

Setup

Part One install git, nodejs, and npm

Go to the website:

<https://git-scm.com/downloads/>

Choose the appropriate installer for your system. See the git tutorial for more help.

Go to the website:

<https://nodejs.org/en/download/>

And choose the installer for your system, if you use linux you can probably just use your system's package manager, for example in ubuntu to install nodejs and npm

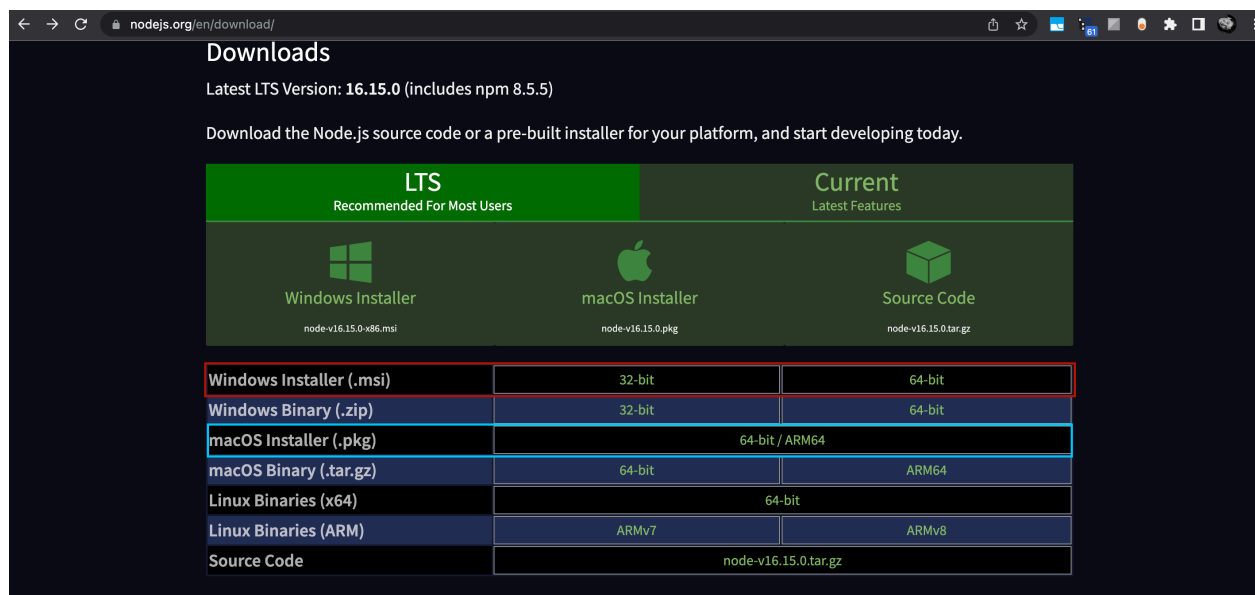
into your console type:

```
sudo apt upgrade && apt update
```

```
sudo apt install nodejs
```

```
sudo apt install npm
```

If you are using a Mac OS, or Windows device than downloading and running the installer should also get you up and running.



Red showing the windows installer, blue showing the Mac osx installer

Part Two Setting up mongodb Atlas

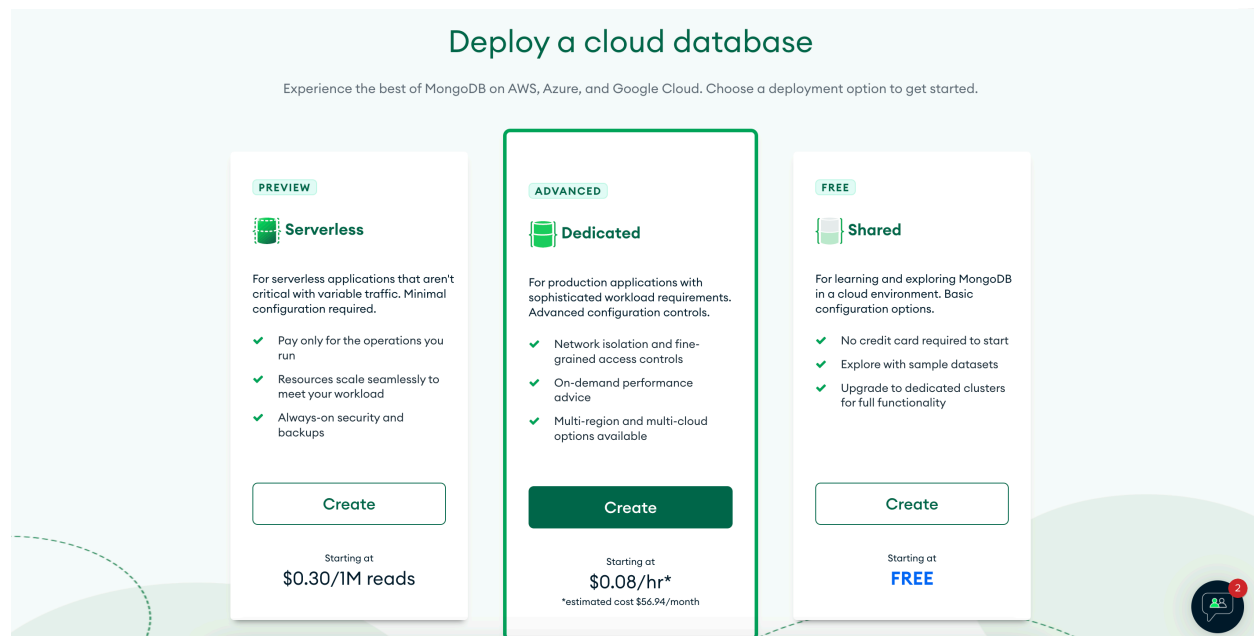
First Navigate to the page mongodb.com

Fill in the information in the form to setup a free cluster of databases, or simply sign up with your Google account

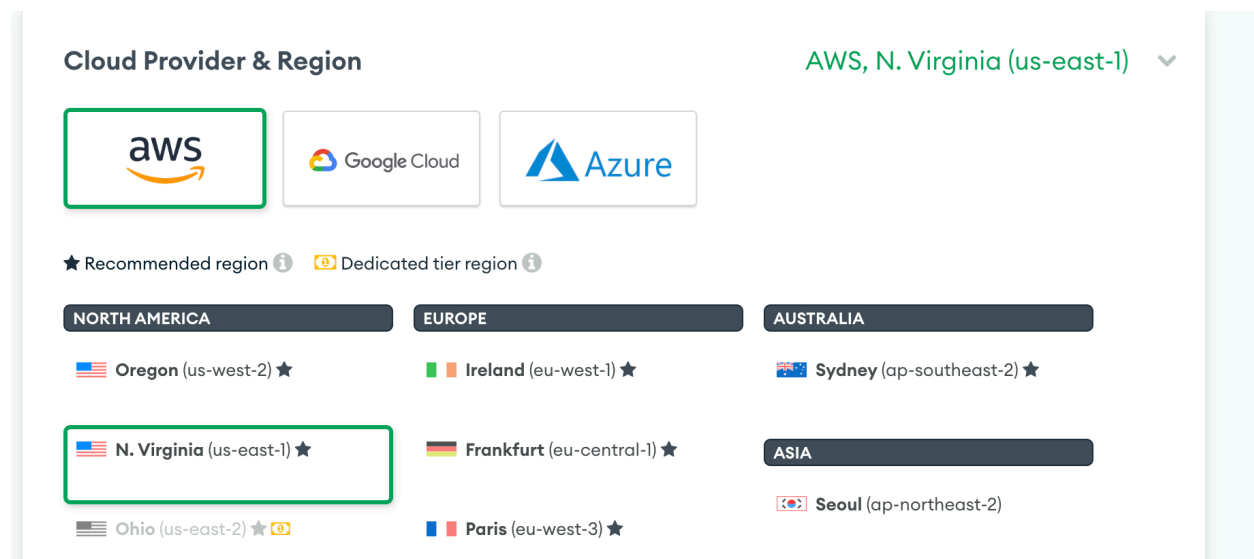
One you sign up you should see the following screen press "Build a Database"

Once you click the button pictured above you will see the following screen:

Press the rightmost "Create" button to create a free shared database.



This will load the following page where you can choose where your mongo database will be hosted:



Please ensure you've chosen the same options as shown above
AWS as your cloud provider, and N. Virginia as your region.

Cluster Tier

M0 Sandbox (Shared RAM, 512 MB Storage)
Encrypted

Additional Settings

MongoDB 5.0, No Backup

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.

Above we can choose a name for our cluster. By default it is cluster0.

A cluster is a collection of databases, so one cluster can have several databases, and a database itself is a collection of what mongodb calls “Collections” which are collections of arbitrary objects.

These collections of objects are kind of like tables of data, except they can be more complex than a 2d table representing any n dimensional table with its tree like structure.

When transmitting these database object records over the network we use typically will use json which is a text format designed for encoding arbitrary data, and it's structure.

for instance if we look at the C++ class Person defined below:

```
class Person {  
    public:  
        std::string first_name, last_name;  
        std::vector<std::string> email_addresses;  
};
```

An instance of a person's data represented as json might look like:

```
{
  "first_name": "John",
  "last_name": "Smith",
  "email_addresses": [
    "example1@gmail.com",
    "example2@gmail.com",
    "example3@gmail.com",
    "example4@gmail.com"
  ]
}
```

First, we should notice the curly braces surrounding the text this denotes that everything between the curly braces will be the definition of an object record.

Inside the curly braces the first thing we see is:

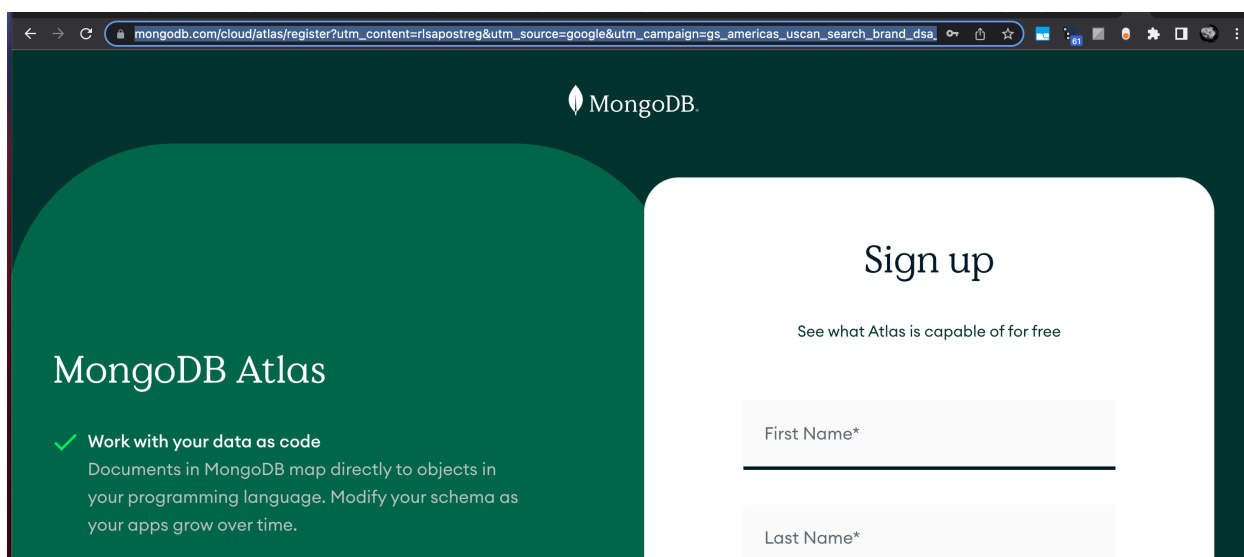
```
"first_name": "John"
```

This is the definition for the field `first_name`, and analogous to an instance of a Person's `first_name` public member.

In json since we cannot encode functions or methods all of the members are public, like a C struct. This is why in the C++ code all the members of person are publicly accessible to simulate this restriction of JSON.

Notice how the field is encoded, first the field name `first_name` wrapped in quotation marks, in json you will always first see a field name, also referred to as a key, and then a colon, and the data associated with that key, or field name. In JSON we never will have missing data because this is a data record we use to send data over the network, and that would only add an extra key value wasting time to send extra useless data.

In this case `first_name`'s data is a string, we know this because it is wrapped in quotes, then followed by a comma to denote another field follows this one, that is the field `last_name`. Note that it is also a string. The final field `email_addresses` is an array of data denoted by the wrapping `[]` (square braces). Inside it is a comma separated list of strings which is analogous to the field `email_addresses` in a person object.



The screenshot shows the MongoDB Atlas sign-up page. The page has a dark green header with the MongoDB logo. Below the header, there's a large green section on the left with the text "MongoDB Atlas" and a checkmark icon next to the text "Work with your data as code". To the right of this, there's a white box with the heading "Sign up" and the subtext "See what Atlas is capable of for free". Below this, there are two input fields: "First Name*" and "Last Name*", each with a red asterisk indicating a required field. The browser's address bar shows the URL "mongodb.com/cloud/atlas/register?utm_content=rlsapostreg&utm_source=google&utm_campaign=gs_americas_uscan_search_brand_dsa".

Next you will create a username, and password that you will use so that your application can connect to the web server.

Atlas

Cluster

Database

1

How would you like to authenticate your connection?


Your first user will have permission to read and write any data in your project.


Username and Password


Certificate

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

Password 

 Autogenerate Secure Password

 Copy

Create User

ioning... 


You will need to remember this to setup the .env file for your application

Next you will add a record that will whitelist all ip addresses so that you can connect your web server from a dynamically changing ip.

2

Where would you like to connect from?


Enable access for any network(s) that need to read and write data to your cluster.



My Local Environment

Use this to add network IP addresses to the IP Access List. This can be modified at any time.

ADVANCED



Cloud Environment

Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.

Add entries to your IP Access List

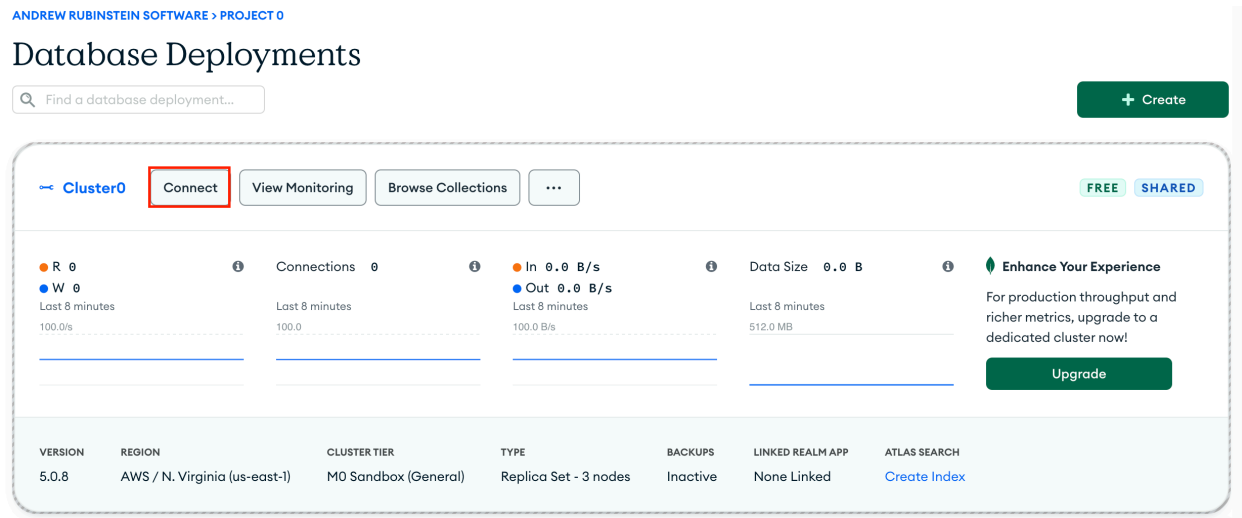
Only an IP address you add to your Access List will be able to connect to your project's clusters.

IP Address	Description		
<input type="text" value="0.0.0.0"/>	<input type="text" value="g connections from all ips"/>	<div>Add Entry</div>	<div>Add My Current IP Address</div>

This is not good practice for security you should have a static ip you buy from your internet service provider, and only allow access from that ip, in a deployed application, but we want this tutorial to be free to use.

Click the “Add Entry” button and now we can begin setting up our mongoose application. Then press the finish and go to next page button on the bottom right once you’ve created your user, and allowed all its to connect.

Once you do you will see the following page:



The following window will then popup, choose “Connect your application” to continue, this will give us a connection string url

Connect to Cluster0


✓ Setup connection security


Choose a connection method


Connect

Choose a connection method [View documentation](#)

Get your pre-formatted connection string by selecting your tool below.

**Connect with the MongoDB Shell**
Interact with your cluster using MongoDB’s interactive Javascript interface

**Connect your application**
Connect your application to your cluster using MongoDB’s native drivers

**Connect using MongoDB Compass**
Explore, modify, and visualize your data with MongoDB’s GUI

Then click the highlighted button in the graphic above to copy this connection string, and make sure to save it somewhere to use in the .env file later

✓ Setup connection security

✓ Choose a connection method

Connect

1 Select your driver and version

DRIVER	VERSION
Node.js	4.1 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://testUser:<password>@cluster0.pqym9.mongodb.net/?
retryWrites=true&w=majority
```

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

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