the new window that just popped up, we will select “Connect your application”. This will redirect to this screen, which will display the code to connect to the database. We need to implement this line into our application.

✓ Setup connection security ✓ Choose a connection method Connect

**1 Select your driver and version**

DRIVER

Node.js

VERSION

4.1 or later

**2 Add your connection string into your application code**

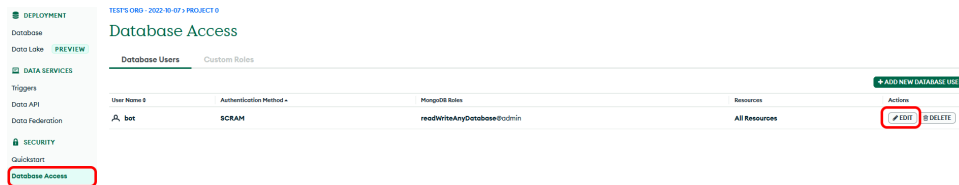
☐ Include full driver code example

```
mongodb+srv://bot:<password>@cluster0.qv5sfvf.mongodb.net/?retryWrites=true&w=majority
```

Replace **<password>** with the password for the **bot** user. Ensure any option params are [URL encoded](#).

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

With the line copied, we need to replace the “<password>” part with the password you set for the database user from earlier. If you do not remember your password, you can click on “Database Access” in the left side menu and edit the user from the page that opens.



An example link could be as follows:

```
mongodb+srv://bot:
093qBcgYHIJx0Hq9@cluster0.qv5sfvf.mongodb.net/?retryWrites=true&w=majority
```

With a completed connection link, we can head to our folder we made and make a new file named “index.js”. In this file, we need to import the mongoose package.



From here, we can connect to the database using the following Mongoose method:

```
await mongoose.connect(link, { useNewUrlParser: true, useUnifiedTopology: true });
```

You need to replace “link” with the link you just retrieved from earlier.

With this connection method, we can check if the connection with a function like the following:

```
const mongoose = require("mongoose")
async function connect() {
  await
  mongoose.connect('mongodb+srv://bot:093qBcgYHIJx0Hq9@cluster0.qv5sfvf.mongodb.net/?re
tryWrites=true&w=majority', { useNewUrlParser: true, useUnifiedTopology: true });

  console.log("Connected to the database!");
  console.log(mongoose.connection.host, mongoose.connection.name)
  await mongoose.connection.close();
  console.log("Disconnected!")
}
connect();
```

The following picture will look like your output when executing the above code:

```
Connected to the database!  
ac-i8ohfxa-shard-00-01.qv5sfvf.mongodb.net test  
Disconnected!  
  
Process finished with exit code 0
```



## Conclusion

By following these instructions, you have learned how to create an account for and connect to a MongoDB cluster. Now that you know how to implement MongoDB into your application, it is now time to use it! Creating, editing, and deleting documents can all be explored [here, in the Mongoose documentation](#).

## Glossary

- Cluster  
A collection of datasets distributed across many servers to achieve better performance in accessing and changing the database.
- Database  
A structured set of data held in a computer, typically one that is accessible and changeable in many ways.
- MongoDB  
A cloud-based document database with immense scalability and flexibility for what any web developer may need.
- Mongoose  
A node package used to make connecting to and interacting with a MongoDB database easier and more convenient.
- NPM  
The software associated with Node to download and install packages curated by other users. Shorthand for Node Package Manager.