

# MongoDB Atlas Setup Tutorial

MongoDB is a powerful NoSQL database that uses documents that closely resemble JSON to store application data. These documents exist inside of collections. If you are familiar with SQL, a relational database, then you will recognize that documents are synonymous with records and collections are synonymous with tables.

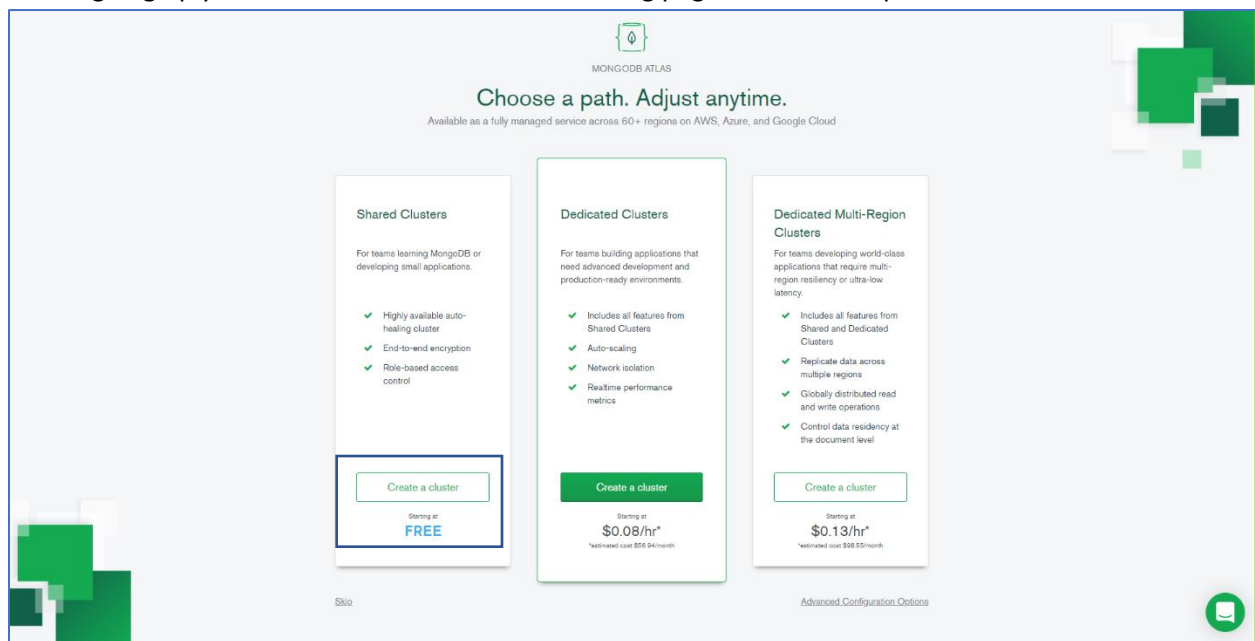
MongoDB can be used as the traditional document database on a server or it can be used as a global cloud documents database with MongoDB Atlas. With cloud computing becoming more advantageous and popular, it makes sense for you to learn how to create clusters and query data from MongoDB Atlas.

## Setting Up MongoDB Atlas Account & Creating Your First Cluster

First, you will need to create a free account with MongoDB Atlas by going here and clicking “Sign Up”:  
<https://account.mongodb.com/account/login>

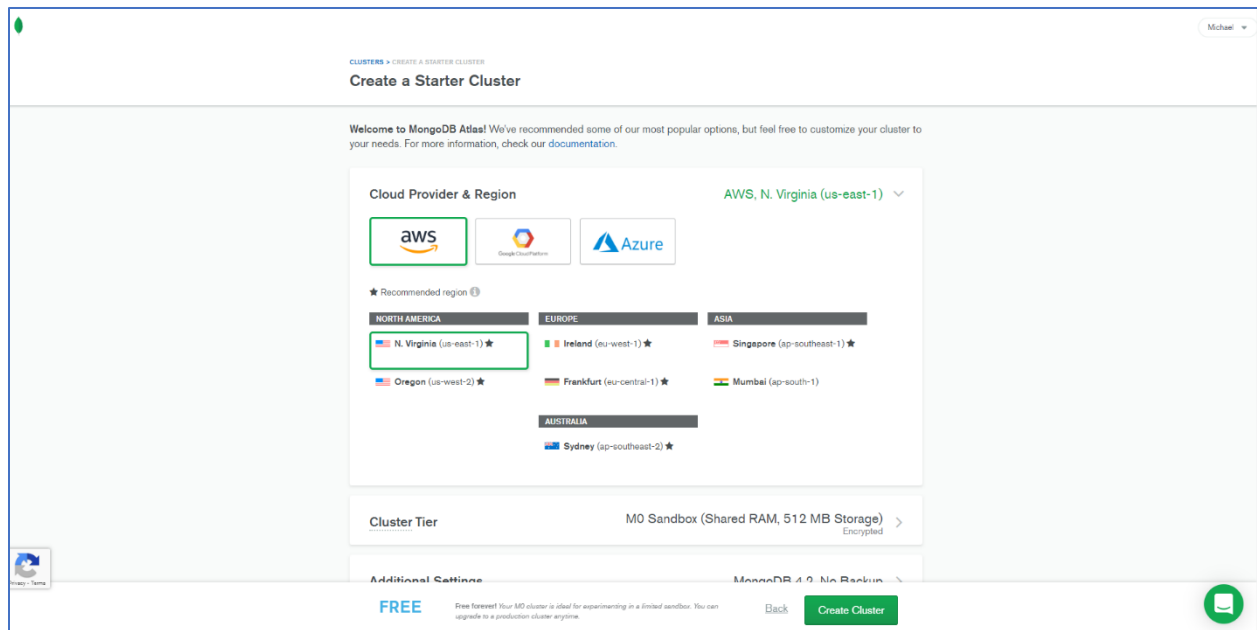
You can either sign up with your Google account or email address.

After signing up you will be redirected to the following page to “choose a path”:



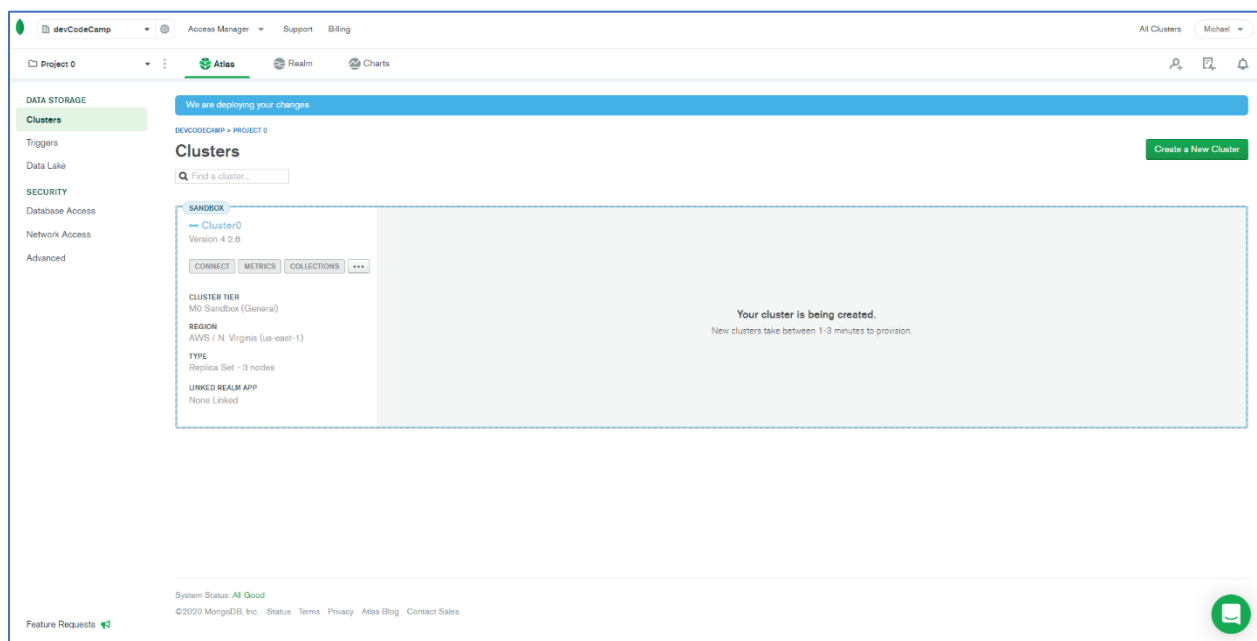
Make sure you click “Create a cluster” start at FREE on the left

From there you will be redirected to the following page where you will choose a cloud provider and region. I recommend AWS and N. Virginia (us-east-1) if you are in the middle to eastern part of the United States. Otherwise, I recommend AWS and Oregon (us-west-2). You also want to make sure the free M0 Sandbox (Shared RAM, 512 MB Storage) is the Cluster Tier selected.

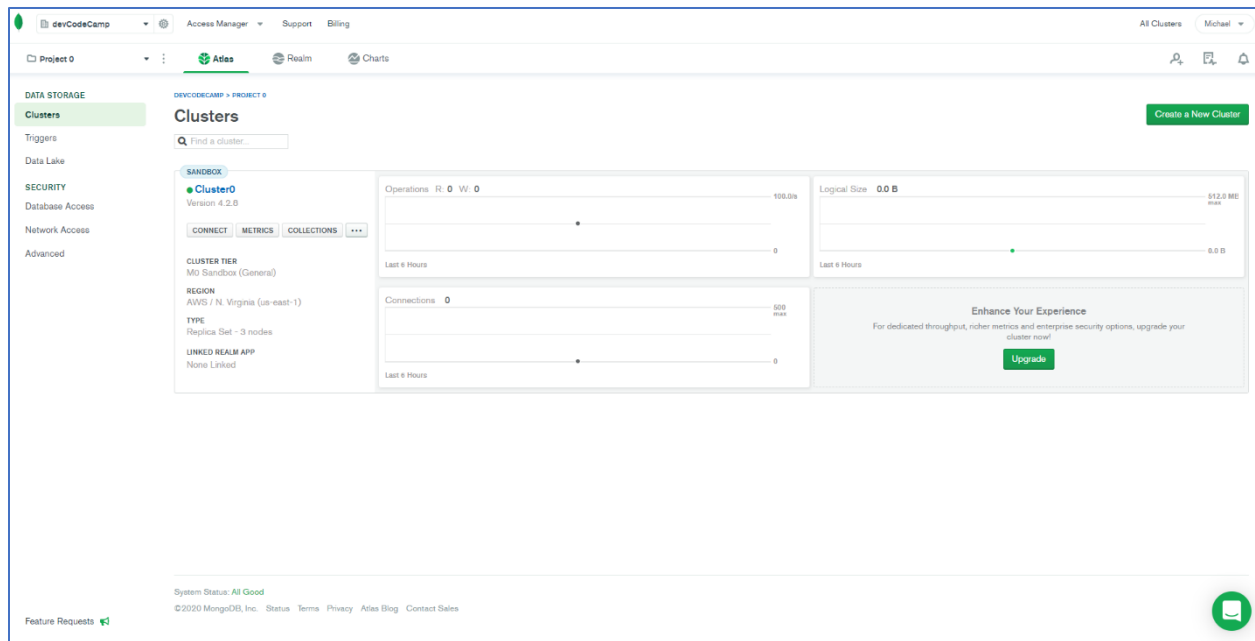


After making the proper selections for you, click “Create Cluster” on the bottom of the page. It will take a few minutes while MongoDB Atlas creates your first cluster.

A **cluster** is a set of nodes comprising a MongoDB deployment. In Atlas, these clusters can be **replica sets** or **sharded**. A replica set cluster is a group of MongoDB servers that maintain the same data set. It is the basis for all production deployments. A sharded cluster is a set of nodes comprising a sharded MongoDB deployment. A **shard** is a partition of data in a database.

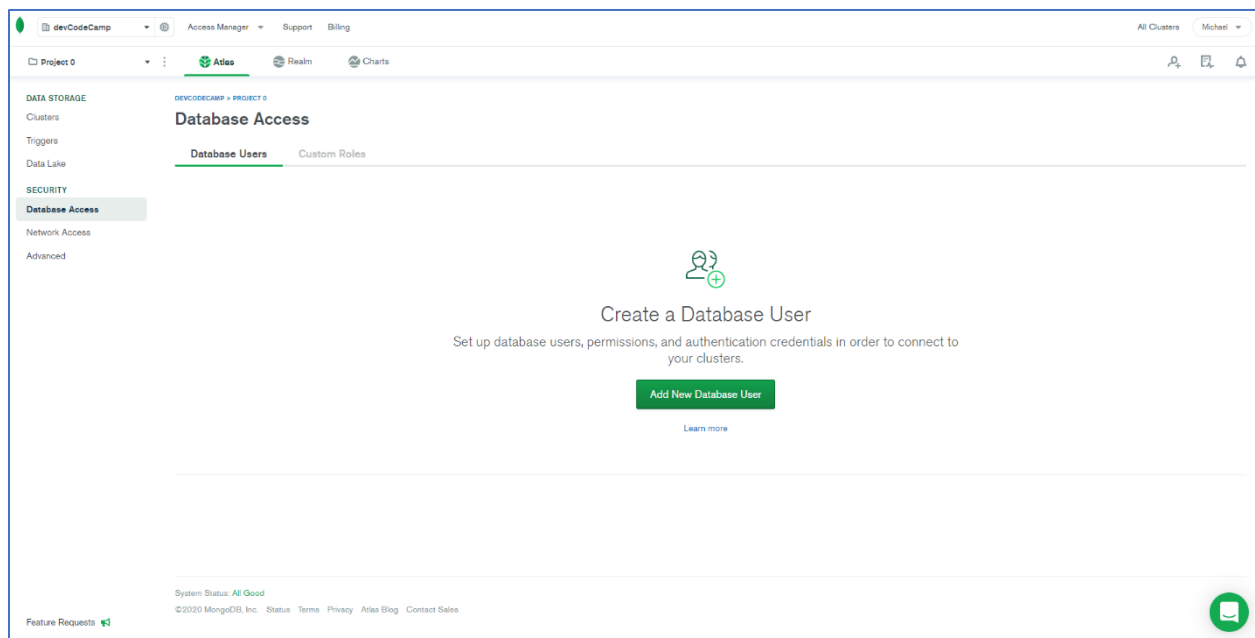


Once the cluster is created, you will see the following screen:



## Adding a New Database User

The side-nav bar will show Data Storage and Security. Clicking “Clusters” will take you to the above screen, which is where all clusters will exist. For now, click on Database Access under Security. This will take you to the following screen where you can create, edit, and delete database users:



Add a new database user by clicking “Add New Database User” in the middle of your screen. You should see the following:

### 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

Password

Certificate  
(M10 and up)

AWS IAM  
(MongoDB 4.4, M10 and up)

MongoDB uses [SCRAM](#) as its default authentication method.

Password Authentication

Database User Privileges

Select a built-in role or privileges for this user.

Temporary User

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

☐ OFF

Make sure “Password” is selected for the Authentication Method. Then fill in a username and password in the Password Authentication section. In the Database User Privileges section, there is a dropdown that will let you choose a built-in role. For now, make sure “Read and write to any database” is selected. If everything is correct, click “Add User” in the bottom right. You will then be redirected to the following screen:

Michael's Org - 2019...

Access Manager

Support

Billing

MERN

Atlas

Realm

Charts

DATA STORAGE

Clusters

Triggers

Data Lake

SECURITY

Database Access

Network Access

Advanced

MICHAEL'S ORG - 2019-10-19 > MERN

Database Access

Database Users

Custom Roles

+ ADD NEW DATABASE USER

User Name	Authentication Method	MongoDB Role	Actions
mike123	SCRAM	readWriteAnyDatabase@admin	<input type="button" value="EDIT"/> <input type="button" value="DELETE"/>

System Status: All Good

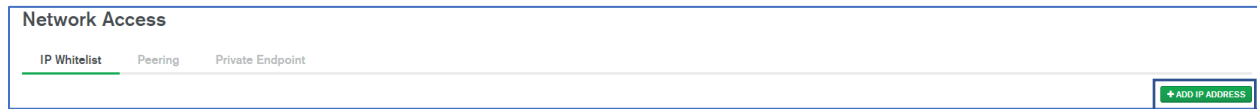
©2020 MongoDB, Inc. [Status](#) [Terms](#) [Privacy](#) [Atlas Blog](#) [Contact Sales](#)

Feature Requests

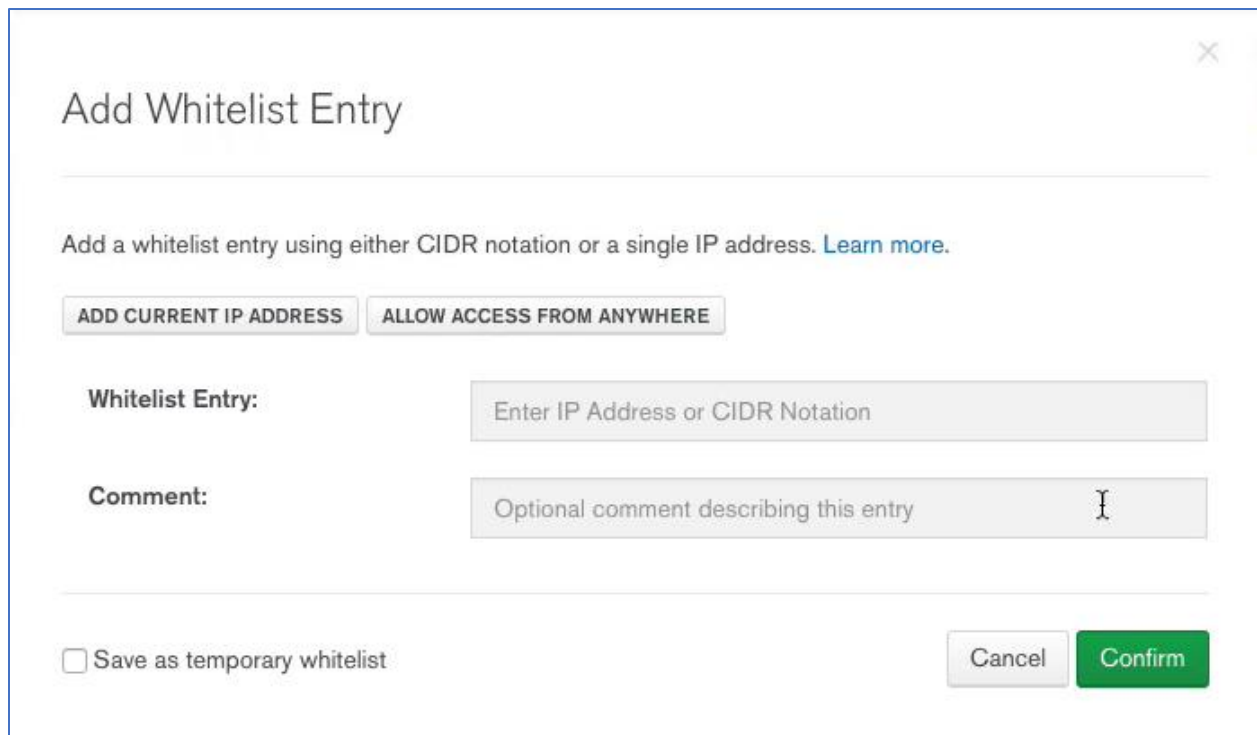
Whitelist IP Address

It is possible to restrict access to a MongoDB database by whitelisting an IP address. This makes everything more secure, which is important given that MongoDB Atlas is a cloud database and not locally ran on your machine.

Click on Network Access under Security in the side-nav bar. In the following image while the IP Whitelist tab is selected, click “Add IP Address”.



A modal will pop up for you to add an IP whitelist entry.

The image shows a screenshot of the 'Add Whitelist Entry' modal. The modal has a title 'Add Whitelist Entry' and a close button (X) in the top right corner. Below the title, there is a line of text: 'Add a whitelist entry using either CIDR notation or a single IP address. [Learn more.](#)'. There are two buttons: 'ADD CURRENT IP ADDRESS' and 'ALLOW ACCESS FROM ANYWHERE'. Below these buttons, there are two input fields. The first is labeled 'Whitelist Entry:' and has a placeholder text 'Enter IP Address or CIDR Notation'. The second is labeled 'Comment:' and has a placeholder text 'Optional comment describing this entry'. At the bottom left, there is a checkbox labeled 'Save as temporary whitelist'. At the bottom right, there are two buttons: 'Cancel' and 'Confirm'.

You have two options when it comes to whitelisting an IP:

1. Add Current IP Address
2. Allow Access From Anywhere

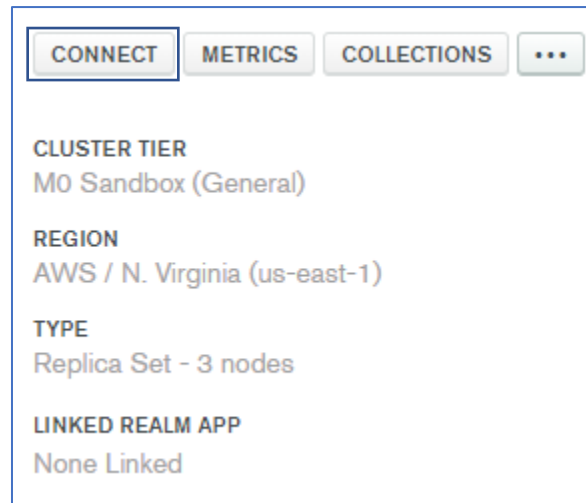
Add Current IP Address will get your IP address automatically and whitelists it. That means only requests from that IP will have access to the database. Despite being secure, IP addresses change frequently, which leads to a need to update the whitelist IP address frequently. So, if you choose this method and cannot access your database anymore, you should look here as the likely reason why that is.

Allow Access From Anywhere will provide an IP address of 0.0.0.0/0, which means access to the database is open to any request from any IP address.

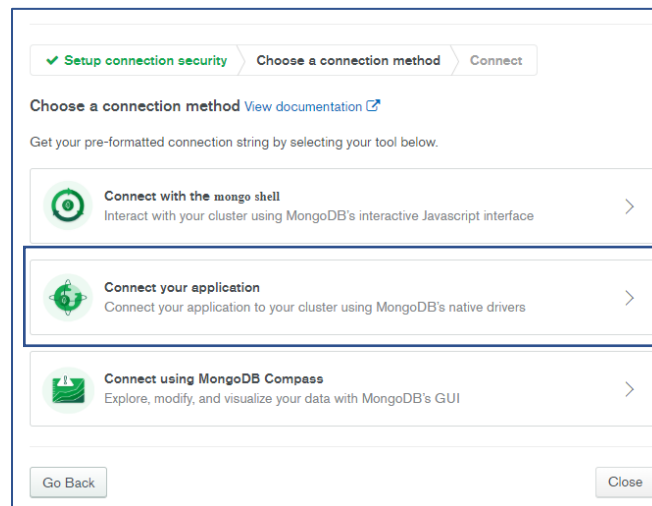
For any database that goes into production, it is highly recommended to use the Add Current IP Address option. It is important to note that once you deploy the database to the server that it is the server's IP address that gets whitelisted.

## Connecting Database to Node.js & Express.js Web Server

Before you can connect your database to your Node.js and Express.js web server you will need to first grab the database connection string. Click "Clusters" on the side-nav bar and select the cluster you want. Then click the "Connect" button.



Presented to you will be three connection methods. The one you will need to select is "Connect your application".



The next step is to copy the connection string that is presented to you in the section titled "Add your connection string into your application code".

✓ Setup connection security ✓ Choose a connection method Connect

1 Select your driver and version

DRIVER: Node.js VERSION: 3.6 or later

2 Add your connection string into your application code

☐ Include full driver code example

mongodb+srv://mike123:<password>@devconnector-zij44.mongodb.net/<dbname>

Copy

Replace <password> with the password for the mike123 user. Replace <dbname> with the name of the database that connections will use by default. Ensure any option params are URL encoded.

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

Go Back Close

After the connection string is copied you will need to locate the config folder inside your application project folder. Inside the **config folder** will be a file called **default.json**. This file will contain a JSON object with one of the keys being **"mongoURI"**. The value to this key is where you will paste your connection string. You will need to put in your password that you created when setting up the database user where it says <password> in the connection string.

```
{
  "mongoURI": "mongodb+srv://mike123:<password>@devconnector-zij44.mongodb.net/test?retryWrites=true&w=majority"
}
```

## Modify .gitignore File

You will need to initialize a Node.js .gitignore when creating your GitHub repository. Anything inside the .gitignore will be ignored by Git. Since GitHub public repos are open-source and the default.json file will contain sensitive information, including your username and password for your database, you will need to include the default.json in your repo's .gitignore file immediately.

```
.gitignore X {} default.json
SocialDev > .gitignore
1 node_modules/
2
3 ./config/default.json
4
```

Make sure to save the .gitignore file after you include the default.json. Then you are clear to git add, git commit, git push your project files to your remote repo. The default.json will not appear on your remote repo on GitHub because it was ignored by Git.

## Conclusion

Congrats! You just set up your MongoDB Atlas account, created your first cloud database, added a new database user, whitelisted an IP address for security reasons, connected your created database to a Node.js and Express.js web server, and added your database connection string to your repo's .gitignore file.