MEAN STACK PROJECT IMPLEMENTATION

The main aim for this project is to explain the DevOps concepts and processes using a MEAN web stack on a simple to-do application. Some developers use this set of framework and tools to develop software products. We would be carrying out this project in the AWS platform and these concepts are very similar to the LAMP,LEMP AND MERN web stack concept.

MEAN is an acronym of sets of technologies used to develop a technical software product.

MongoDB

Express

AngularJS

NodeJS

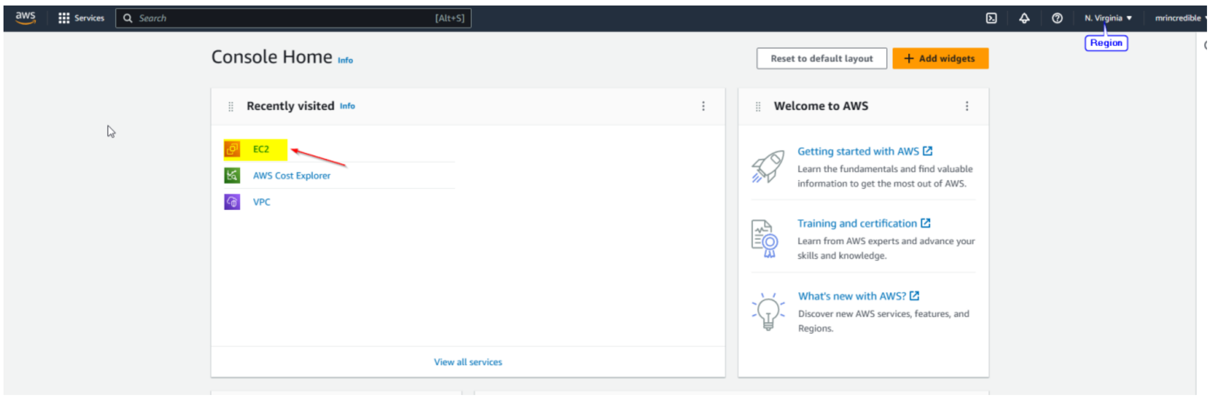
MongoDB is the document database that stores and allow it possible to retrieve data. Express JS is the back end application that makes a request to the Database for reads and write and gets response from the database . AngularJS is the front end application that handles Client and Server requests.Node.js is the JavaScript

Pre-requisite for the projects is the following.

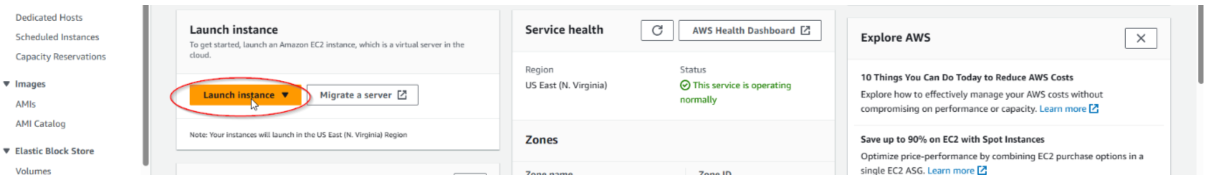
1) **Fundamental Knowledge of Installing and downloading software**
2) **Basic Understanding of Linux Commands**
3) **AWS account login with EC2 instance**
4) **Internet connection**

IMPLEMENTATION STEPS:

i) **Ensure you login with your details to your AWS console via the https://aws.amazon.com**
ii) **Click on the EC2 link to create instances.**


Services
Search
[Alt+S]
N. Virginia
Region
Console Home Info
Reset to default layout
+ Add widgets
Recently visited Info
EC2
AWS Cost Explorer
VPC
View all services
Welcome to AWS
Getting started with AWS
Learn the fundamentals and find valuable information to get the most out of AWS.
Training and certification
Learn from AWS experts and advance your skills and knowledge.
What's new with AWS?
Discover new AWS services, features, and Regions.

iii)Click on launch instance dropdown button and select launch instance .



Dedicated Hosts
Scheduled Instances
Capacity Reservations
▼ Images
AMIs
AMI Catalog
▼ Elastic Block Store
Volumes

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch instance ▼ | Migrate a server

Note: Your instances will launch in the US East (N. Virginia) Region

Service health

AWS Health Dashboard

Region: US East (N. Virginia)

Status: This service is operating normally

Zones

Zone name | Zone ID

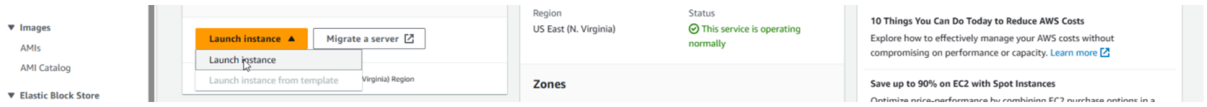
Explore AWS

10 Things You Can Do Today to Reduce AWS Costs

Explore how to effectively manage your AWS costs without compromising on performance or capacity. Learn more

Save up to 90% on EC2 with Spot Instances

Optimize price-performance by combining EC2 purchase options in a single EC2 ASG. Learn more



▼ Images
AMIs
AMI Catalog
▼ Elastic Block Store

Launch instance ▲ | Migrate a server

Launch instance
Launch instance from template

Virginia) Region

Region: US East (N. Virginia)

Status: This service is operating normally

Zones

10 Things You Can Do Today to Reduce AWS Costs

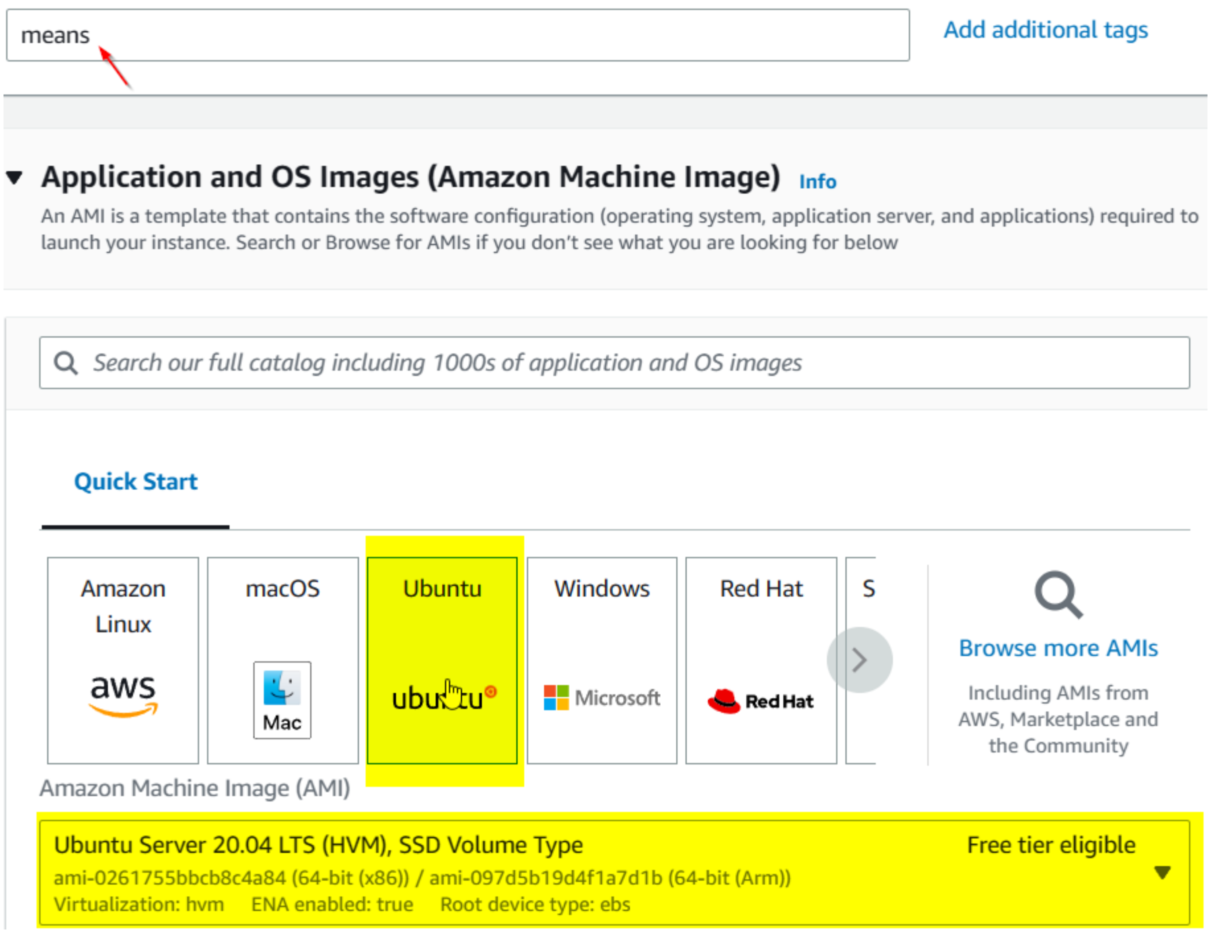
Explore how to effectively manage your AWS costs without compromising on performance or capacity. Learn more

Save up to 90% on EC2 with Spot Instances

iv)Fill in all relevant details to the MEAN project such as:

Type in the name and additional tag to the project (mean). Select ubuntu from the quick start option .Also note that the Amazon machine image selection varies from user to user

Select Ubuntu server 20.04 LTS (HVM),SSD Volume Type (Free Tier)



means | Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Amazon Machine Image (AMI)

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type — Free tier eligible

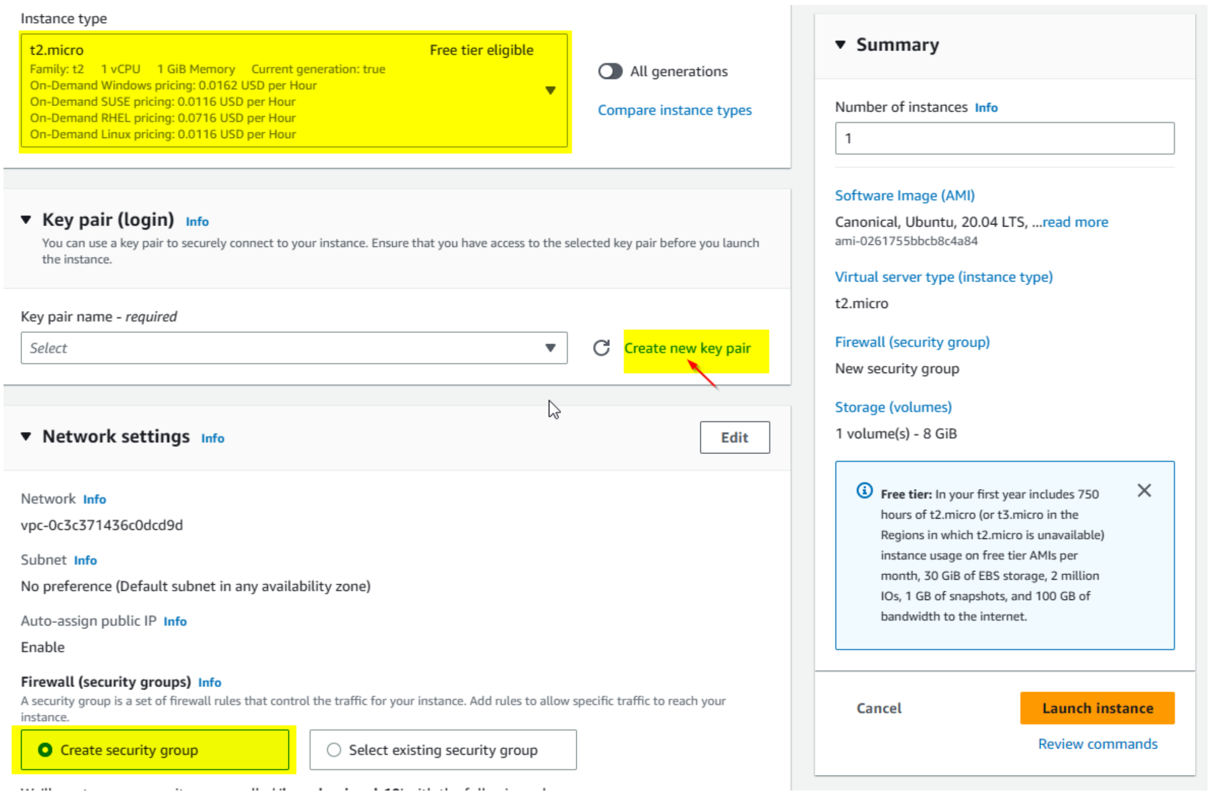
ami-0261755bbcb8c4a84 (64-bit (x86)) / ami-097d5b19d4f1a7d1b (64-bit (Arm))

Virtualization: hvm   ENA enabled: true   Root device type: ebs

v)The instance type selected in the configuration is the t2 micro -free tier.

Click on the “Create new key pair” link.

Ensure the Checkbox remains unchanged on the “Create security group”.



Instance type

t2.micro Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows pricing: 0.0162 USD per Hour
On-Demand SUSE pricing: 0.0116 USD per Hour
On-Demand RHEL pricing: 0.0716 USD per Hour
On-Demand Linux pricing: 0.0116 USD per Hour

All generations
Compare instance types

Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*
Select
Create new key pair

Network settings Info Edit

Network Info
vpc-0c3c371436c0dcd9d

Subnet Info
No preference (Default subnet in any availability zone)

Auto-assign public IP Info
Enable

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
[✓ Create security group] Select existing security group

Summary

Number of instances Info
1

Software Image (AMI)
Canonical, Ubuntu, 20.04 LTS, ...read more
ami-0261755bbcb8c4a84

Virtual server type (instance type)
t2.micro

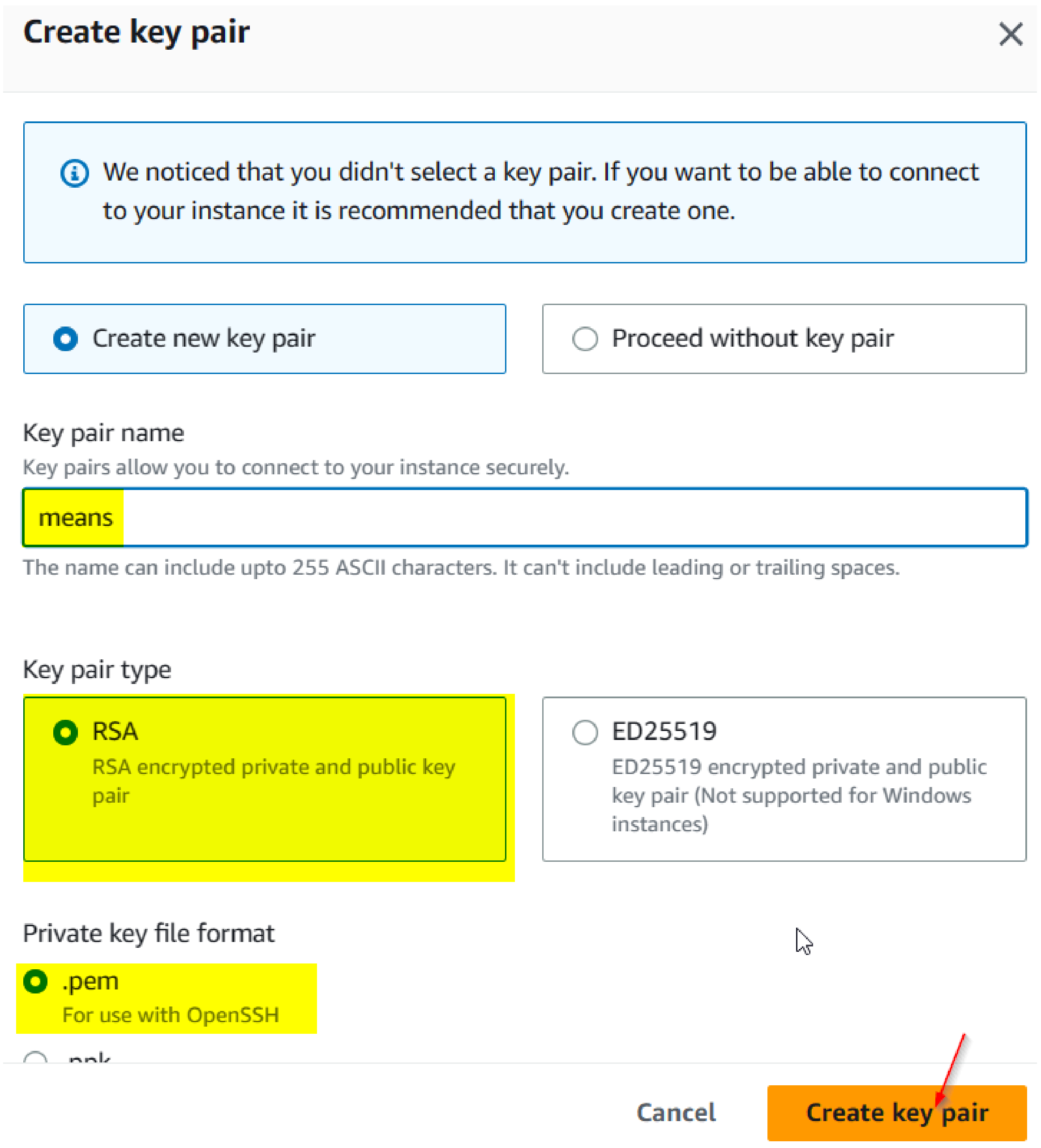
Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch instance
Review commands

vi)Type in the key pair name, chose the default key pair type and private key file format (rsa and .pem) and clicked the “Create key pair button”


Create key pair

We noticed that you didn't select a key pair. If you want to be able to connect to your instance it is recommended that you create one.

[✓ Create new key pair] Proceed without key pair

Key pair name

Key pairs allow you to connect to your instance securely.

means

The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

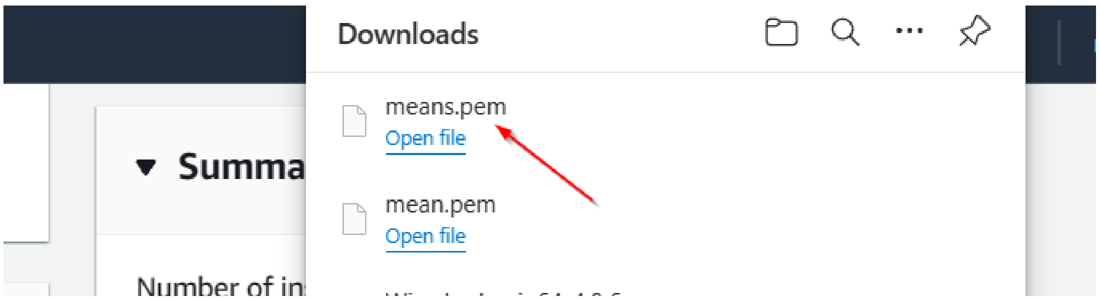
[✓ RSA] RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format

[✓ .pem] For use with OpenSSH

vii) The .pem file was downloaded successfully

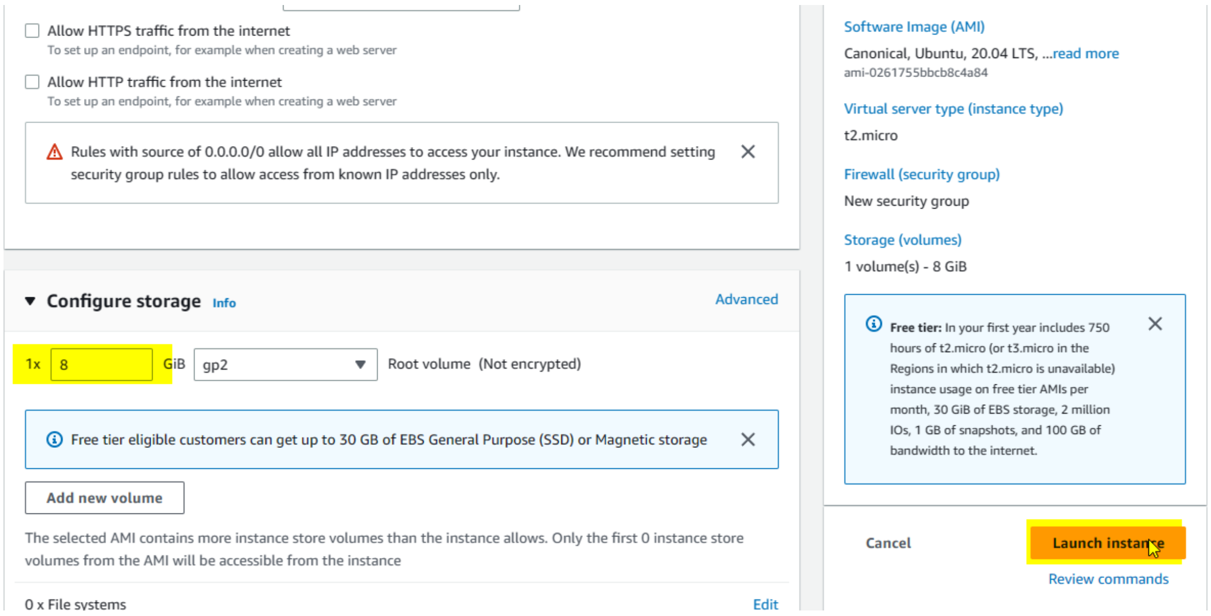

Downloads

means.pem

mean.pem

viii) I have deliberately chosen default settings to allow SSH traffic from anywhere as well as the storage volume given by AWS.

Then we proceed to launch our instance finally.



☐ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

▼ Configure storage Info Advanced

1x 8 GiB gp2 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

0 x File systems Edit

Software Image (AMI)
Canonical, Ubuntu, 20.04 LTS, ...read more
ami-0261755bbcb8c4a84

Virtual server type (instance type)
t2.micro

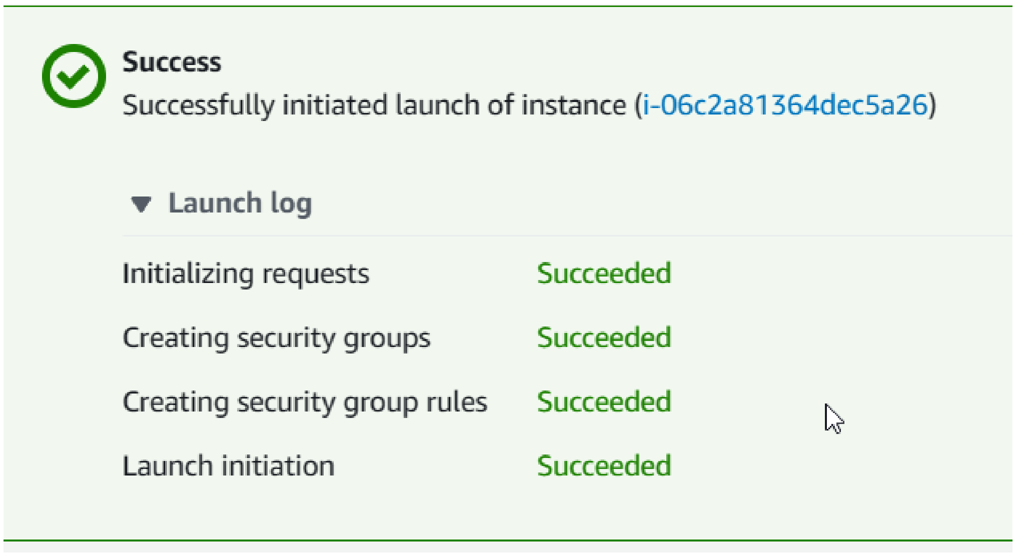
Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch instance
Review commands

Instance successfully launched and click to view Instance details with the IP address.

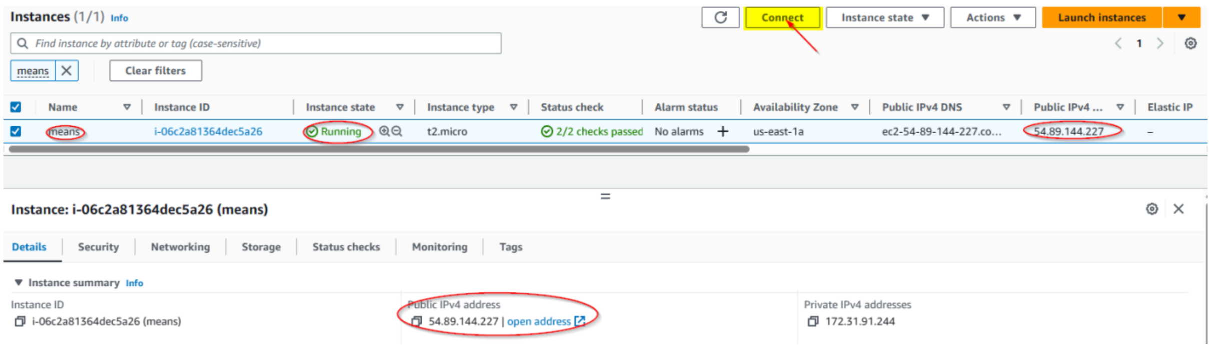


Success
Successfully initiated launch of instance (i-06c2a81364dec5a26)

▼ Launch log

Initializing requests	Succeeded
Creating security groups	Succeeded
Creating security group rules	Succeeded
Launch initiation	Succeeded

Click the “Connect” button and copy the ssh client details we would be using on the git bash console.



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
means	i-06c2a81364dec5a26	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-54-89-144-227.co...	54.89.144.227	–

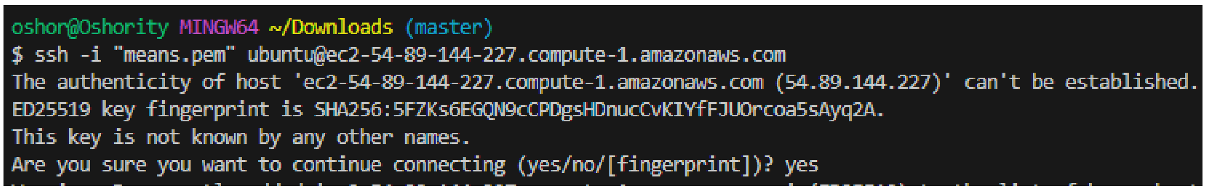
Instance: i-06c2a81364dec5a26 (means)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-06c2a81364dec5a26 (means)	54.89.144.227 | open address	172.31.91.244

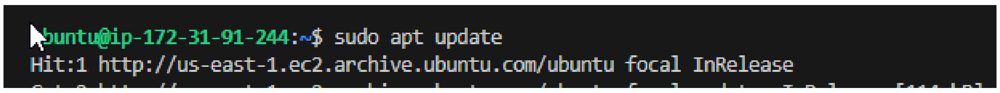
Open git bash on visual studio code or whichever console is convenient to use. We are using git bash here with Visual Studio Code and Type YES to connect.



```
oshor@Oshority MINGW64 ~/Downloads (master)
$ ssh -i "means.pem" ubuntu@ec2-54-89-144-227.compute-1.amazonaws.com
The authenticity of host 'ec2-54-89-144-227.compute-1.amazonaws.com (54.89.144.227)' can't be established.
ED25519 key fingerprint is SHA256:5FZKs6EGQN9cCPDgsHDnucCvKIYfFJUOrcoa5sAyq2A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

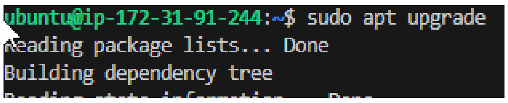
You have successful connected to the EC2 instance launched on AWS via ssh

Type clear to have a clear console and proceed to updating the lists of packages in the package manager.

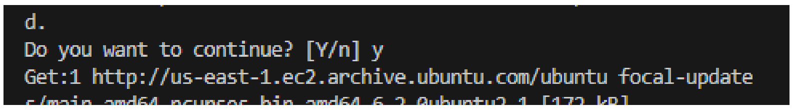


```
ubuntu@ip-172-31-91-244:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
```

Then we proceed to upgrade the packages and Type YES to continue.

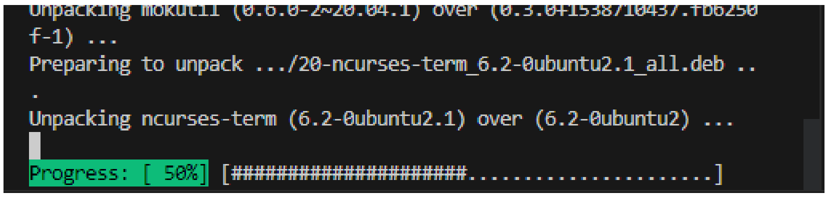


```
ubuntu@ip-172-31-91-244:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree
```

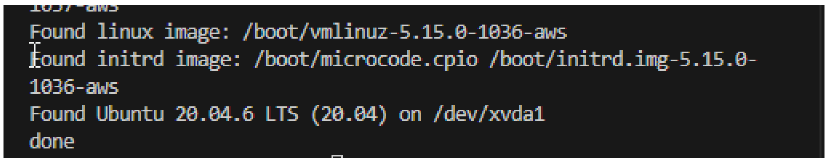


```
d.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-update
```

Still upgrading and finally completed as shown below

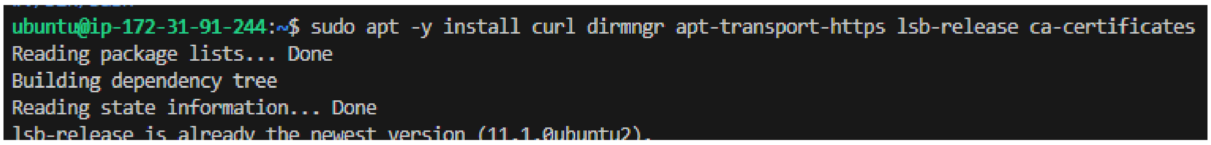


```
f-1) ...
Preparing to unpack .../20-ncurses-term_6.2-0ubuntu2.1_all.deb ..
.
Unpacking ncurses-term (6.2-0ubuntu2.1) over (6.2-0ubuntu2) ...

Progress: [ 50%] [######################.....................]
```

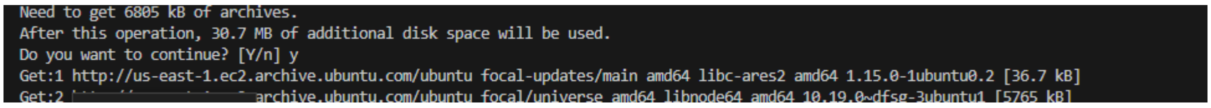


```
Found linux image: /boot/vmlinuz-5.15.0-1036-aws
Found initrd image: /boot/microcode.cpio /boot/initrd.img-5.15.0-
1036-aws
Found Ubuntu 20.04.6 LTS (20.04) on /dev/xvda1
done
```

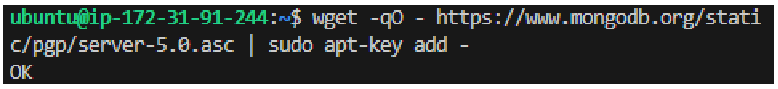
Now we add the certificates. Type YES to continue and check the node version



```
ubuntu@ip-172-31-91-244:~$ sudo apt -y install curl dirmngr apt-transport-https lsb-release ca-certificates
Reading package lists... Done
Building dependency tree
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu2).
```



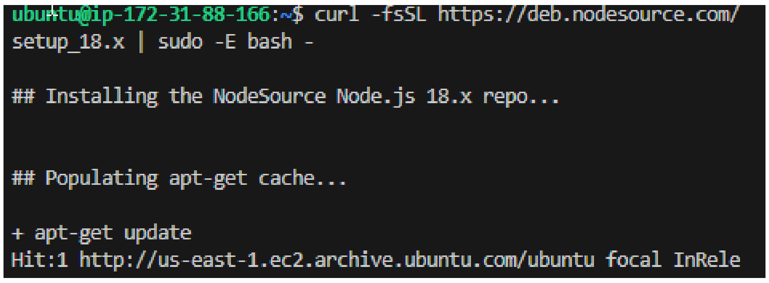
```
Need to get 6805 kB of archives.
After this operation, 30.7 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 libc-ares2 amd64 1.15.0-1ubuntu0.2 [36.7 kB]
Get:2 [illegible]archive.ubuntu.com/ubuntu focal/universe amd64 libnode64 amd64 10.19.0~dfsg-3ubuntu1 [5765 kB]
```



```
ubuntu@ip-172-31-91-244:~$ wget -qO - https://www.mongodb.org/stati
c/pgp/server-5.0.asc | sudo apt-key add -
OK
```

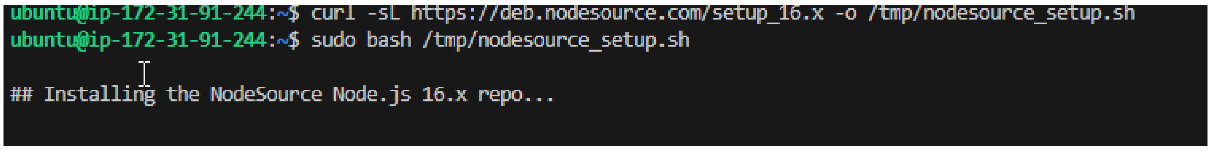
NODE.JS INSTALLATION

Now we need to get the location of Node.js using a node source PPA

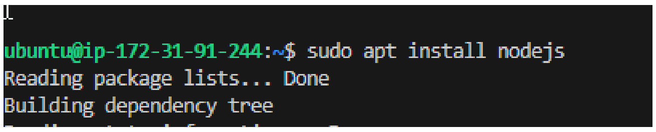


```
ubuntu@ip-172-31-88-166:~$ curl -fsSL https://deb.nodesource.com/
setup_18.x | sudo -E bash -

## Installing the NodeSource Node.js 18.x repo...


## Populating apt-get cache...

+ apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRele
```

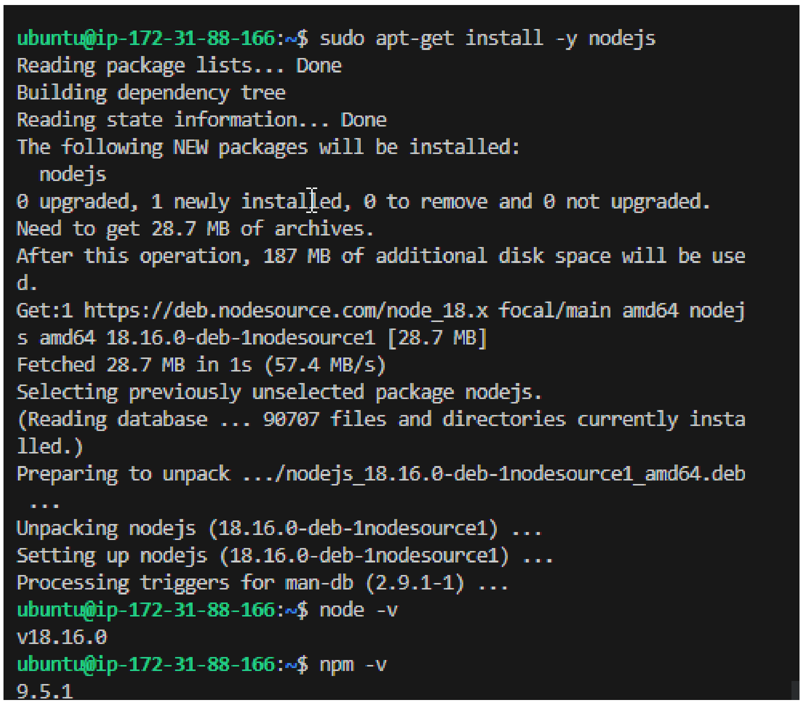
We can now install Node.js on the server and confirm the versions of the node and npm package managers as shown below.



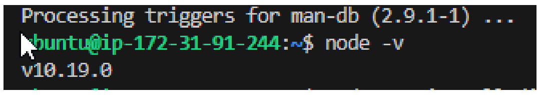
```
ubuntu@ip-172-31-91-244:~$ curl -sL https://deb.nodesource.com/setup_16.x -o /tmp/nodesource_setup.sh
ubuntu@ip-172-31-91-244:~$ sudo bash /tmp/nodesource_setup.sh

## Installing the NodeSource Node.js 16.x repo...
```



```
ubuntu@ip-172-31-91-244:~$ sudo apt install nodejs
Reading package lists... Done
Building dependency tree
```



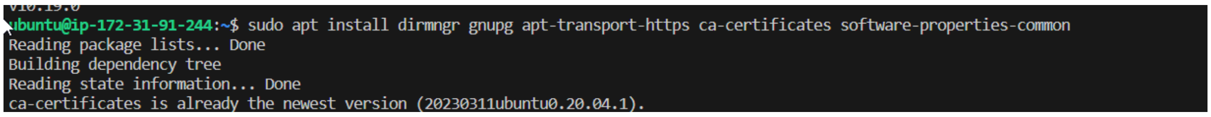
```
ubuntu@ip-172-31-88-166:~$ sudo apt-get install -y nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  nodejs
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 28.7 MB of archives.
After this operation, 187 MB of additional disk space will be use
d.
Get:1 https://deb.nodesource.com/node_18.x focal/main amd64 nodej
s amd64 18.16.0-deb-1nodesource1 [28.7 MB]
Fetched 28.7 MB in 1s (57.4 MB/s)
Selecting previously unselected package nodejs.
(Reading database ... 90707 files and directories currently insta
lled.)
Preparing to unpack .../nodejs_18.16.0-deb-1nodesource1_amd64.deb
 ...
Unpacking nodejs (18.16.0-deb-1nodesource1) ...
Setting up nodejs (18.16.0-deb-1nodesource1) ...
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-88-166:~$ node -v
v18.16.0
ubuntu@ip-172-31-88-166:~$ npm -v
9.5.1
```



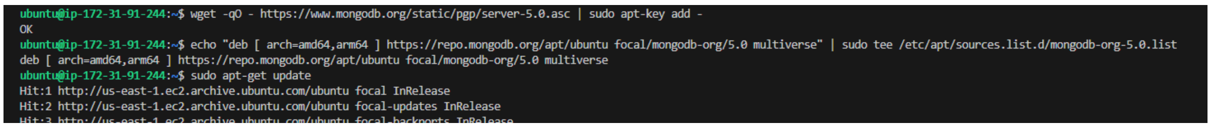
```
Processing triggers for man-db (2.9.1-1) ...
ubuntu@ip-172-31-91-244:~$ node -v
v10.19.0
```

INSTALLATION OF MONGODB

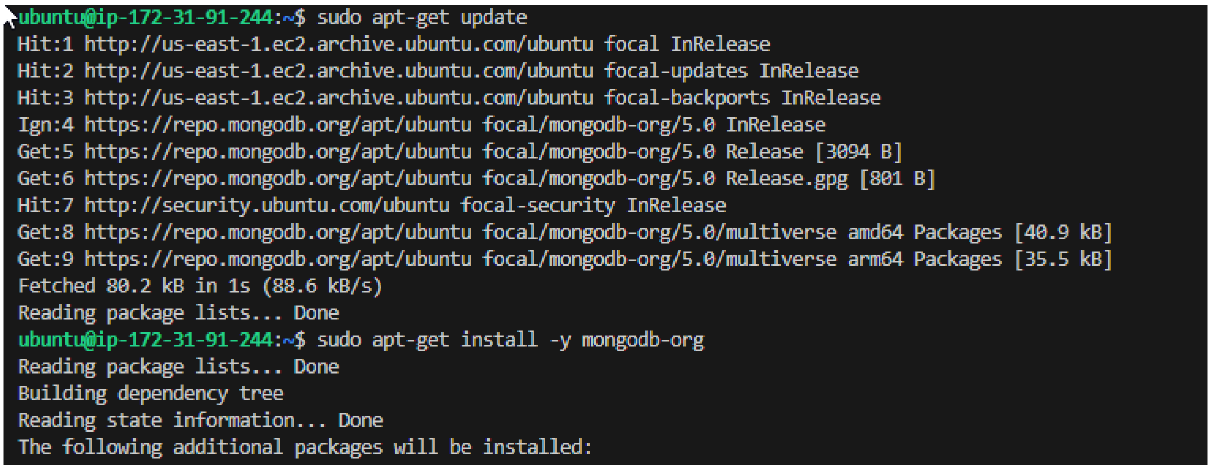
MongoDB stores data in flexible JSON format documents and can vary from document to data structure which can be changed over time .Here we would be adding a book records to the database containing the book name ,ISBN number ,Author and number of pages .We run the following command as shown below to download MongoDB



```
ubuntu@ip-172-31-91-244:~$ sudo apt install dirmngr gnupg apt-transport-https ca-certificates software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
ca-certificates is already the newest version (20230311ubuntu0.20.04.1).
```

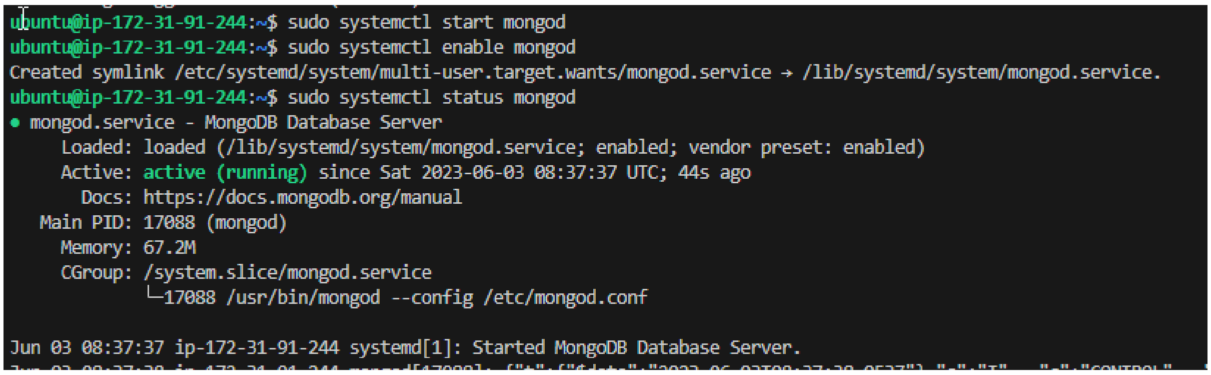


```
ubuntu@ip-172-31-91-244:~$ wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | sudo apt-key add -
OK
ubuntu@ip-172-31-91-244:~$ echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-5.0.list
deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 multiverse
ubuntu@ip-172-31-91-244:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
```



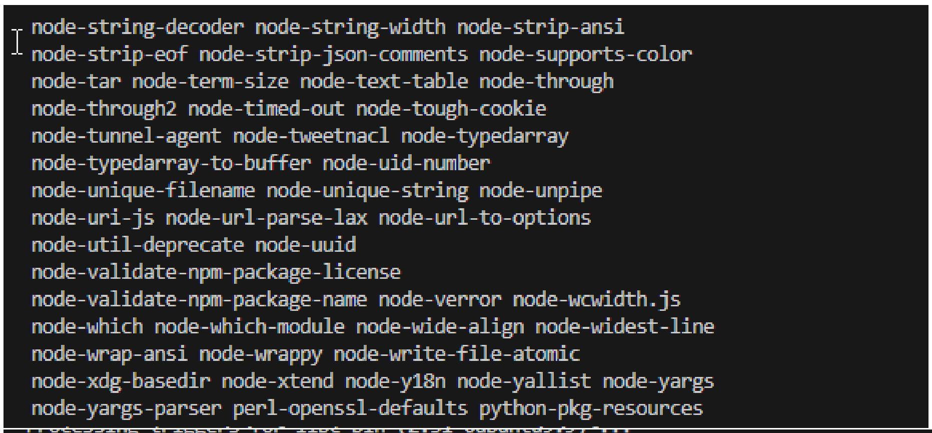
```
ubuntu@ip-172-31-91-244:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Ign:4 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 InRelease
Get:5 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 Release [3094 B]
Get:6 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 Release.gpg [801 B]
Hit:7 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:8 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0/multiverse amd64 Packages [40.9 kB]
Get:9 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0/multiverse arm64 Packages [35.5 kB]
Fetched 80.2 kB in 1s (88.6 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-91-244:~$ sudo apt-get install -y mongodb-org
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
```

Once installed we have to start the server and verify that it is running as expected as shown below . MongoDB is active and running

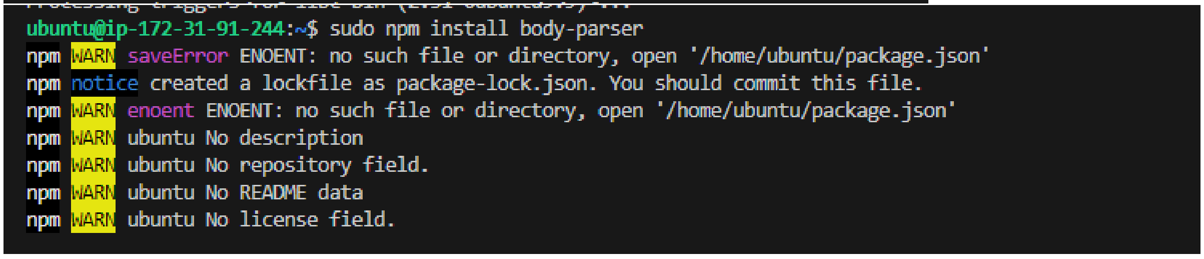


```
ubuntu@ip-172-31-91-244:~$ sudo systemctl start mongod
ubuntu@ip-172-31-91-244:~$ sudo systemctl enable mongod
Created symlink /etc/systemd/system/multi-user.target.wants/mongod.service → /lib/systemd/system/mongod.service.
ubuntu@ip-172-31-91-244:~$ sudo systemctl status mongod
● mongod.service - MongoDB Database Server
     Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-06-03 08:37:37 UTC; 44s ago
       Docs: https://docs.mongodb.org/manual
   Main PID: 17088 (mongod)
     Memory: 67.2M
     CGroup: /system.slice/mongod.service
             └─17088 /usr/bin/mongod --config /etc/mongod.conf

Jun 03 08:37:37 ip-172-31-91-244 systemd[1]: Started MongoDB Database Server.
```

Next we install node package manager and body-parser package

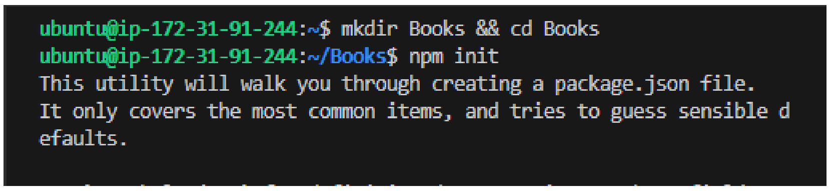


```
  node-string-decoder node-string-width node-strip-ansi
  node-strip-eof node-strip-json-comments node-supports-color
  node-tar node-term-size node-text-table node-through
  node-through2 node-timed-out node-tough-cookie
  node-tunnel-agent node-tweetnacl node-typedarray
  node-typedarray-to-buffer node-uid-number
  node-unique-filename node-unique-string node-unpipe
  node-uri-js node-url-parse-lax node-url-to-options
  node-util-deprecate node-uuid
  node-validate-npm-package-license
  node-validate-npm-package-name node-verror node-wcwidth.js
  node-which node-which-module node-wide-align node-widest-line
  node-wrap-ansi node-wrappy node-write-file-atomic
  node-xdg-basedir node-xtend node-y18n node-yallist node-yargs
  node-yargs-parser perl-openssl-defaults python-pkg-resources
```



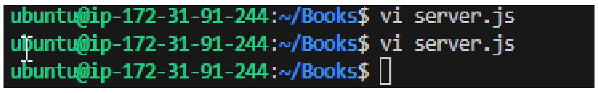
```
ubuntu@ip-172-31-91-244:~$ sudo npm install body-parser
npm WARN saveError ENOENT: no such file or directory, open '/home/ubuntu/package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN enoent ENOENT: no such file or directory, open '/home/ubuntu/package.json'
npm WARN ubuntu No description
npm WARN ubuntu No repository field.
npm WARN ubuntu No README data
npm WARN ubuntu No license field.
```

And then proceed to create the folder named Books and change directory to the new Books folder.

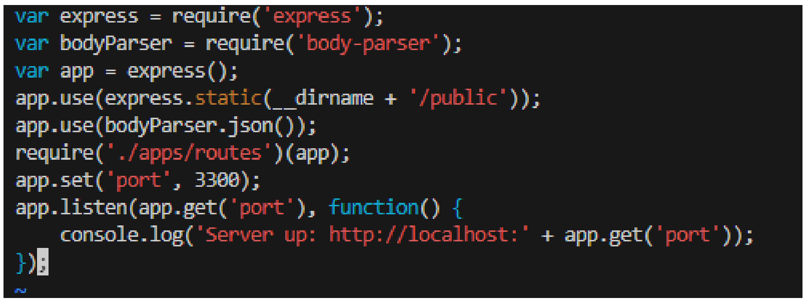


```
ubuntu@ip-172-31-91-244:~$ mkdir Books && cd Books
ubuntu@ip-172-31-91-244:~/Books$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible d
efaults.
```

In the Books directory ,we initialize the npm project and add a file into server.js



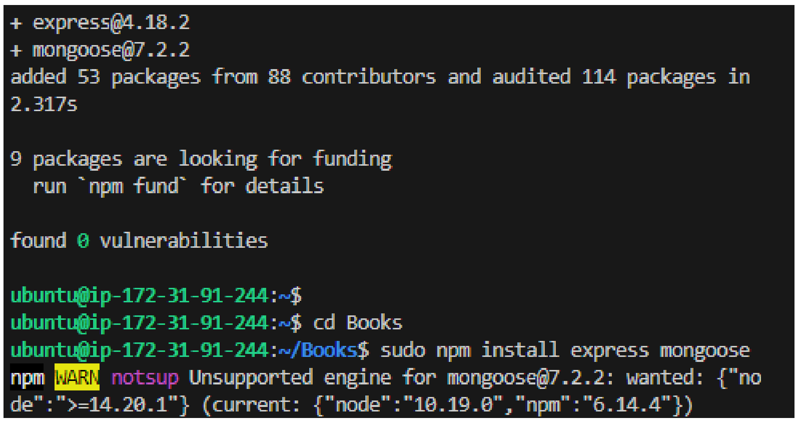
```
ubuntu@ip-172-31-91-244:~/Books$ vi server.js
ubuntu@ip-172-31-91-244:~/Books$ vi server.js
ubuntu@ip-172-31-91-244:~/Books$ 
```



```
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'));
});
~
```

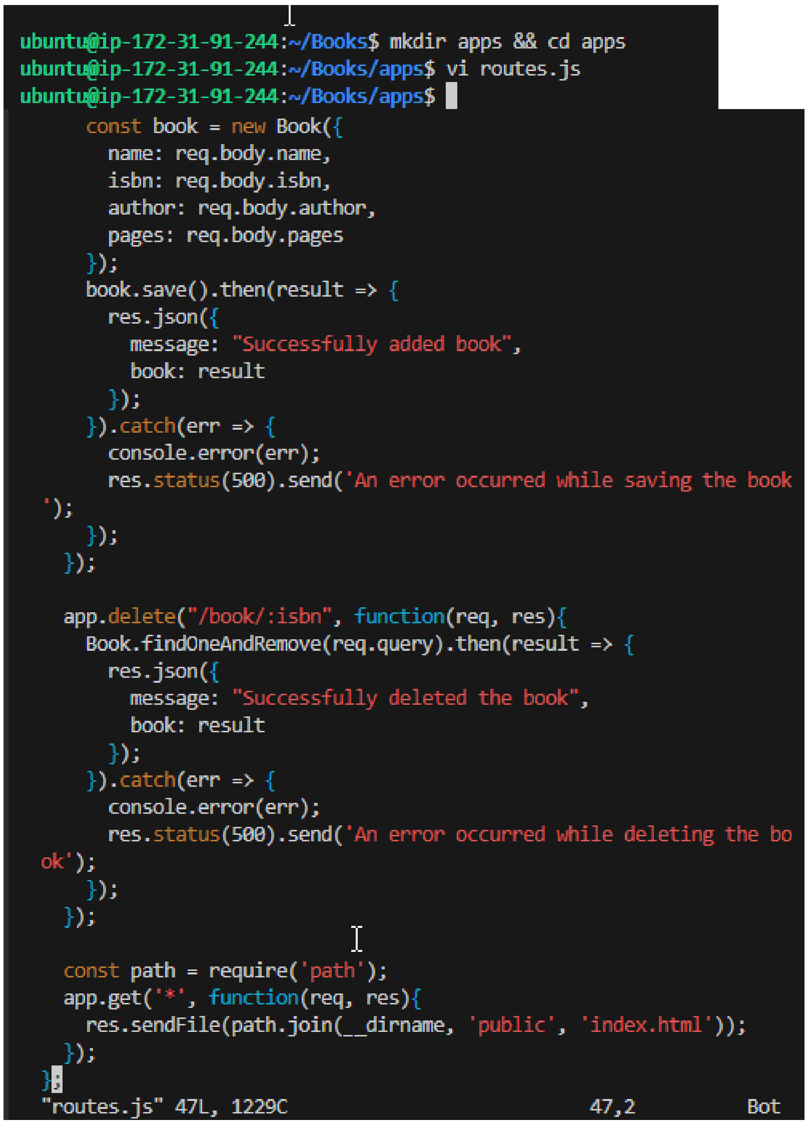
INSTALLATION EXPRESS AND SET UP ROUTES TO THE SERVER

Express is a minimal and flexible Node.js web application framework that provides features for web and mobile application .We would be installing a Mongoose package which provides straight forward ,schema-based solution to model application data



```
+ express@4.18.2
+ mongoose@7.2.2
added 53 packages from 88 contributors and audited 114 packages in
2.317s

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

found 0 vulnerabilities

ubuntu@ip-172-31-91-244:~$
ubuntu@ip-172-31-91-244:~$ cd Books
ubuntu@ip-172-31-91-244:~/Books$ sudo npm install express mongoose
npm WARN notsup Unsupported engine for mongoose@7.2.2: wanted: {"no
de":">=14.20.1"} (current: {"node":"10.19.0","npm":"6.14.4"})
```

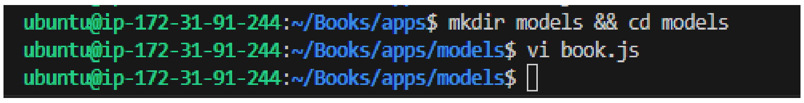
We create the folder named apps and change directory to the new apps folder. Edit the route.js file and input the codes below



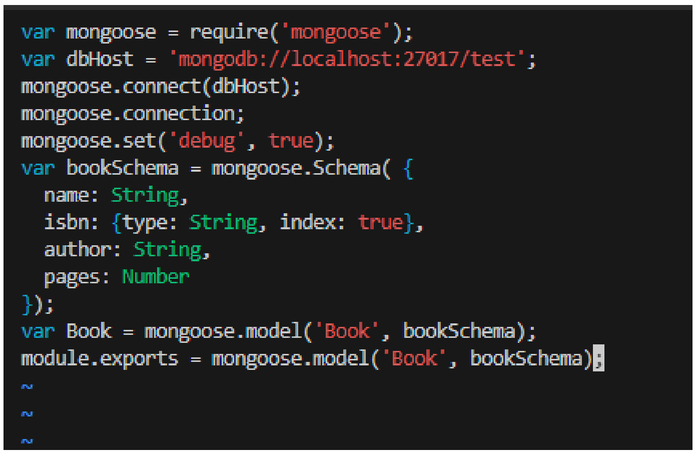
```
ubuntu@ip-172-31-91-244:~/Books$ mkdir apps && cd apps
ubuntu@ip-172-31-91-244:~/Books/apps$ vi routes.js
ubuntu@ip-172-31-91-244:~/Books/apps$
```

```
    const book = new Book({
      name: req.body.name,
      isbn: req.body.isbn,
      author: req.body.author,
      pages: req.body.pages
    });
    book.save().then(result => {
      res.json({
        message: "Successfully added book",
        book: result
      });
    }).catch(err => {
      console.error(err);
      res.status(500).send('An error occurred while saving the book
');
    });
  });

  app.delete("/book/:isbn", function(req, res){
    Book.findOneAndRemove(req.query).then(result => {
      res.json({
        message: "Successfully deleted the book",
        book: result
      });
    }).catch(err => {
      console.error(err);
      res.status(500).send('An error occurred while deleting the bo
ok');
    });
  });

  const path = require('path');
  app.get('*', function(req, res){
    res.sendFile(path.join(__dirname, 'public', 'index.html'));
  });
};
"routes.js" 47L, 1229C                              47,2          Bot
```

We create the folder named models and change directory to the new models folder. Edit the book.js file and input the codes below.



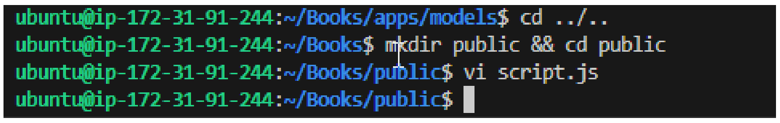
```
ubuntu@ip-172-31-91-244:~/Books/apps$ mkdir models && cd models
ubuntu@ip-172-31-91-244:~/Books/apps/models$ vi book.js
ubuntu@ip-172-31-91-244:~/Books/apps/models$
```



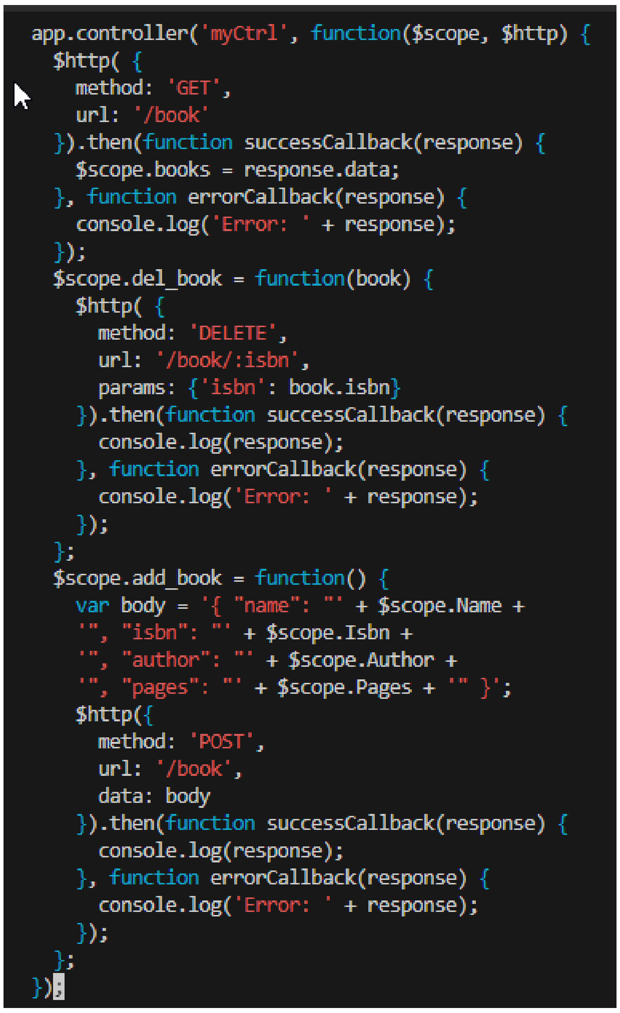
```
var mongoose = require('mongoose');
var dbHost = 'mongodb://localhost:27017/test';
mongoose.connect(dbHost);
mongoose.connection;
mongoose.set('debug', true);
var bookSchema = mongoose.Schema( {
  name: String,
  isbn: {type: String, index: true},
  author: String,
  pages: Number
});
var Book =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

ACCESSING THE ROUTES WITH ANGULARJS

AngularJS provides a web framework for creating dynamic views in your web applications. We would be changing the directory back to Books and then we create a new folder named public and change directory to the new folder. Edit the script.js file and input the codes below.

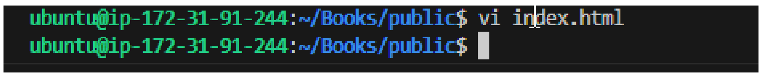


```
ubuntu@ip-172-31-91-244:~/Books/apps/models$ cd ../..
ubuntu@ip-172-31-91-244:~/Books$ mkdir public && cd public
ubuntu@ip-172-31-91-244:~/Books/public$ vi script.js
ubuntu@ip-172-31-91-244:~/Books/public$
```

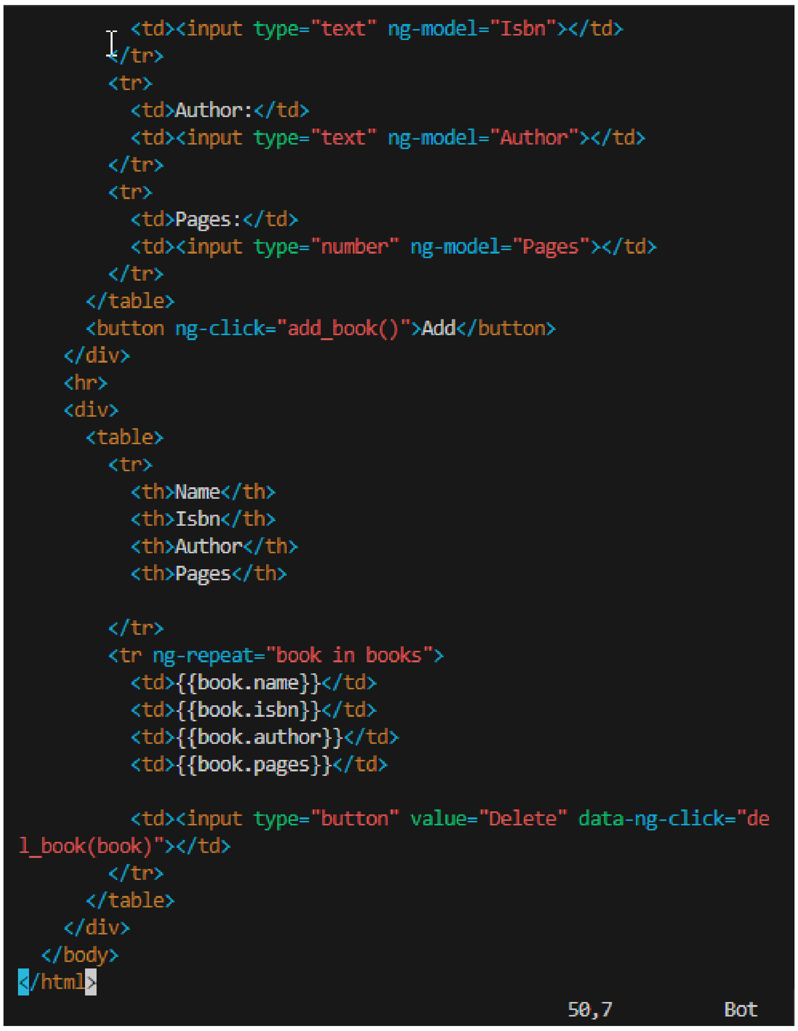


```
app.controller('myCtrl', function($scope, $http) {
  $http( {
    method: 'GET',
    url: '/book'
  }).then(function successCallback(response) {
    $scope.books = response.data;
  }, function errorCallback(response) {
    console.log('Error: ' + response);
  });
  $scope.del_book = function(book) {
    $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);
    });
  };
});
```

We would creating a new folder named index.html and edit the file and input the codes below.

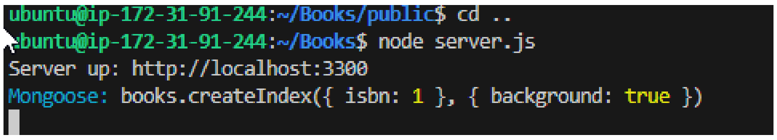


```
ubuntu@ip-172-31-91-244:~/Books/public$ vi index.html
ubuntu@ip-172-31-91-244:~/Books/public$
```



```
          <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>{{book.name}}</td>
          <td>{{book.isbn}}</td>
          <td>{{book.author}}</td>
          <td>{{book.pages}}</td>

          <td><input type="button" value="Delete" data-ng-click="de
l_book(book)"></td>
        </tr>
      </table>
    </div>
  </body>
</html>
                                                      50,7          Bot
```

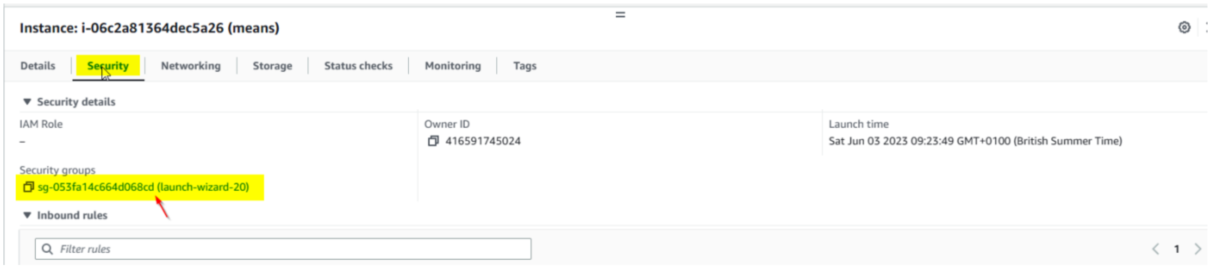
We would then navigate back to Books and start the server by running a command.



```
ubuntu@ip-172-31-91-244:~/Books/public$ cd ..
ubuntu@ip-172-31-91-244:~/Books$ node server.js
Server up: http://localhost:3300
Mongoose: books.createIndex({ isbn: 1 }, { background: true })
```

We see the server is up at port:3300

For this to happen we need to open TCP port 3300 in our AWS Web Console accessing the security group through the EC2 instance



Instance: i-06c2a81364dec5a26 (means)

▼ Security details

IAM Role
-

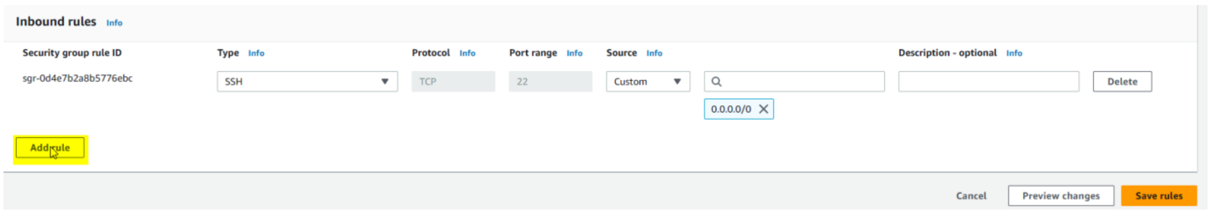
Owner ID
416591745024

Launch time
Sat Jun 03 2023 09:23:49 GMT+0100 (British Summer Time)

Security groups
sg-053fa14c664d068cd (launch-wizard-20)

▼ Inbound rules

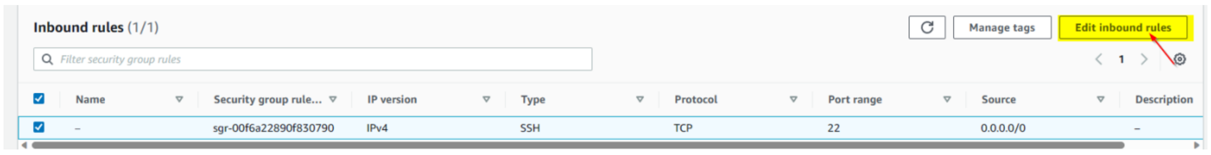
Add a new rule.


Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0d4e7b2a8b5776ebc	SSH	TCP	22	Custom	0.0.0.0/0		Delete

Add rule

Cancel | Preview changes | Save rules

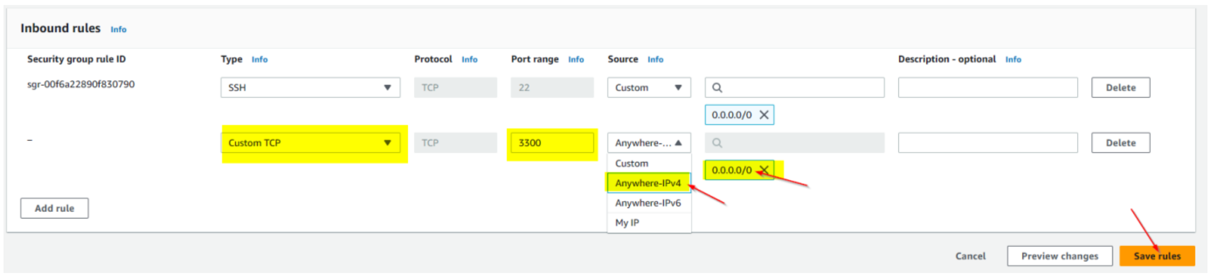

Inbound rules (1/1)

Manage tags | Edit inbound rules

Filter security group rules

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
–	sgr-00f6a22890f830790	IPv4	SSH	TCP	22	0.0.0.0/0	–

Type in the port range 3300 and click “Anywhere ipv4”

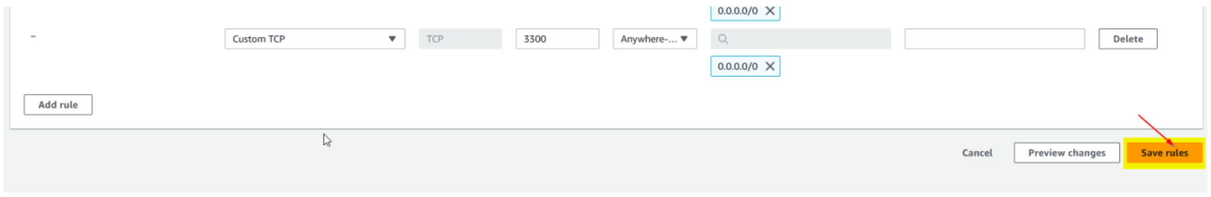

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-00f6a22890f830790	SSH	TCP	22	Custom	0.0.0.0/0		Delete
–	Custom TCP	TCP	3300	Anywhere-...	0.0.0.0/0		Delete

Add rule

Cancel | Preview changes | Save rules

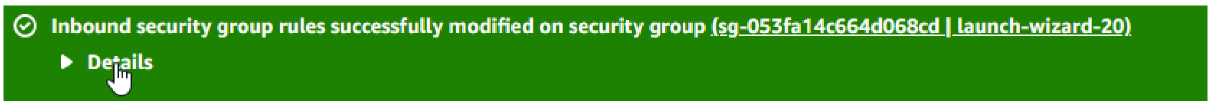
Click the “Save rules” Button


					0.0.0.0/0		
–	Custom TCP	TCP	3300	Anywhere-...	0.0.0.0/0		Delete

Add rule

Cancel | Preview changes | Save rules

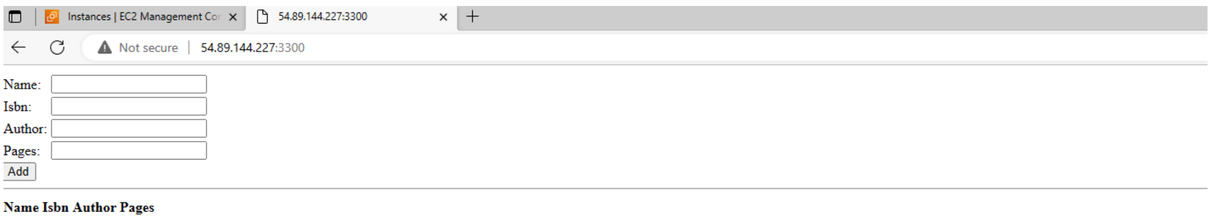
Inbound rule successfully modified


Inbound security group rules successfully modified on security group (sg-053fa14c664d068cd | launch-wizard-20)

Details

Open any browser of your choice and access the URL

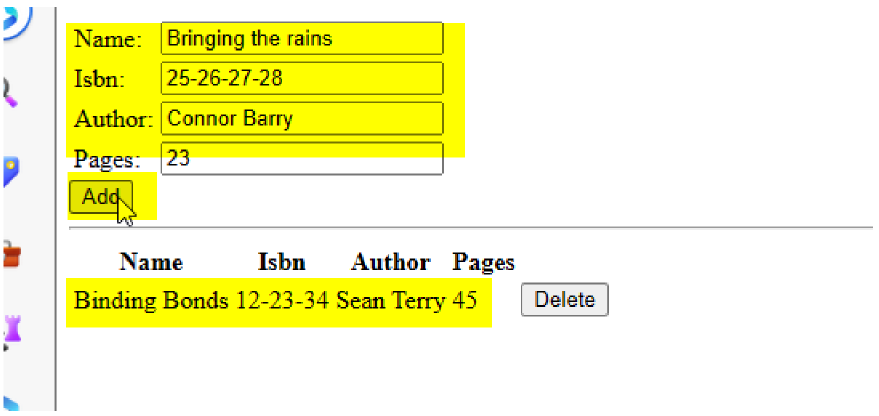
http://54.89.144.227:3300


Instances | EC2 Management Co
54.89.144.227:3300
Not secure | 54.89.144.227:3300

Name:
Isbn:
Author:
Pages:
Add

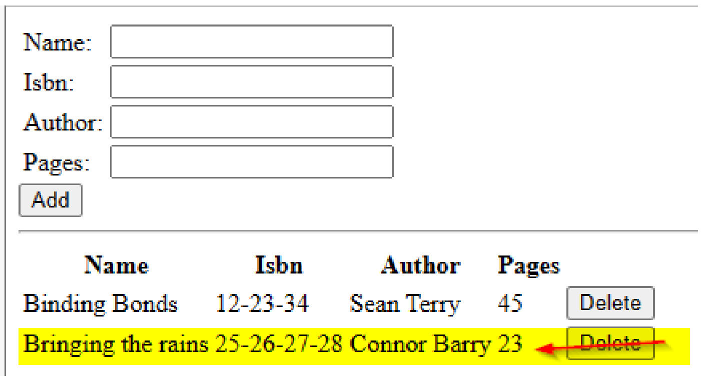
Name Isbn Author Pages

Book web page successfully displayed.We have to test the web page by inputting data and get it to be populated below


Name: Bringing the rains
Isbn: 25-26-27-28
Author: Connor Barry
Pages: 23
Add

Name	Isbn	Author	Pages	
Binding Bonds	12-23-34	Sean Terry	45	Delete

Refresh the page and see the data below


Name:
Isbn:
Author:
Pages:
Add

Name	Isbn	Author	Pages	
Binding Bonds	12-23-34	Sean Terry	45	Delete
Bringing the rains	25-26-27-28	Connor Barry	23	Delete

Congratulations we have successfully launched our Web Book Register