MEAN Stack Deployment on Linux

MEAN Stack comprises of the following;

- MongoDB (Document database) - Stores and retrieve data.
- Express (Back-end application framework) - Makes requests to Database and return a response
- Angular (Front-end application framework) - Handles Client and Server Requests
- Node.js (JavaScript runtime environment) - Accept requests and display results to end user

I started this project by creating an AWS account and provisioning a free virtual server with Ubuntu Server OS called EC2 (Elastic Compute Cloud) for this project since the oracle Virtual Machine earlier configured will no longer be used going forward.

AWS Management Console

After creating the account which involves using a valid email, entering payment information and address etc, click on AWS management console and sign in as ROOT USER then enter ROOT USER email address. Click next and enter password

Search for Elastic Cloud Compute (EC2) in the “find services” search box and select EC2 virtual services in the cloud.

Select Launch Instance and click of Free tier only under the quick start bar.

Choose an Amazon Machine Image (AMI)

Search for Ubuntu under this step and select Ubuntu server 20.04 LTS

Choose an Instance Type

Select the general purpose free tier t2.micro instance type with 1 CPU and 1GB Memory etc

Configure Instance Details

Leave everything as default

Add Storage

Leave everything as default but make sure that delete on termination is checked

Add Tags

No tags added

Configure Security Group

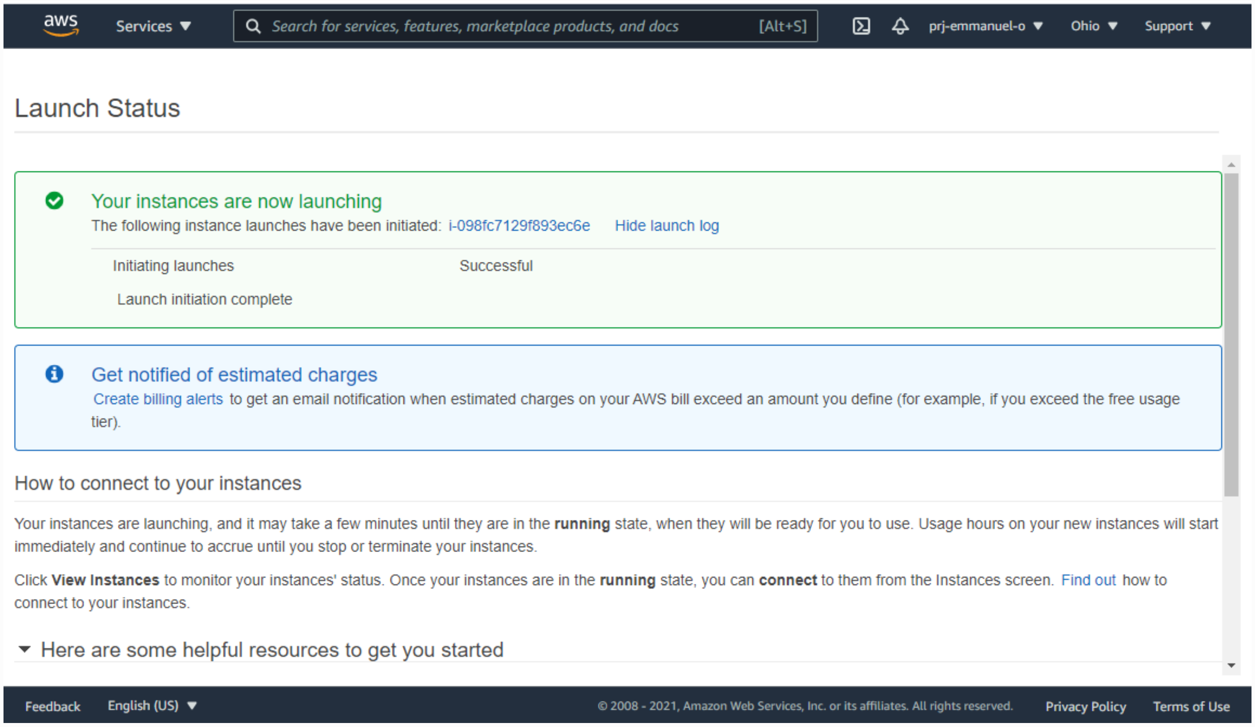
Click on select an existing security group and use the default group provided under security group ID.

Review Instance Launch

Select create a new key pair in the dialogue window that pops up

Name the key pair and download it

Launch the instance



aws
Services ▼
Search for services, features, marketplace products, and docs [Alt+S]
prj-emmanuel-o ▼
Ohio ▼
Support ▼

Launch Status

✓ Your instances are now launching

The following instance launches have been initiated: i-098fc7129f893ec6e Hide launch log

Initiating launches Successful

Launch initiation complete

Get notified of estimated charges

Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances

Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.

Click **View Instances** to monitor your instances' status. Once your instances are in the **running** state, you can **connect** to them from the Instances screen. Find out how to connect to your instances.

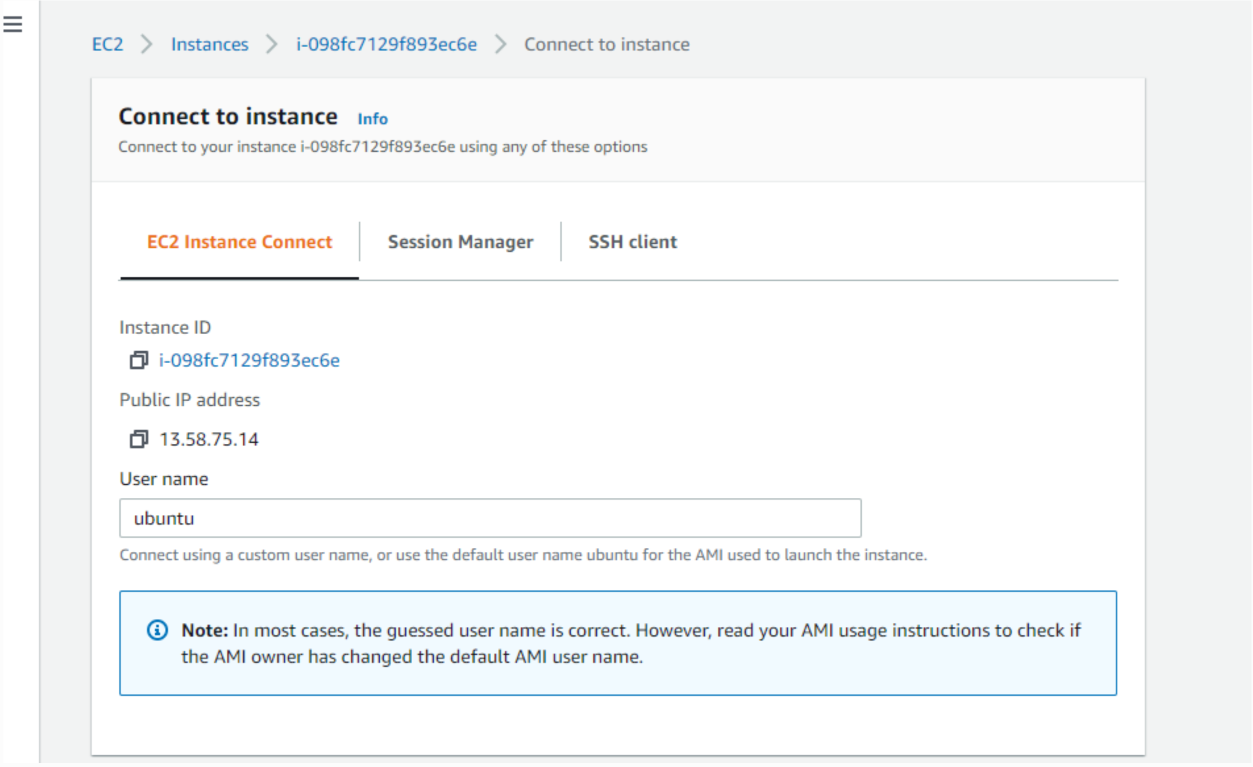
▼ Here are some helpful resources to get you started

Feedback
English (US) ▼
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Click on the instance ID **i-098fc7129f893ec6e** to view the instance.

Note: everytime you bring up an instance, it assigns a public IP needed to get into the machine (IPv4 Public IP). The IP address changes each time the computer is turned off.

Check the instance name and under Actions, click connect



Connect to instance

Connect to your instance i-098fc7129f893ec6e using any of these options

Instance ID

i-098fc7129f893ec6e

Public IP address

13.58.75.14

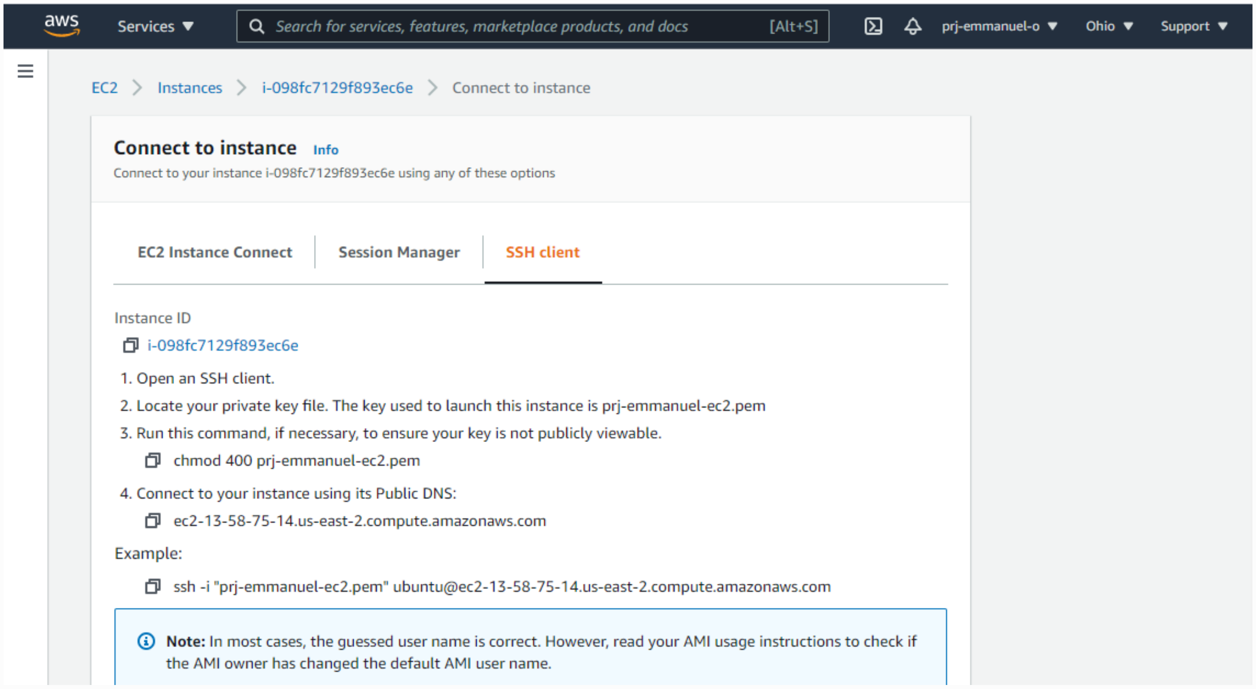
User name

ubuntu

Connect using a custom user name, or use the default user name ubuntu for the AMI used to launch the instance.

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

CLick on SSH client tab


aws
Services ▼
Search for services, features, marketplace products, and docs
[Alt+S]
prj-emmanuel-o ▼
Ohio ▼
Support ▼
EC2 > Instances > i-098fc7129f893ec6e > Connect to instance
Info
EC2 Instance Connect
Session Manager
SSH client

Connect to instance

Connect to your instance i-098fc7129f893ec6e using any of these options

Instance ID

i-098fc7129f893ec6e

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is prj-emmanuel-ec2.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.

```
chmod 400 prj-emmanuel-ec2.pem
```

4. Connect to your instance using its Public DNS:

```
ec2-13-58-75-14.us-east-2.compute.amazonaws.com
```

Example:

```
ssh -i "prj-emmanuel-ec2.pem" ubuntu@ec2-13-58-75-14.us-east-2.compute.amazonaws.com
```

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Copy the following command

```
ssh -i "prj-emmanuel-ec2.pem"
ubuntu@ec2-13-58-75-14.us-east-2.compute.amazonaws.com
```

Since am using windows, I had to download Mobaxterm free version on (https://mobaxterm.mobatek.net/download.html) to be able to connect to my EC2 instance. This application enables you to use the key pair file ec2.pem without converting to ppk as in using the Putty.

Installing Mobaxterm

Extract and install Mobaxterm.

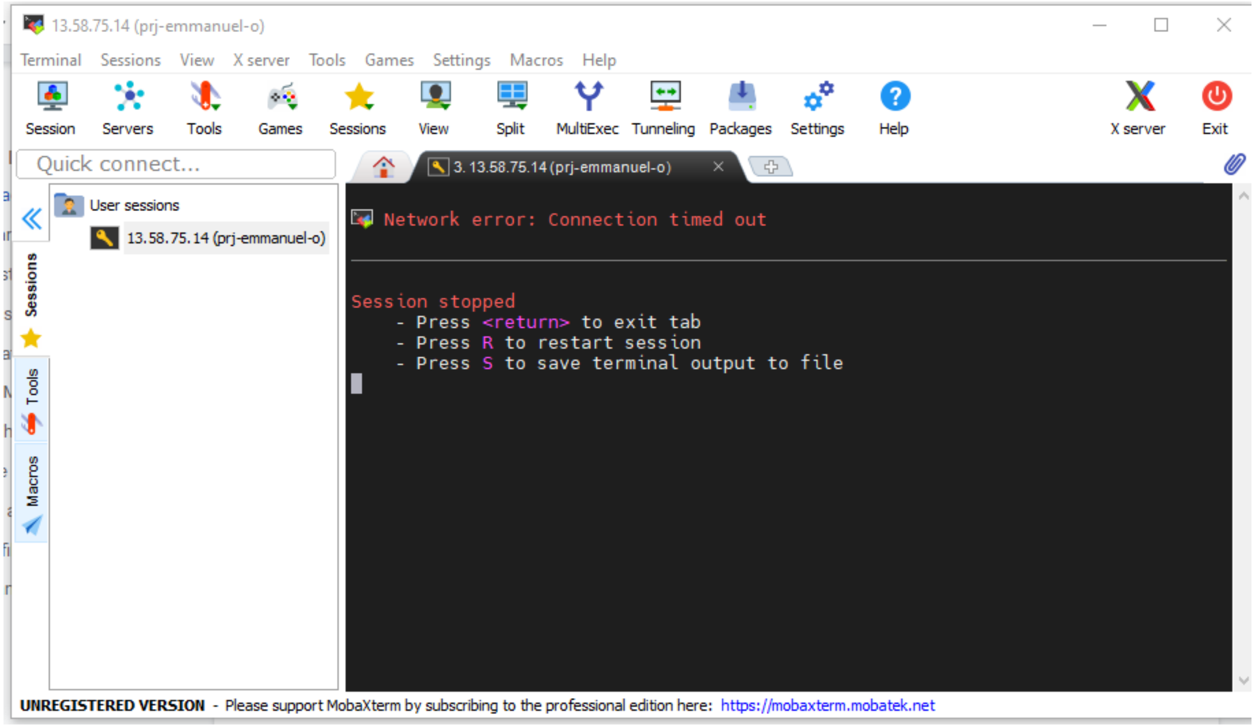
Open the application and click on session to start a session.

Select SSH as the session type

Copy the public IP address and paste into the Remote host box in the application

Click on Advances SSH settings and check the “use private key” box then load the key pair

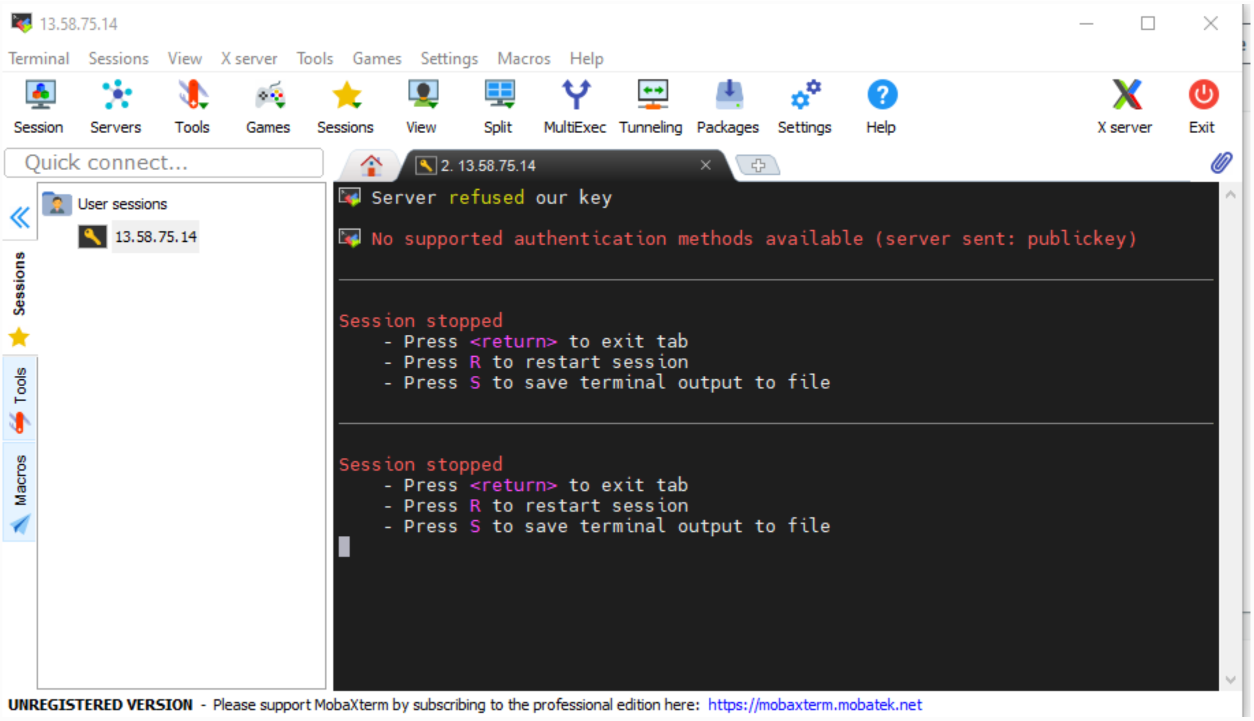
Click on specify username and enter a username and click ok to start session


13.58.75.14 (prj-emmanuel-o)
Terminal Sessions View X server Tools Games Settings Macros Help
Session
Servers
Tools
Games
Sessions
View
Split
MultiExec
Tunneling
Packages
Settings
Help
X server
Exit
Quick connect...
3. 13.58.75.14 (prj-emmanuel-o)
User sessions
13.58.75.14 (prj-emmanuel-o)
Sessions
Tools
Macros

```
Network error: Connection timed out

Session stopped
    - Press <return> to exit tab
    - Press R to restart session
    - Press S to save terminal output to file
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

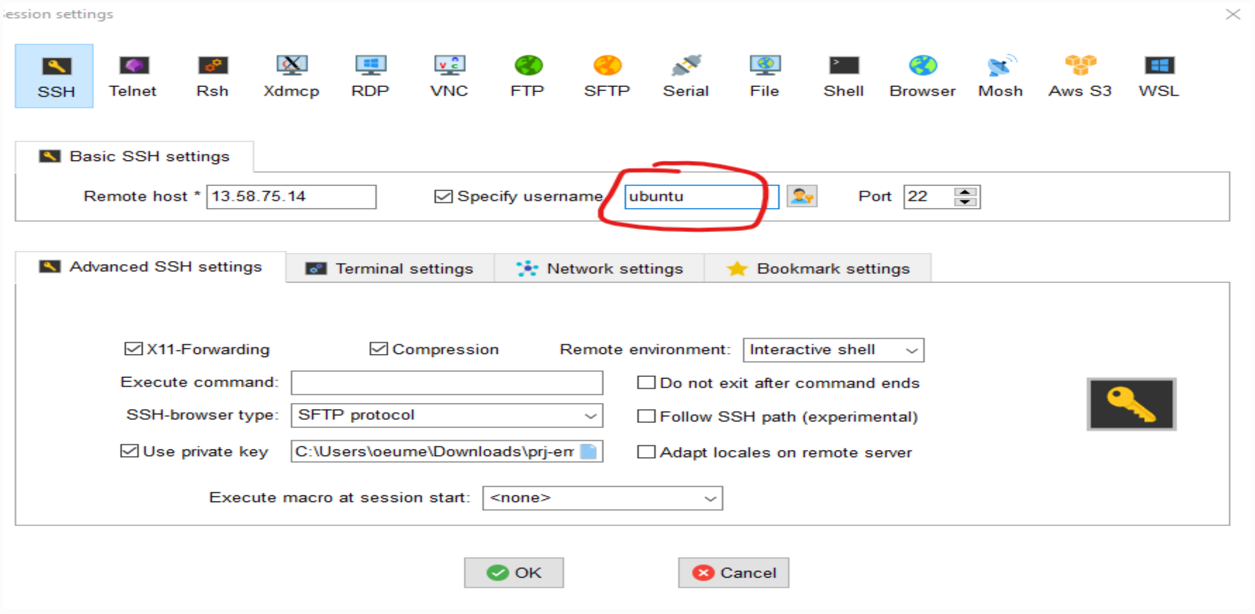
I got the above errors after trying to start up the first time



```
Server refused our key

No supported authentication methods available (server sent: publickey)

Session stopped
    - Press <return> to exit tab
    - Press R to restart session
    - Press S to save terminal output to file

Session stopped
    - Press <return> to exit tab
    - Press R to restart session
    - Press S to save terminal output to file
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

I only needed to change \the username to ubuntu and was able to connect as shown below



Remote host * 13.58.75.14

☑ Specify username ubuntu

Port 22

☑ X11-Forwarding

☑ Compression

Remote environment: Interactive shell

Execute command:

☐ Do not exit after command ends

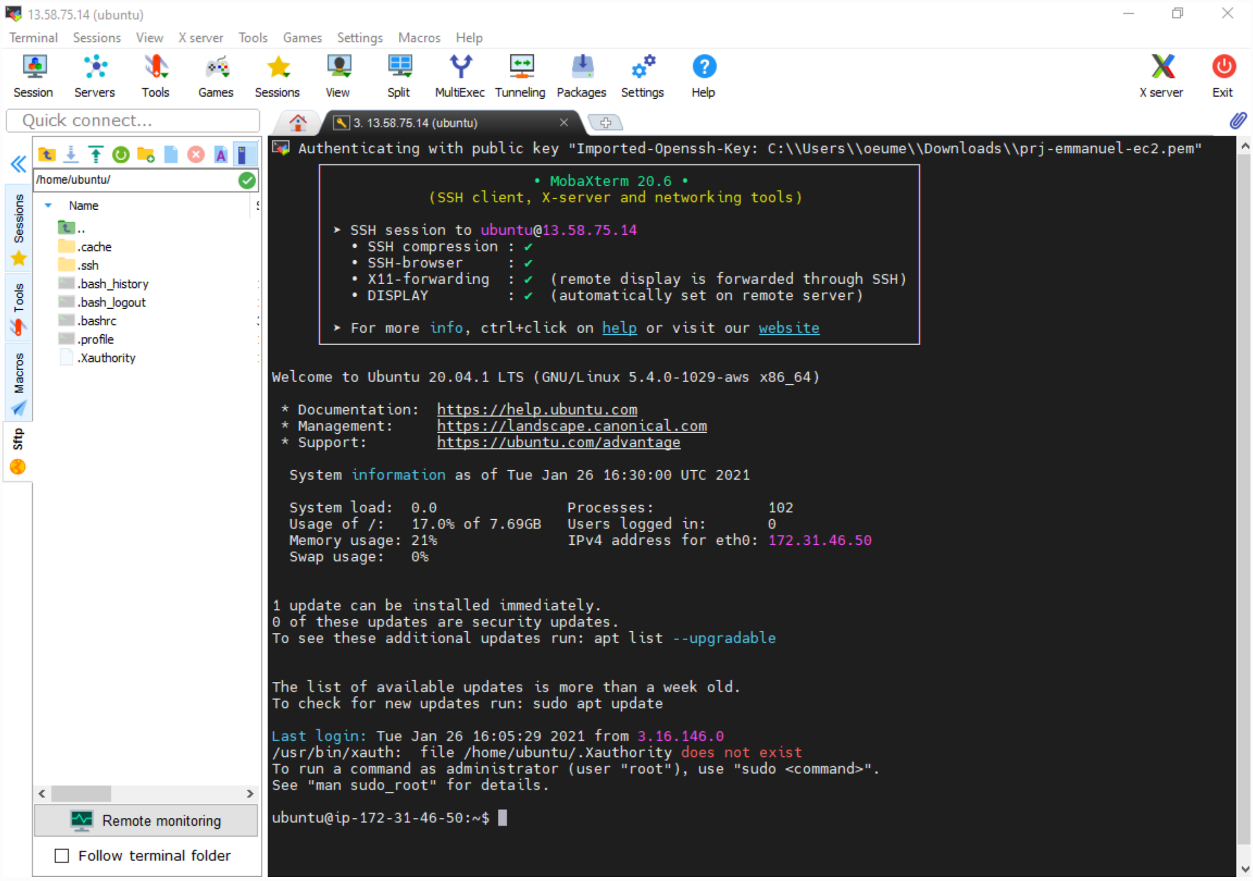
SSH-browser type: SFTP protocol

☐ Follow SSH path (experimental)

☑ Use private key C:\Users\oeume\Downloads\prj-em

☐ Adapt locales on remote server

Execute macro at session start: <none>



```
Authenticating with public key "Imported-Openssh-Key: C:\\Users\\oeume\\Downloads\\prj-emmanuel-ec2.pem"
                    • MobaXterm 20.6 •
           (SSH client, X-server and networking tools)

  ➤ SSH session to ubuntu@13.58.75.14
    • SSH compression : ✔
    • SSH-browser     : ✔
    • X11-forwarding  : ✔  (remote display is forwarded through SSH)
    • DISPLAY         : ✔  (automatically set on remote server)

  ➤ For more info, ctrl+click on help or visit our website

Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-1029-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Jan 26 16:30:00 UTC 2021

  System load:  0.0               Processes:             102
  Usage of /:   17.0% of 7.69GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for eth0: 172.31.46.50
  Swap usage:   0%


1 update can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Jan 26 16:05:29 2021 from 3.16.146.0
/usr/bin/xauth:  file /home/ubuntu/.Xauthority does not exist
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-46-50:~$
```

Step1: Install NodeJs

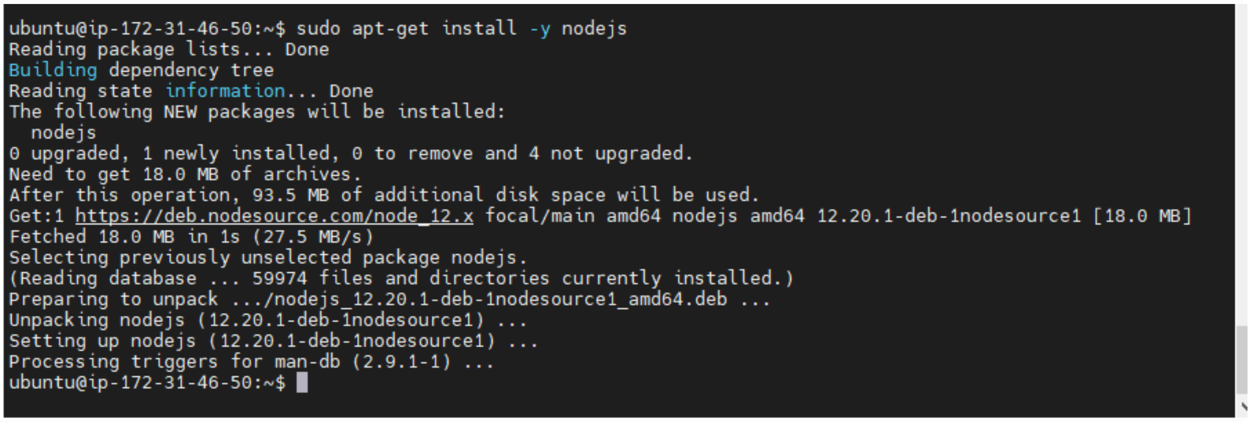
Node.js is a JavaScript runtime built on Chrome’s V8 JavaScript engine and is used to set up the Express routes and AngularJS controllers.

$ sudo apt-get install -y nodejs

Before installing NodeJs, I ran the following commands to update and upgrade ubuntu
sudo apt-get update
sudo apt-get upgrade

I also used the command below to get the location of nodejs software from ubuntu repositories over the internet.
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -

Next was to install nodeJS using **sudo apt-get install -y nodejs**

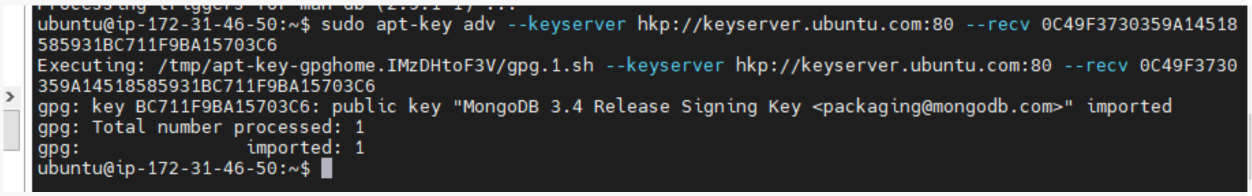


```
ubuntu@ip-172-31-46-50:~$ sudo apt-get install -y nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nodejs
0 upgraded, 1 newly installed, 0 to remove and 4 not upgraded.
Need to get 18.0 MB of archives.
After this operation, 93.5 MB of additional disk space will be used.
Get:1 https://deb.nodesource.com/node_12.x focal/main amd64 nodejs amd64 12.20.1-deb-1nodesource1 [18.0 MB]
Fetched 18.0 MB in 1s (27.5 MB/s)
Selecting previously unselected package nodejs.
(Reading database ... 59974 files and directories currently installed.)
Preparing to unpack .../nodejs_12.20.1-deb-1nodesource1_amd64.deb ...
Unpacking nodejs (12.20.1-deb-1nodesource1) ...
Setting up nodejs (12.20.1-deb-1nodesource1) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-46-50:~$
```

Step 2: Install MongoDB

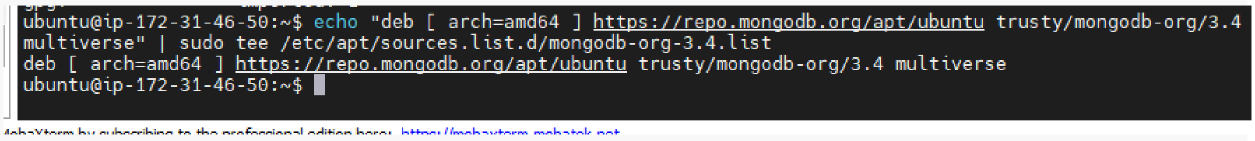
MongoDB stores data in flexible, JSON-like documents. Fields in a database can vary from document to document and data structure can be changed over time. For this project, we are adding book records to MongoDB that contain book name, isbn number, author, and number of pages.

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80
--recv 0C49F3730359A14518585931BC711F9BA15703C6
```



```
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-46-50:~$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 0C49F3730359A14518
585931BC711F9BA15703C6
Executing: /tmp/apt-key-gpghome.IMzDHtoF3V/gpg.1.sh --keyserver hkp://keyserver.ubuntu.com:80 --recv 0C49F3730
359A14518585931BC711F9BA15703C6
gpg: key BC711F9BA15703C6: public key "MongoDB 3.4 Release Signing Key <packaging@mongodb.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
ubuntu@ip-172-31-46-50:~$
```

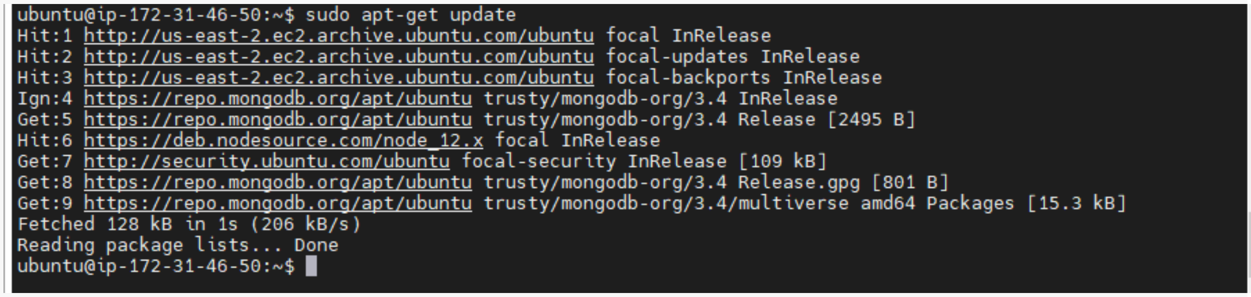
```
echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu
trusty/mongodb-org/3.4 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb-org-3.4.list
```



```
gpg:               imported: 1
ubuntu@ip-172-31-46-50:~$ echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4
multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.4.list
deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4 multiverse
ubuntu@ip-172-31-46-50:~$
MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

Next update the package manager

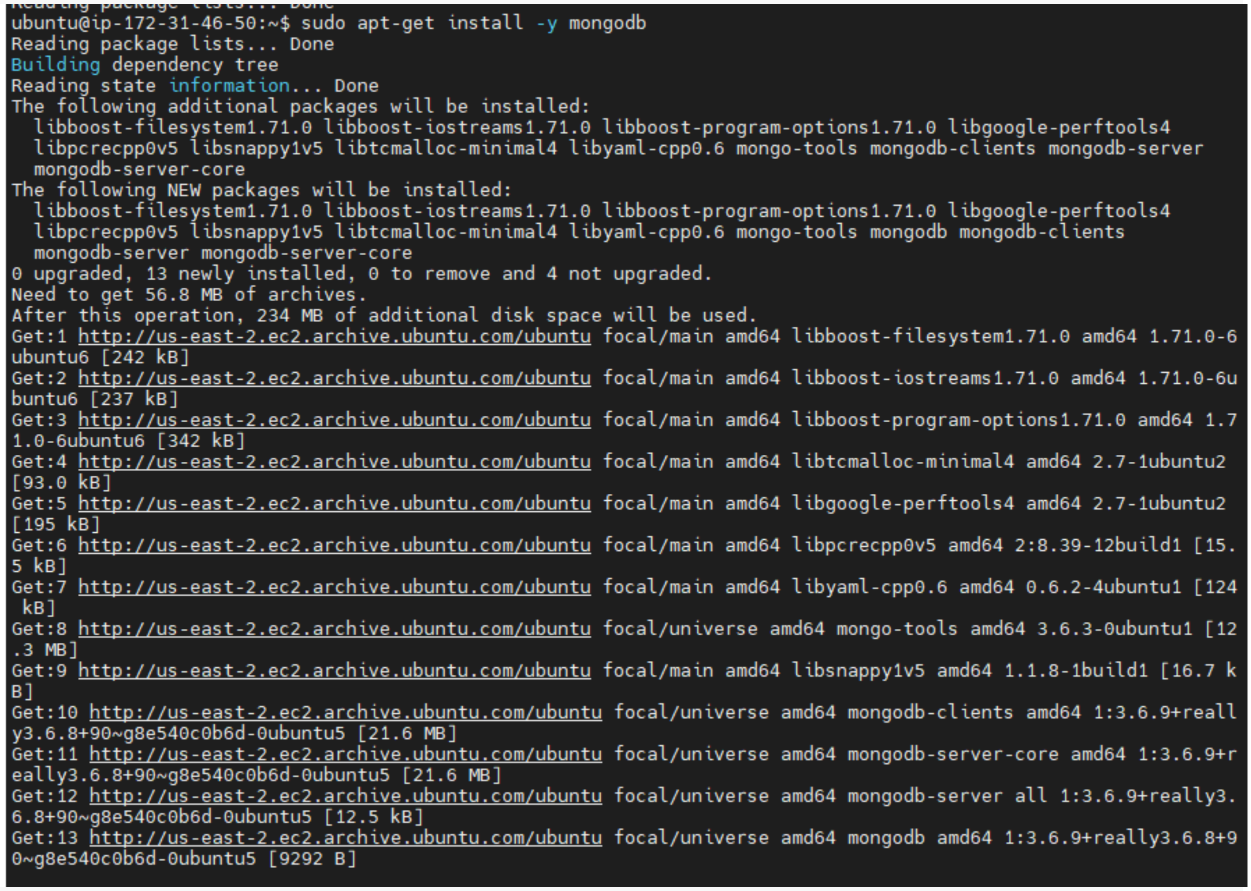
sudo apt-get update



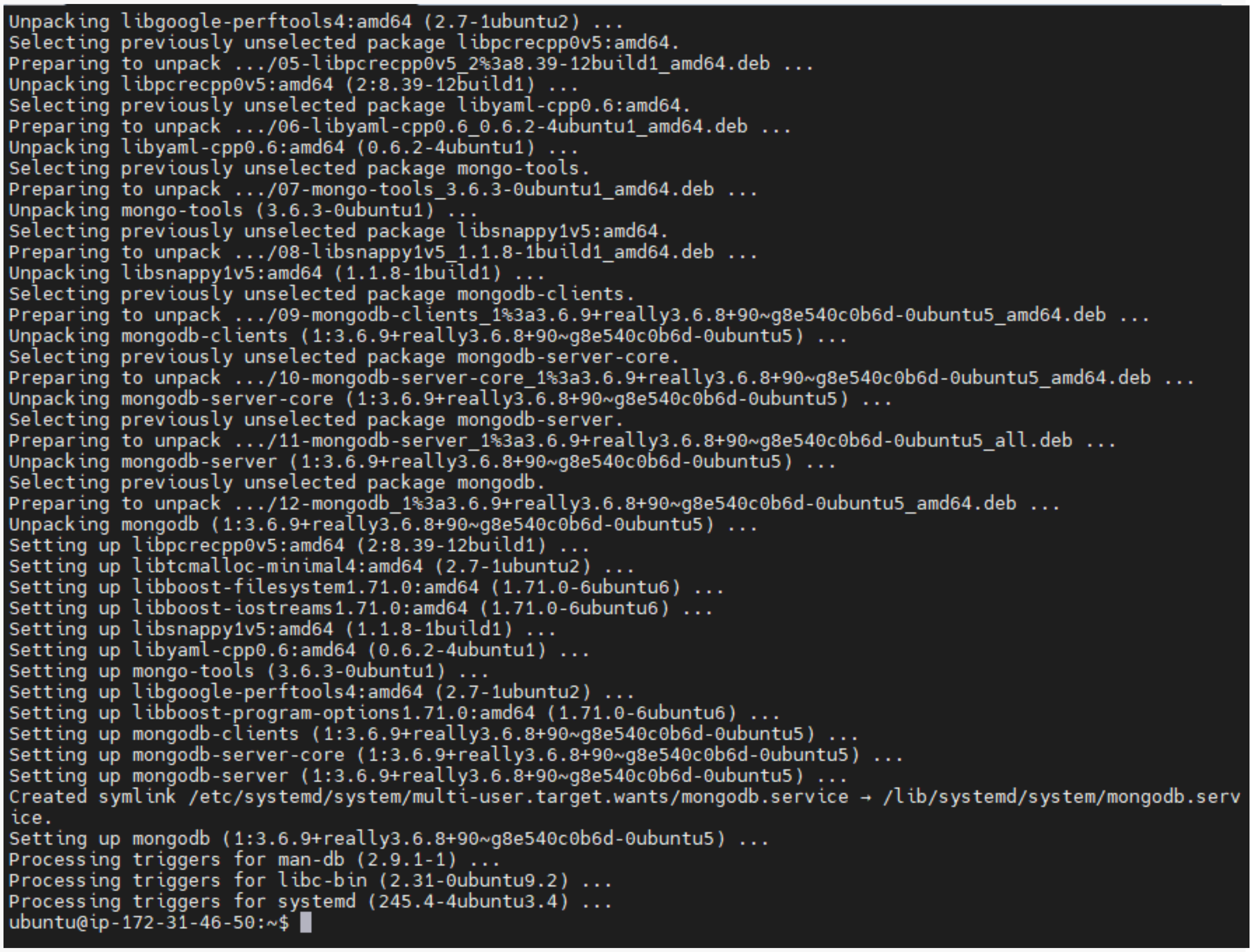
```
ubuntu@ip-172-31-46-50:~$ sudo apt-get update
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Ign:4 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4 InRelease
Get:5 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4 Release [2495 B]
Hit:6 https://deb.nodesource.com/node_12.x focal InRelease
Get:7 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Get:8 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4 Release.gpg [801 B]
Get:9 https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.4/multiverse amd64 Packages [15.3 kB]
Fetched 128 kB in 1s (206 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-46-50:~$ 
```

Install MongoDB

sudo apt-get install -y mongodb



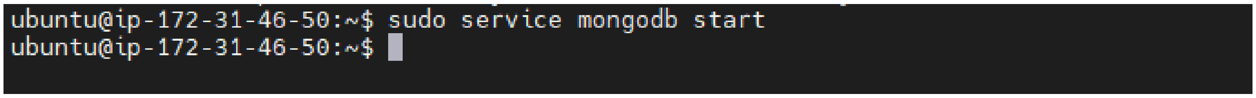
```
Reading package lists... Done
ubuntu@ip-172-31-46-50:~$ sudo apt-get install -y mongodb
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libboost-filesystem1.71.0 libboost-iostreams1.71.0 libboost-program-options1.71.0 libgoogle-perftools4
  libpcrecpp0v5 libsnappy1v5 libtcmalloc-minimal4 libyaml-cpp0.6 mongo-tools mongodb-clients mongodb-server
  mongodb-server-core
The following NEW packages will be installed:
  libboost-filesystem1.71.0 libboost-iostreams1.71.0 libboost-program-options1.71.0 libgoogle-perftools4
  libpcrecpp0v5 libsnappy1v5 libtcmalloc-minimal4 libyaml-cpp0.6 mongo-tools mongodb mongodb-clients
  mongodb-server mongodb-server-core
0 upgraded, 13 newly installed, 0 to remove and 4 not upgraded.
Need to get 56.8 MB of archives.
After this operation, 234 MB of additional disk space will be used.
Get:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libboost-filesystem1.71.0 amd64 1.71.0-6
ubuntu6 [242 kB]
Get:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libboost-iostreams1.71.0 amd64 1.71.0-6u
buntu6 [237 kB]
Get:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libboost-program-options1.71.0 amd64 1.7
1.0-6ubuntu6 [342 kB]
Get:4 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libtcmalloc-minimal4 amd64 2.7-1ubuntu2
[93.0 kB]
Get:5 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libgoogle-perftools4 amd64 2.7-1ubuntu2
[195 kB]
Get:6 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libpcrecpp0v5 amd64 2:8.39-12build1 [15.
5 kB]
Get:7 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libyaml-cpp0.6 amd64 0.6.2-4ubuntu1 [124
 kB]
Get:8 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 mongo-tools amd64 3.6.3-0ubuntu1 [12
.3 MB]
Get:9 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 libsnappy1v5 amd64 1.1.8-1build1 [16.7 k
B]
Get:10 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 mongodb-clients amd64 1:3.6.9+reall
y3.6.8+90~g8e540c0b6d-0ubuntu5 [21.6 MB]
Get:11 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 mongodb-server-core amd64 1:3.6.9+r
eally3.6.8+90~g8e540c0b6d-0ubuntu5 [21.6 MB]
Get:12 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 mongodb-server all 1:3.6.9+really3.
6.8+90~g8e540c0b6d-0ubuntu5 [12.5 kB]
Get:13 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 mongodb amd64 1:3.6.9+really3.6.8+9
0~g8e540c0b6d-0ubuntu5 [9292 B]
```



```
Unpacking libgoogle-perftools4:amd64 (2.7-1ubuntu2) ...
Selecting previously unselected package libpcrecpp0v5:amd64.
Preparing to unpack .../05-libpcrecpp0v5_2%3a8.39-12build1_amd64.deb ...
Unpacking libpcrecpp0v5:amd64 (2:8.39-12build1) ...
Selecting previously unselected package libyaml-cpp0.6:amd64.
Preparing to unpack .../06-libyaml-cpp0.6_0.6.2-4ubuntu1_amd64.deb ...
Unpacking libyaml-cpp0.6:amd64 (0.6.2-4ubuntu1) ...
Selecting previously unselected package mongo-tools.
Preparing to unpack .../07-mongo-tools_3.6.3-0ubuntu1_amd64.deb ...
Unpacking mongo-tools (3.6.3-0ubuntu1) ...
Selecting previously unselected package libsnappy1v5:amd64.
Preparing to unpack .../08-libsnappy1v5_1.1.8-1build1_amd64.deb ...
Unpacking libsnappy1v5:amd64 (1.1.8-1build1) ...
Selecting previously unselected package mongodb-clients.
Preparing to unpack .../09-mongodb-clients_1%3a3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5_amd64.deb ...
Unpacking mongodb-clients (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Selecting previously unselected package mongodb-server-core.
Preparing to unpack .../10-mongodb-server-core_1%3a3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5_amd64.deb ...
Unpacking mongodb-server-core (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Selecting previously unselected package mongodb-server.
Preparing to unpack .../11-mongodb-server_1%3a3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5_all.deb ...
Unpacking mongodb-server (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Selecting previously unselected package mongodb.
Preparing to unpack .../12-mongodb_1%3a3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5_amd64.deb ...
Unpacking mongodb (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Setting up libpcrecpp0v5:amd64 (2:8.39-12build1) ...
Setting up libtcmalloc-minimal4:amd64 (2.7-1ubuntu2) ...
Setting up libboost-filesystem1.71.0:amd64 (1.71.0-6ubuntu6) ...
Setting up libboost-iostreams1.71.0:amd64 (1.71.0-6ubuntu6) ...
Setting up libsnappy1v5:amd64 (1.1.8-1build1) ...
Setting up libyaml-cpp0.6:amd64 (0.6.2-4ubuntu1) ...
Setting up mongo-tools (3.6.3-0ubuntu1) ...
Setting up libgoogle-perftools4:amd64 (2.7-1ubuntu2) ...
Setting up libboost-program-options1.71.0:amd64 (1.71.0-6ubuntu6) ...
Setting up mongodb-clients (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Setting up mongodb-server-core (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Setting up mongodb-server (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Created symlink /etc/systemd/system/multi-user.target.wants/mongodb.service → /lib/systemd/system/mongodb.serv
ice.
Setting up mongodb (1:3.6.9+really3.6.8+90~g8e540c0b6d-0ubuntu5) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Processing triggers for systemd (245.4-4ubuntu3.4) ...
ubuntu@ip-172-31-46-50:~$
```

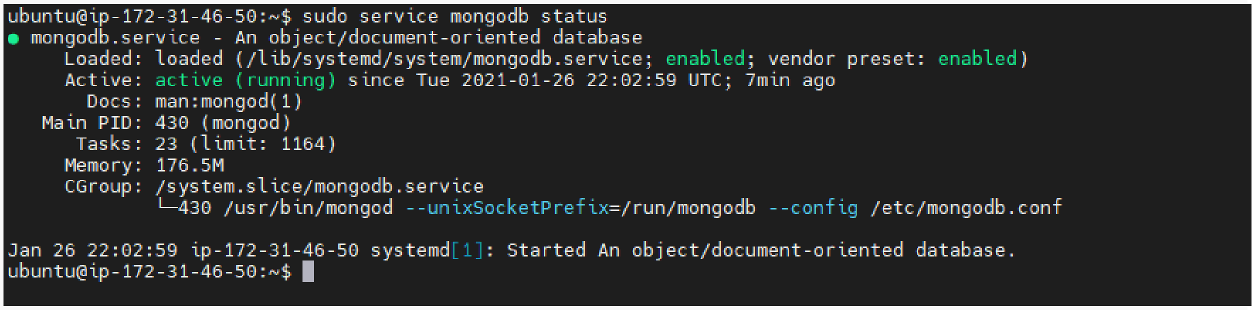
Start The server

```
sudo service mongodb start
```



```
ubuntu@ip-172-31-46-50:~$ sudo service mongodb start
ubuntu@ip-172-31-46-50:~$
```

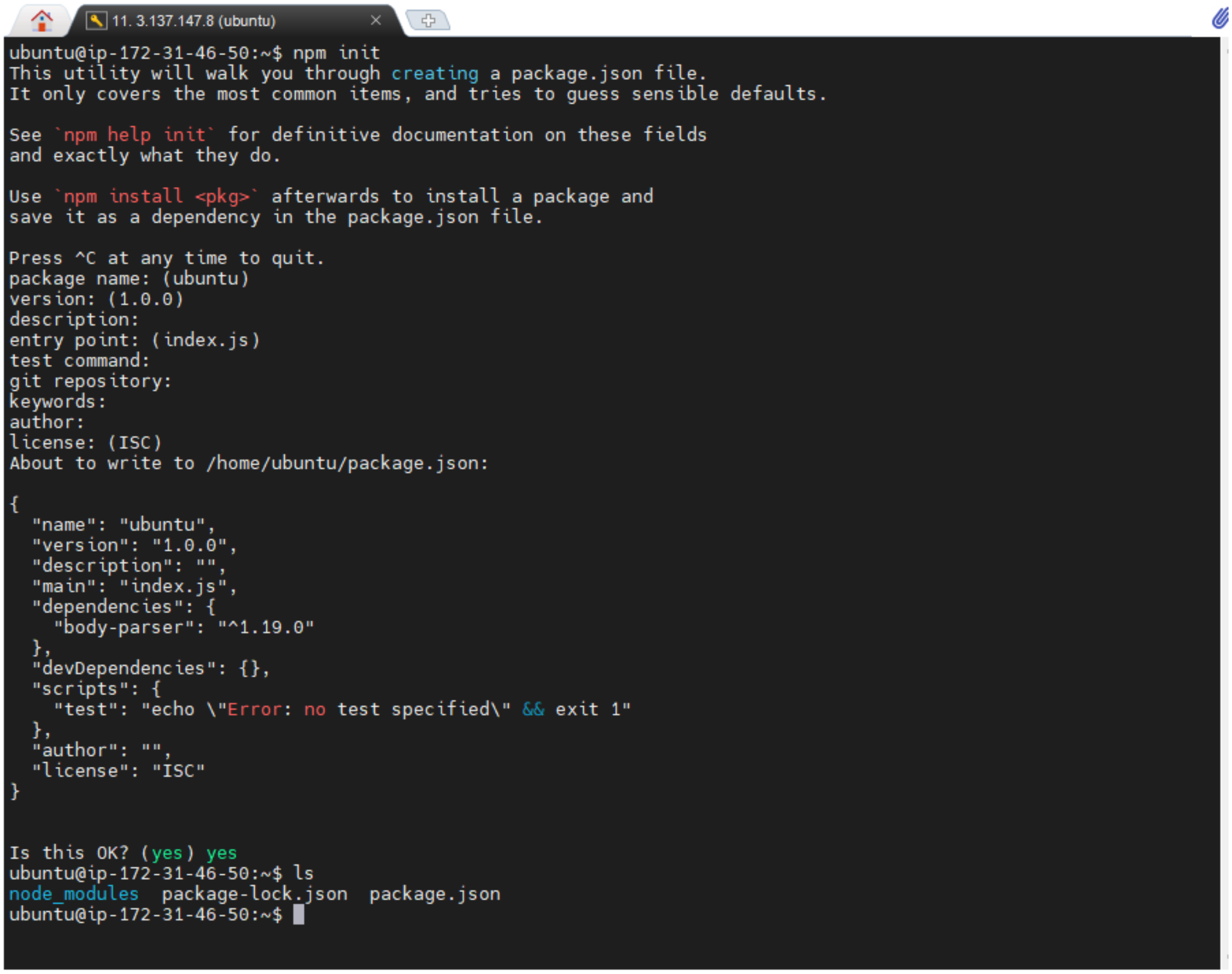
I restarted the instance and then ran status command.
sudo service mongodb status



```
ubuntu@ip-172-31-46-50:~$ sudo service mongodb status
● mongodb.service - An object/document-oriented database
     Loaded: loaded (/lib/systemd/system/mongodb.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2021-01-26 22:02:59 UTC; 7min ago
       Docs: man:mongod(1)
   Main PID: 430 (mongod)
      Tasks: 23 (limit: 1164)
     Memory: 176.5M
     CGroup: /system.slice/mongodb.service
             └─430 /usr/bin/mongod --unixSocketPrefix=/run/mongodb --config /etc/mongodb.conf

Jan 26 22:02:59 ip-172-31-46-50 systemd[1]: Started An object/document-oriented database.
ubuntu@ip-172-31-46-50:~$
```

We also need to install the body-parser package to help us process the JSON passed in requests to the server. Install the npm package manager. sudo apt-get install npm

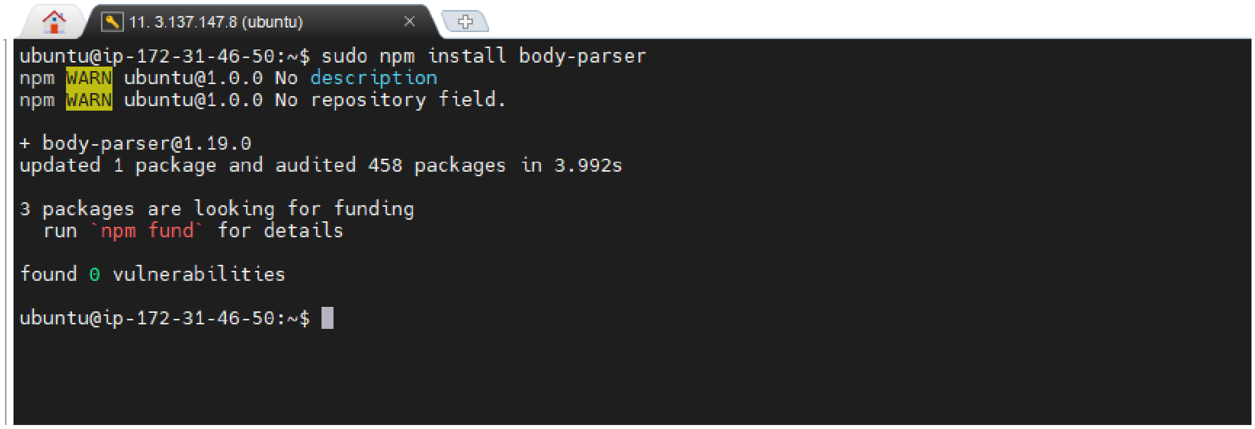
Run **npm init** to initialise the project, so that a new file named `package.json` will be created.



```
ubuntu@ip-172-31-46-50:~$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (ubuntu)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/ubuntu/package.json:

{
  "name": "ubuntu",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "dependencies": {
    "body-parser": "^1.19.0"
  },
  "devDependencies": {},
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}


Is this OK? (yes) yes
ubuntu@ip-172-31-46-50:~$ ls
node_modules  package-lock.json  package.json
ubuntu@ip-172-31-46-50:~$
```

Install the body-parser package

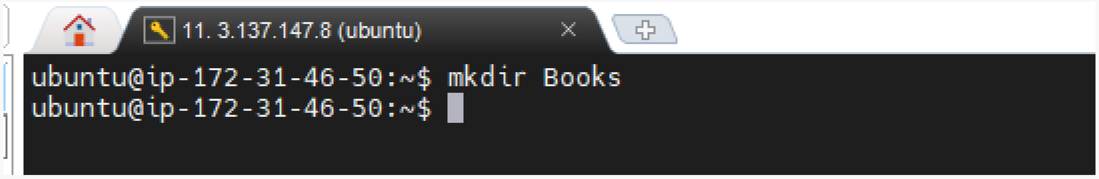
```
sudo npm install body-parser
```



```
ubuntu@ip-172-31-46-50:~$ sudo npm install body-parser
npm WARN ubuntu@1.0.0 No description
npm WARN ubuntu@1.0.0 No repository field.

+ body-parser@1.19.0
updated 1 package and audited 458 packages in 3.992s

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

found 0 vulnerabilities

ubuntu@ip-172-31-46-50:~$
```

Create a folder named Books

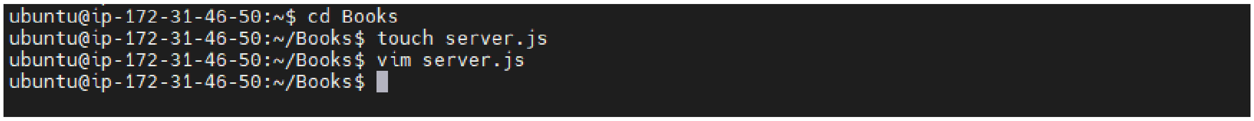
```
mkdir Books
```



```
ubuntu@ip-172-31-46-50:~$ mkdir Books
ubuntu@ip-172-31-46-50:~$
```

Add a file to it named server.js

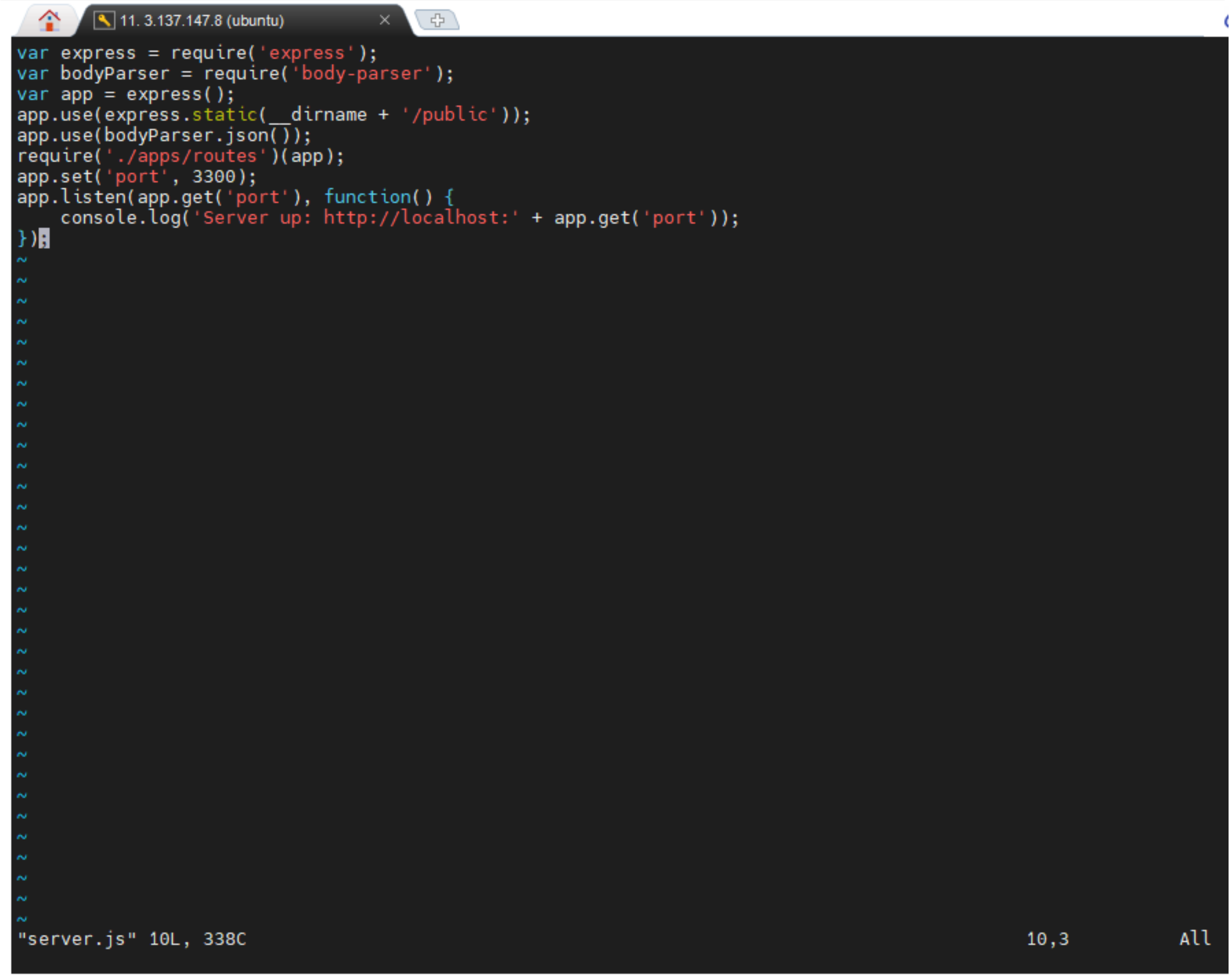
```
touch server.js
```



```
ubuntu@ip-172-31-46-50:~$ cd Books
ubuntu@ip-172-31-46-50:~/Books$ touch server.js
ubuntu@ip-172-31-46-50:~/Books$ vim server.js
ubuntu@ip-172-31-46-50:~/Books$
```

Copy and paste the web server code below into the server.js file.

```
var express = require('express');
var bodyParser = require('body-parser');
var app = express();
app.use(express.static(__dirname + '/public'));
app.use(bodyParser.json());
require('./apps/routes')(app);
app.set('port', 3300);
app.listen(app.get('port'), function() {
    console.log('Server up: http://localhost:' +
app.get('port'));
});
```



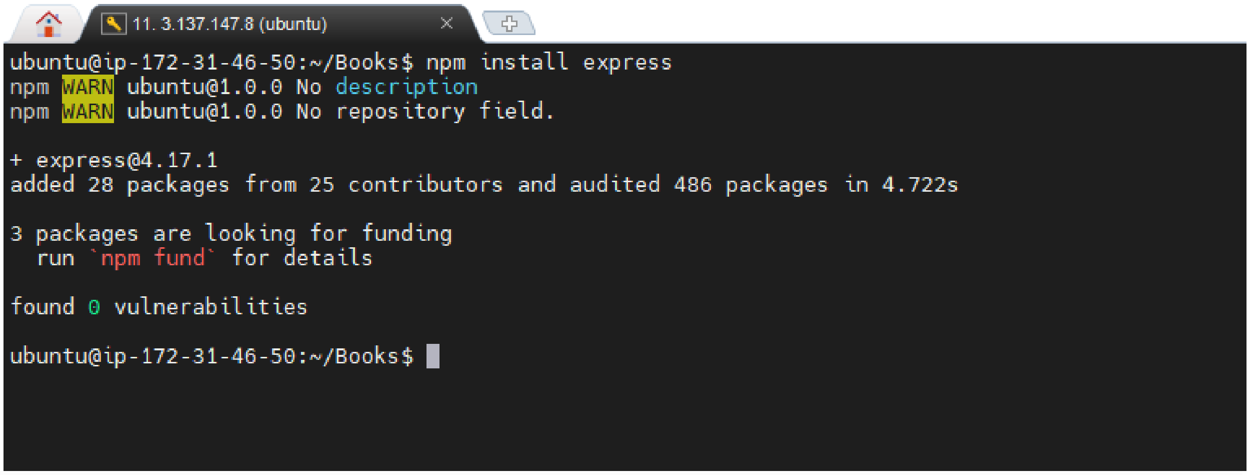
```
var express = require('express');
var bodyParser = require('body-parser');
var app = express();
app.use(express.static(__dirname + '/public'));
app.use(bodyParser.json());
require('./apps/routes')(app);
app.set('port', 3300);
app.listen(app.get('port'), function() {
    console.log('Server up: http://localhost:' + app.get('port'));
});
~
"server.js" 10L, 338C                                          10,3          All
```

Step 3: Install Express and set up routes to the server

Express is a minimal and flexible Node.js web application framework that provides features for web and mobile applications. Express was used in this project to pass book information to and from the MongoDB database. Mongoose provides a straight-forward, schema-based solution to model application data. Mongoose was used in this project to provide a book schema for the database.

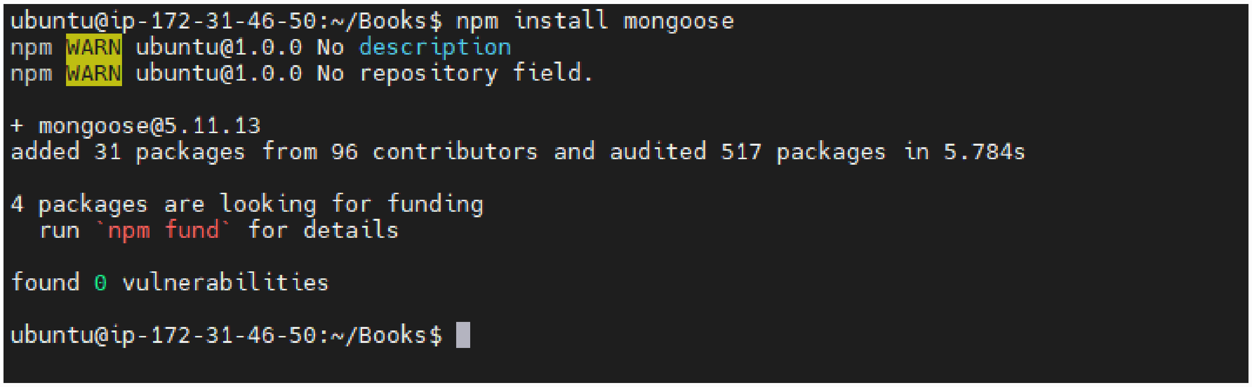
Next step is to run npm install express and np, install mongoose in the Books folder.

npm install express



```
ubuntu@ip-172-31-46-50:~/Books$ npm install express
npm WARN ubuntu@1.0.0 No description
npm WARN ubuntu@1.0.0 No repository field.

+ express@4.17.1
added 28 packages from 25 contributors and audited 486 packages in 4.722s

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

found 0 vulnerabilities

ubuntu@ip-172-31-46-50:~/Books$
```

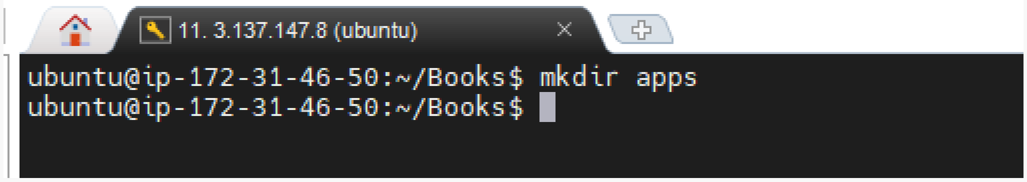
npm install mongoose



```
ubuntu@ip-172-31-46-50:~/Books$ npm install mongoose
npm WARN ubuntu@1.0.0 No description
npm WARN ubuntu@1.0.0 No repository field.

+ mongoose@5.11.13
added 31 packages from 96 contributors and audited 517 packages in 5.784s

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

found 0 vulnerabilities

ubuntu@ip-172-31-46-50:~/Books$
```

Create a folder named apps

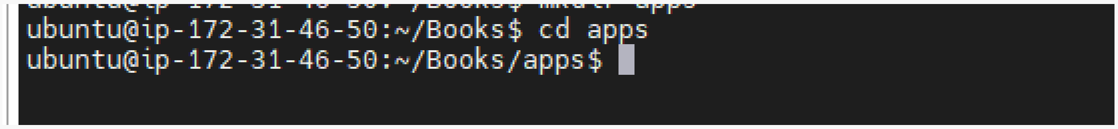
```
mkdir apps
```



```
ubuntu@ip-172-31-46-50:~/Books$ mkdir apps
ubuntu@ip-172-31-46-50:~/Books$
```

Move into apps folder

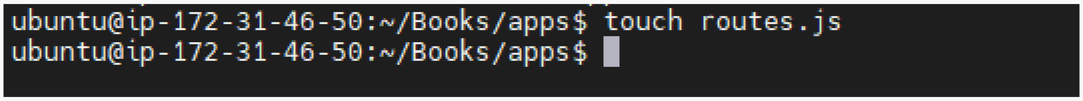
cd apps



```
ubuntu@ip-172-31-46-50:~/Books$ mkdir apps
ubuntu@ip-172-31-46-50:~/Books$ cd apps
ubuntu@ip-172-31-46-50:~/Books/apps$
```

Add a file named routes.js

touch routes.js

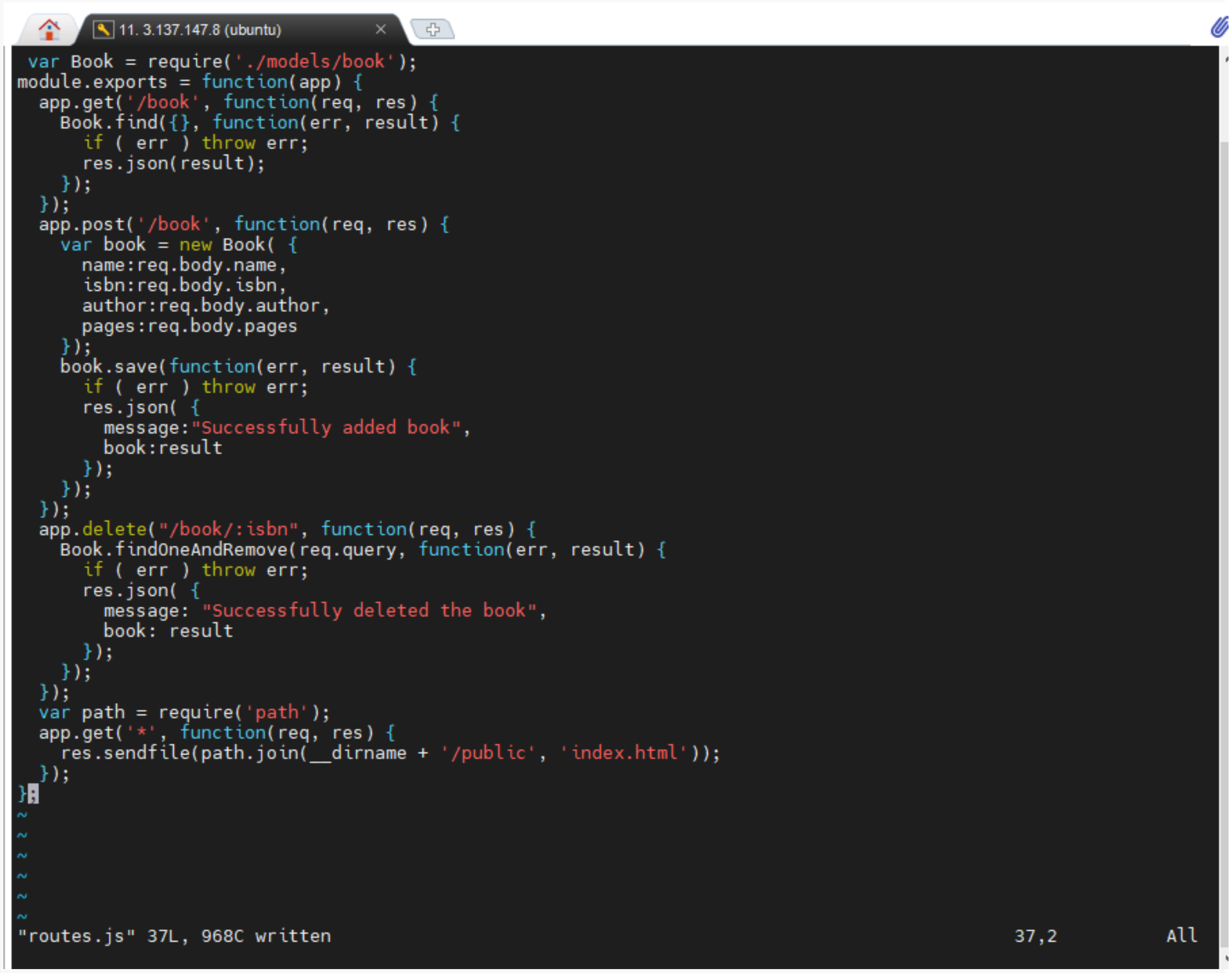


```
ubuntu@ip-172-31-46-50:~/Books/apps$ touch routes.js
ubuntu@ip-172-31-46-50:~/Books/apps$
```

Copy and paste the code below into routes.js

```
    var Book = require('./models/book');
module.exports = function(app) {
  app.get('/book', function(req, res) {
    Book.find({}, function(err, result) {
      if ( err ) throw err;
      res.json(result);
    });
  });
  app.post('/book', function(req, res) {
    var book = new Book( {
      name:req.body.name,
      isbn:req.body.isbn,
      author:req.body.author,
      pages:req.body.pages
    });
    book.save(function(err, result) {
      if ( err ) throw err;
      res.json( {
        message:"Successfully added book",
        book:result
      });
    });
  });
  app.delete("/book/:isbn", function(req, res) {
    Book.findOneAndRemove(req.query, function(err, result) {
      if ( err ) throw err;
      res.json( {
        message: "Successfully deleted the book",
        book: result
      });
    });
  });
  var path = require('path');
  app.get('*', function(req, res) {
    res.sendfile(path.join(__dirname + '/public',
'index.html'));
  });
};
```

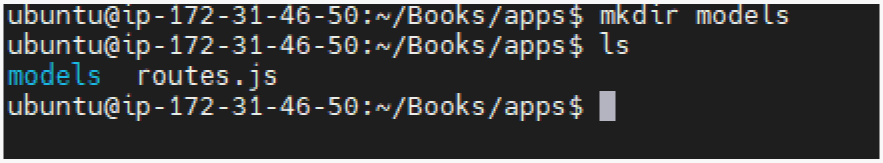
```
vim routes.js
```



```
 var Book = require('./models/book');
module.exports = function(app) {
  app.get('/book', function(req, res) {
    Book.find({}, function(err, result) {
      if ( err ) throw err;
      res.json(result);
    });
  });
  app.post('/book', function(req, res) {
    var book = new Book( {
      name:req.body.name,
      isbn:req.body.isbn,
      author:req.body.author,
      pages:req.body.pages
    });
    book.save(function(err, result) {
      if ( err ) throw err;
      res.json( {
        message:"Successfully added book",
        book:result
      });
    });
  });
  app.delete("/book/:isbn", function(req, res) {
    Book.findOneAndRemove(req.query, function(err, result) {
      if ( err ) throw err;
      res.json( {
        message: "Successfully deleted the book",
        book: result
      });
    });
  });
  var path = require('path');
  app.get('*', function(req, res) {
    res.sendfile(path.join(__dirname + '/public', 'index.html'));
  });
};
~
~
~
~
~
~
"routes.js" 37L, 968C written                                          37,2          All
```

In the apps folder, create a folder named models

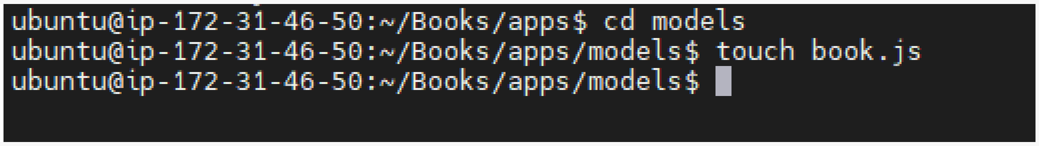
mkdir models



```
ubuntu@ip-172-31-46-50:~/Books/apps$ mkdir models
ubuntu@ip-172-31-46-50:~/Books/apps$ ls
models  routes.js
ubuntu@ip-172-31-46-50:~/Books/apps$
```

Add a file named book.js

```
touch book.js
```

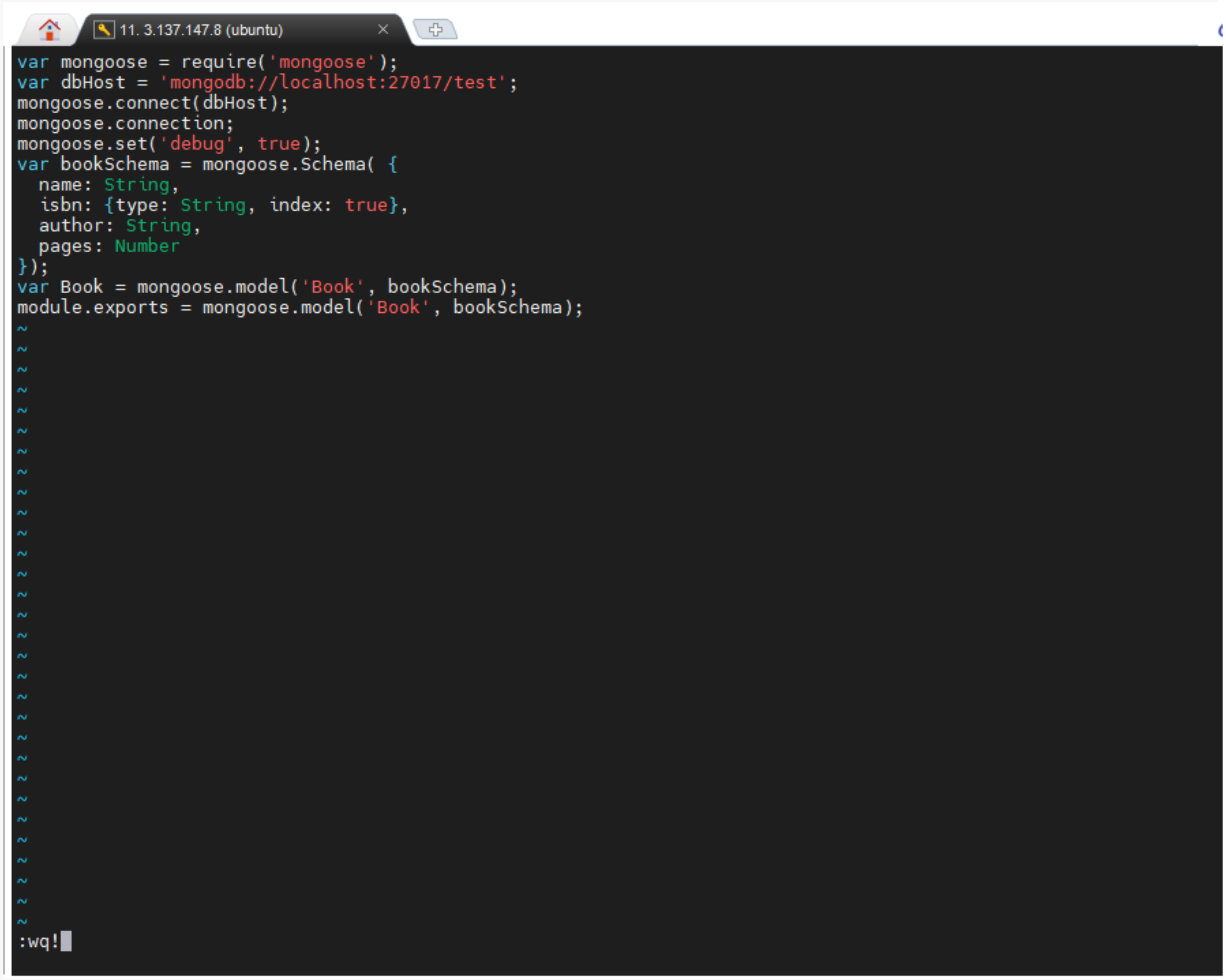


```
ubuntu@ip-172-31-46-50:~/Books/apps$ cd models
ubuntu@ip-172-31-46-50:~/Books/apps/models$ touch book.js
ubuntu@ip-172-31-46-50:~/Books/apps/models$
```

Copy and paste the code below into book.js

```
var mongoose = require('mongoose');
var dbHost = 'mongodb://localhost:27017/test';
mongoose.connect(dbHost);
mongoose.connection;
mongoose.set('debug', true);
var bookSchema = mongoose.Schema( {
  name: String,
  isbn: {type: String, index: true},
  author: String,
  pages: Number
});
var Book = mongoose.model('Book', bookSchema);
module.exports = mongoose.model('Book', bookSchema);
```

```
vim book.js
```



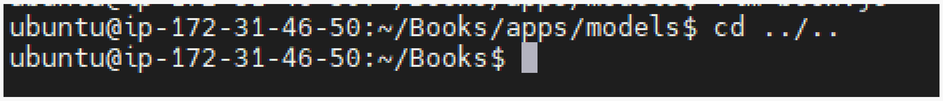
```
var mongoose = require('mongoose');
var dbHost = 'mongodb://localhost:27017/test';
mongoose.connect(dbHost);
mongoose.connection;
mongoose.set('debug', true);
var bookSchema = mongoose.Schema( {
  name: String,
  isbn: {type: String, index: true},
  author: String,
  pages: Number
});
var Book = mongoose.model('Book', bookSchema);
module.exports = mongoose.model('Book', bookSchema);
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq!
```

Access the routes with AngularJS

AngularJS provides a web framework for creating dynamic views in your web applications. In this tutorial, we use AngularJS to connect our web page with Express and perform actions on our book database.

Change the directory back up to Books

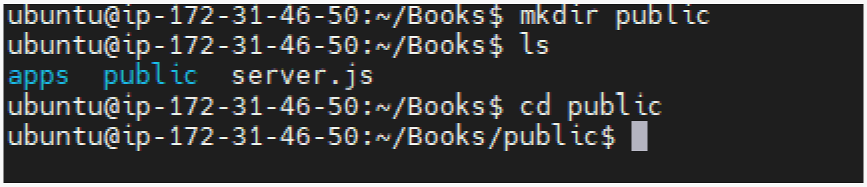
```
cd ../..
```



```
ubuntu@ip-172-31-46-50:~/Books/apps/models$ cd ../..
ubuntu@ip-172-31-46-50:~/Books$
```

Create a folder named public

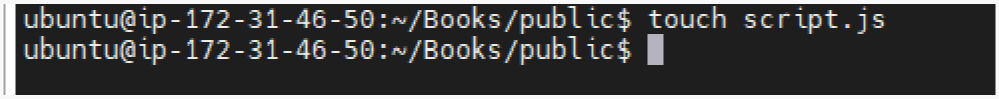
mkdir public and move into public using **cd public**



```
ubuntu@ip-172-31-46-50:~/Books$ mkdir public
ubuntu@ip-172-31-46-50:~/Books$ ls
apps  public  server.js
ubuntu@ip-172-31-46-50:~/Books$ cd public
ubuntu@ip-172-31-46-50:~/Books/public$
```

Add a file named script.js

```
touch script.js
```

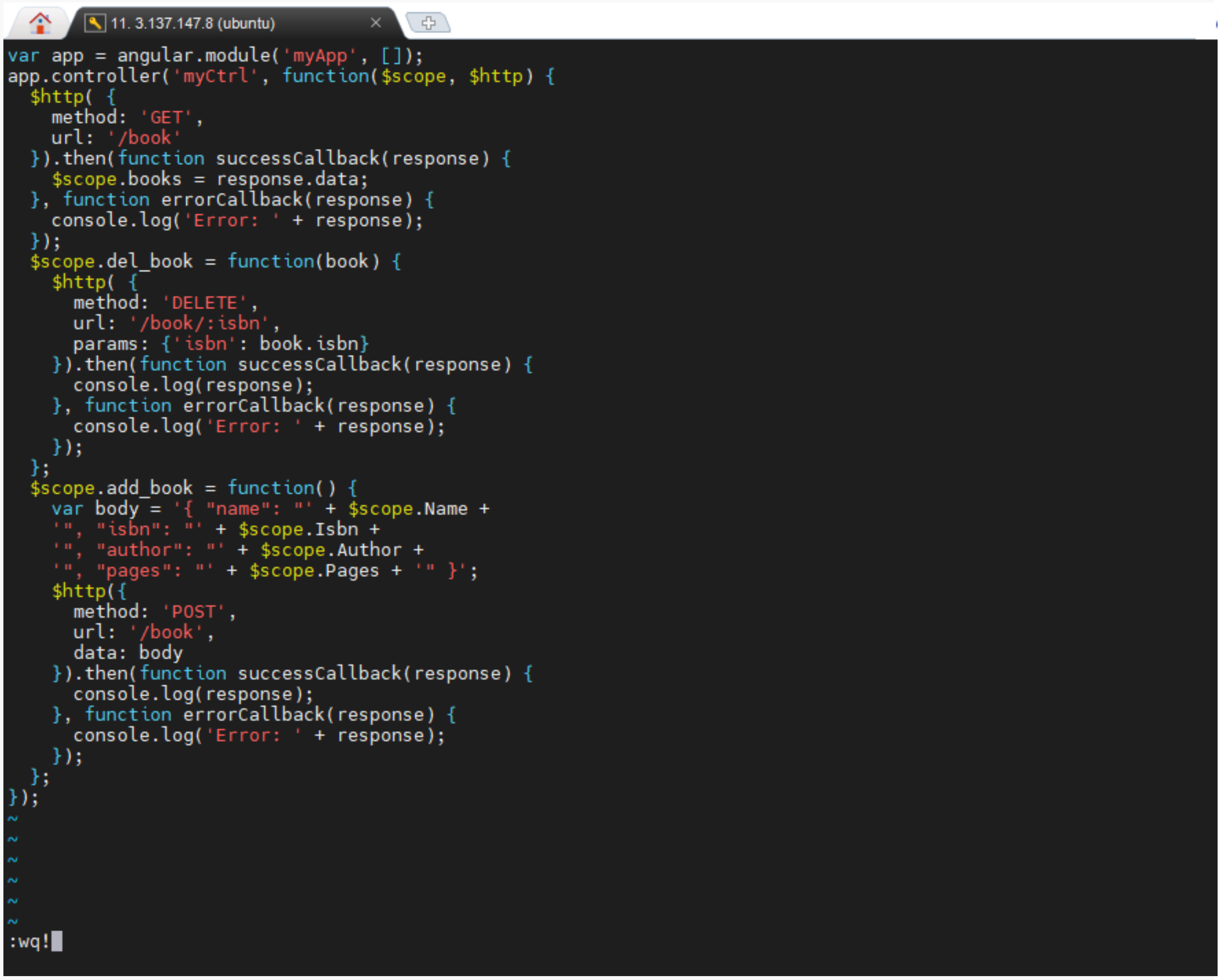


```
ubuntu@ip-172-31-46-50:~/Books/public$ touch script.js
ubuntu@ip-172-31-46-50:~/Books/public$
```

Copy and paste the required Code (controller configuration defined) into the script.js file.

```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $http) {
  $http( {
    method: 'GET',
    url: '/book'
  }).then(function successCallback(response) {
    $scope.books = response.data;
  }, function errorCallback(response) {
    console.log('Error: ' + response);
  });
  $scope.del_book = function(book) {
    $http( {
      method: 'DELETE',
      url: '/book/:isbn',
      params: {'isbn': book.isbn}
    }).then(function successCallback(response) {
      console.log(response);
    }, function errorCallback(response) {
      console.log('Error: ' + response);
    });
  };
  $scope.add_book = function() {
    var body = '{ "name": "' + $scope.Name +
    '", "isbn": "' + $scope.Isbn +
    '", "author": "' + $scope.Author +
    '", "pages": "' + $scope.Pages + '" }';
    $http({
      method: 'POST',
      url: '/book',
      data: body
    }).then(function successCallback(response) {
      console.log(response);
    }, function errorCallback(response) {
      console.log('Error: ' + response);
    });
  };
});
```

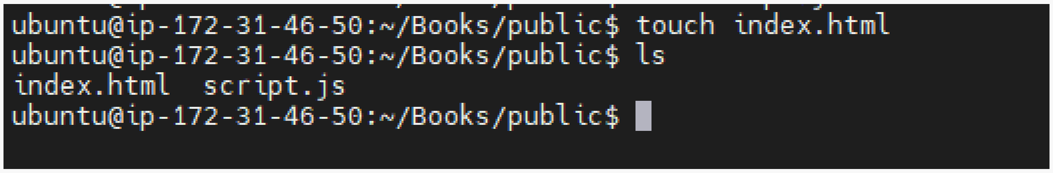
```
vim script.js
```



```
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $http) {
  $http( {
    method: 'GET',
    url: '/book'
  }).then(function successCallback(response) {
    $scope.books = response.data;
  }, function errorCallback(response) {
    console.log('Error: ' + response);
  });
  $scope.del_book = function(book) {
    $http( {
      method: 'DELETE',
      url: '/book/:isbn',
      params: {'isbn': book.isbn}
    }).then(function successCallback(response) {
      console.log(response);
    }, function errorCallback(response) {
      console.log('Error: ' + response);
    });
  };
  $scope.add_book = function() {
    var body = '{ "name": "' + $scope.Name +
    '", "isbn": "' + $scope.Isbn +
    '", "author": "' + $scope.Author +
    '", "pages": "' + $scope.Pages + '" }';
    $http({
      method: 'POST',
      url: '/book',
      data: body
    }).then(function successCallback(response) {
      console.log(response);
    }, function errorCallback(response) {
      console.log('Error: ' + response);
    });
  };
});
~
~
~
~
~
~
:wq!
```

In the public folder, create a file named index.html

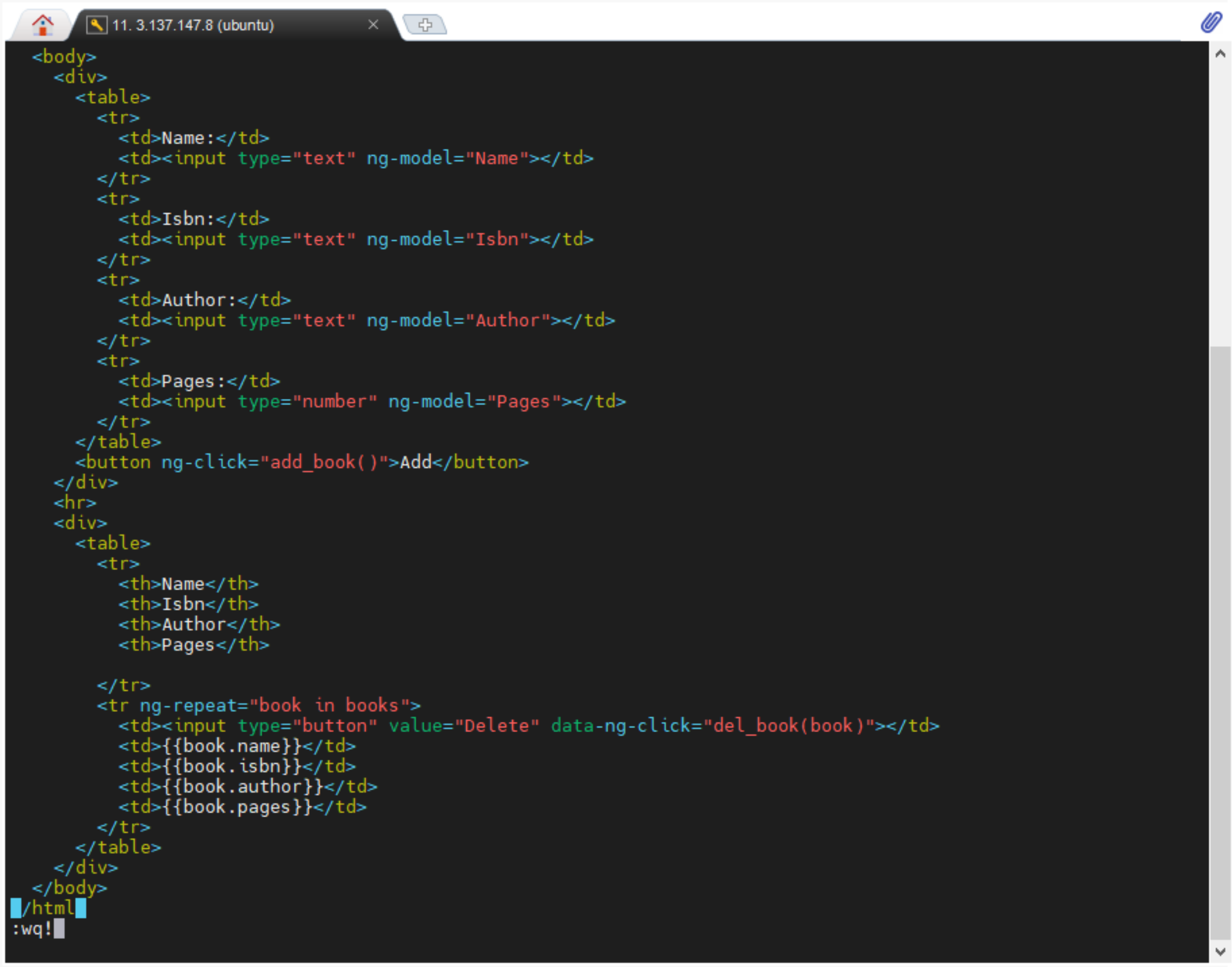
touch index.html



```
ubuntu@ip-172-31-46-50:~/Books/public$ touch index.html
ubuntu@ip-172-31-46-50:~/Books/public$ ls
index.html  script.js
ubuntu@ip-172-31-46-50:~/Books/public$ 
```

Copy and paste the required code into index.html file.

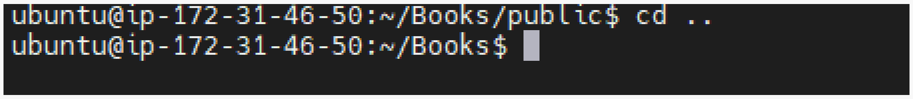
vim index.html



```
  <body>
    <div>
      <table>
        <tr>
          <td>Name:</td>
          <td><input type="text" ng-model="Name"></td>
        </tr>
        <tr>
          <td>Isbn:</td>
          <td><input type="text" ng-model="Isbn"></td>
        </tr>
        <tr>
          <td>Author:</td>
          <td><input type="text" ng-model="Author"></td>
        </tr>
        <tr>
          <td>Pages:</td>
          <td><input type="number" ng-model="Pages"></td>
        </tr>
      </table>
      <button ng-click="add_book()">Add</button>
    </div>
    <hr>
    <div>
      <table>
        <tr>
          <th>Name</th>
          <th>Isbn</th>
          <th>Author</th>
          <th>Pages</th>

        </tr>
        <tr ng-repeat="book in books">
          <td><input type="button" value="Delete" data-ng-click="del_book(book)"></td>
          <td>{{book.name}}</td>
          <td>{{book.isbn}}</td>
          <td>{{book.author}}</td>
          <td>{{book.pages}}</td>
        </tr>
      </table>
    </div>
  </body>
</html>
:wq!
```

Change the directory back up to Books

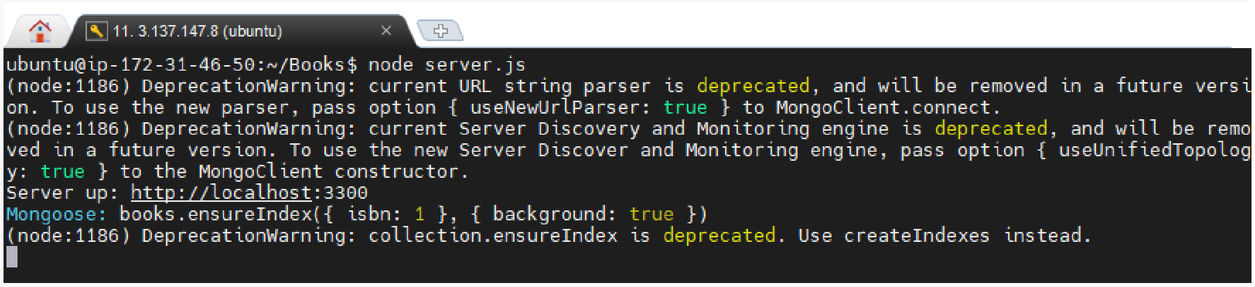
cd ..



```
ubuntu@ip-172-31-46-50:~/Books/public$ cd ..
ubuntu@ip-172-31-46-50:~/Books$
```

Start the server by running this command:

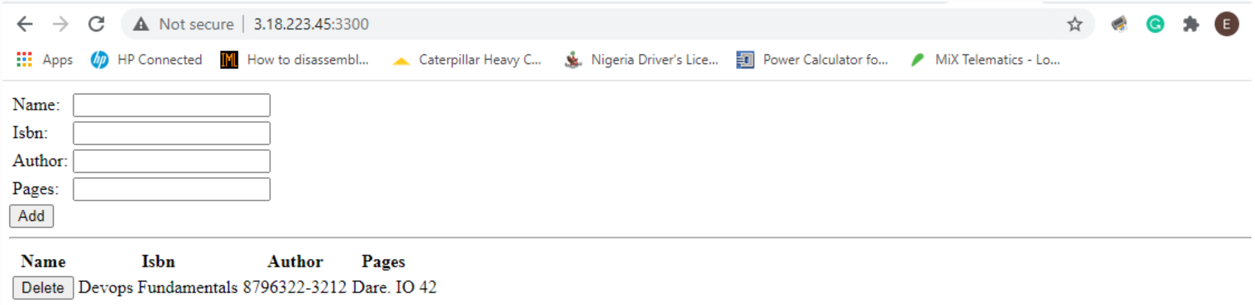
node server.js



```
ubuntu@ip-172-31-46-50:~/Books$ node server.js
(node:1186) DeprecationWarning: current URL string parser is deprecated, and will be removed in a future versi
on. To use the new parser, pass option { useNewUrlParser: true } to MongoClient.connect.
(node:1186) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be remo
ved in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopolog
y: true } to the MongoClient constructor.
Server up: http://localhost:3300
Mongoose: books.ensureIndex({ isbn: 1 }, { background: true })
(node:1186) DeprecationWarning: collection.ensureIndex is deprecated. Use createIndexes instead.
```

Open a web browser to the address ip:3300

http://3.18.223.45:3300/



Name:	
Isbn:	
Author:	
Pages:	

Add

	Name	Isbn	Author	Pages
Delete	Devops Fundamentals	8796322-3212	Dare. IO	42

In this project, I understood the concept behind hosting a web application for storing book details using MongoDB to hold the book records such as book name, isbn number, author, and number of pages. Node.js (JavaScript runtime environment) -which accept requests and display results to the end user was used to set up the Express routes and AngularJS controllers.