MEAN Stack is a combination of the following components:

MongoDB (Document database) – Stores and allows to retrieve data.

Express (Back-end application framework) – Makes requests to Database for Reads and Writes.

Angular (Front-end application framework) – Handles Client and Server Requests

Node.js (JavaScript runtime environment) – Accepts requests and displays results to end user

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

Update ubuntu
sudo apt update

Upgrade ubuntu
sudo apt upgrade

Add certificates
sudo apt -y install curl dirmngr apt-transport-https lsb-release ca-certificates
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -

Install NodeJS
sudo apt install -y nodejs

```
ubuntu@ip-172-31-47-198:~$ sudo apt 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 used.
Get:1 https://deb.nodesource.com/node_18.x focal/main amd64 nodejs amd64 18.16.0-deb-1nodesource1 [28.7 MB]
Fetched 28.7 MB in 0s (58.8 MB/s)
Selecting previously unselected package nodejs.
(Reading database ... 90711 files and directories currently installed.)
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-172-31-47-198:~$
```

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.

sudo apt-get install gnupg curl

Curl -fsSL https://pgp.mongodb.com/server-6.0.asc | sudo gpg -/usr/share/keyrings/mongodb-server-6.0.gpg –dearmor

Create a List file For MongoDB
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

```
ubuntu@ip-172-31-26-175:~$ echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-6.0.gpg ]
 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 multiverse" | sudo tee /etc/apt/sources.list.d/mong
odb-org-6.0.list
deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-6.0.gpg ] https://repo.mongodb.org/apt/ubu
ntu jammy/mongodb-org/6.0 multiverse
```

Install MongoDB
sudo apt install -y mongodb

```
ubuntu@ip-172-31-26-175:~$ sudo apt-get update && sudo apt-get install -y mongodb-org
Hit:1 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Ign:4 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 InRelease
Hit:5 https://deb.nodesource.com/node_18.x jammy InRelease
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:7 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 Release [3094 B]
Get:8 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 Release.gpg [866 B]
Get:9 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [894 kB]
Ign:8 https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 Release.gpg
Reading package lists... Done
W: GPG error: https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 Release: The following signatures coul
dn't be verified because the public key is not available: NO_PUBKEY 6A26B1AE64C3C388
E: The repository 'https://repo.mongodb.org/apt/ubuntu jammy/mongodb-org/6.0 Release' is not signed.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
ubuntu@ip-172-31-26-175:~$
```

Start The server
sudo service mongodb start

Verify that the service is up and running
sudo systemctl status mongodb

```
ubuntu@ip-172-31-47-198:~$ sudo systemctl status mongodb
● mongodb.service - An object/document-oriented database
     Loaded: loaded (/lib/systemd/system/mongodb.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-06-14 16:18:52 UTC; 1min 30s ago
       Docs: man:mongod(1)
   Main PID: 31651 (mongod)
      Tasks: 23 (limit: 1141)
     Memory: 42.7M
     CGroup: /system.slice/mongodb.service
             └─31651 /usr/bin/mongod --unixSocketPrefix=/run/mongodb --config /etc/mongodb.conf

Jun 14 16:18:52 ip-172-31-47-198 systemd[1]: Started An object/document-oriented database.
ubuntu@ip-172-31-47-198:~$
```

Install body-parser package
We need 'body-parser' package to help us process JSON files passed in requests to the server.

sudo npm install body-parser

```
ubuntu@ip-172-31-26-175:~$ sudo npm install body-parser

added 31 packages in 2s

7 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New minor version of npm available! 9.6.7 -> 9.8.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.8.1
npm notice Run npm install -g npm@9.8.1 to update!
npm notice
ubuntu@ip-172-31-26-175:~$
```

Create a folder named 'Books'
mkdir Books && cd Books

In the Books directory, Initialize npm project
npm init

```
ubuntu@ip-172-31-26-175:~/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 defa
ults.

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: (books) booksproject
version: (1.0.0)
description: A simple book register app
entry point: (index.js) server.js
test command:
git repository:
keywords:
author: shola.io
license: (ISC)
About to write to /home/ubuntu/Books/package.json:

{
  "name": "booksproject",
  "version": "1.0.0",
  "description": "A simple book register app",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "shola.io",
  "license": "ISC"
```

Add a file to it named server.js, to insert the server code.
vi server.js

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

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. We will use Express in to passbook information to and from our MongoDB database.

Installing Express
sudo npm install express mongoose

```
ubuntu@ip-172-31-26-175:~/Books$ sudo npm install express mongoose

added 82 packages, and audited 83 packages in 6s

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

found 0 vulnerabilities
ubuntu@ip-172-31-26-175:~/Books$
```

In 'Books' folder, create a folder named apps

mkdir apps && cd apps

Create a file named routes.js
vi routes.js

Copy and paste the code below into routes.js

```
const Book = require('./models/book');

module.exports = function(app) {

  app.get('/book', function(req, res) {

    Book.find({}).then(result => {

      res.json(result);
```

```
  }).catch(err => {
    console.error(err);
    res.status(500).send('An error occurred while retrieving books');
  });
});
app.post('/book', function(req, res) {
  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 book');
```

```
  });
 });

 const path = require('path');
 app.get('*', function(req, res) {
  res.sendFile(path.join(__dirname, 'public', 'index.html'));
 });
};
```

In the ‘apps’ folder, create a folder named models

mkdir models && cd models

Create a file named book.js

vi book.js

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);
```

```
found 0 vulnerabilities
ubuntu@ip-172-31-26-175:~/Books$ mkdir apps && cd apps
ubuntu@ip-172-31-26-175:~/Books/apps$ vi routes.js
ubuntu@ip-172-31-26-175:~/Books/apps$ mkdir models && cd models
ubuntu@ip-172-31-26-175:~/Books/apps/models$ vi book.js
ubuntu@ip-172-31-26-175:~/Books/apps/models$
```

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 register.

Change the directory back to 'Books'

cd ../..

Create a folder named public

mkdir public && cd public

Add a file named script.js

vi script.js

Copy and paste the Code below (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);
   });
  };
});
```

Save and exit with :wq
In public folder, create a file named index.html;

vi index.html
Cpoy and paste the code below into index.html file.

```
<!doctype html>
<html ng-app="myApp" ng-controller="myCtrl">
 <head>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></script>
  <script src="script.js"></script>
 </head>
 <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>{{book.name}}</td>
        <td>{{book.isbn}}</td>
        <td>{{book.author}}</td>
        <td>{{book.pages}}</td>

        <td><input type="button" value="Delete" data-ng-click="del_book(book)"></td>
       </tr>
     </table>
    </div>
  </body>
</html>
```

Change the directory back up to Books

cd ..

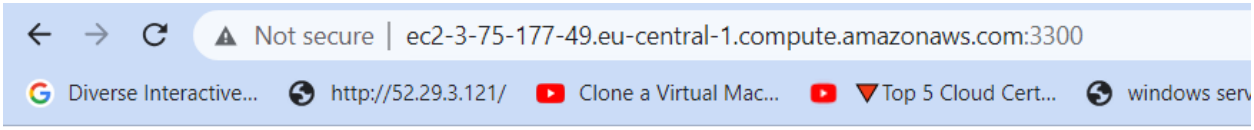
Start the server by running this command:

node server.js

open TCP port 3300 in your AWS Web Console for your EC2 Instance.

```
ubuntu@ip-172-31-47-198:~$ curl -s http://localhost:3300
<!doctype html>
<html ng-app="myApp" ng-controller="myCtrl">
  <head>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js"></script>
    <script src="script.js"></script>
  </head>
  <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>
```

```
ubuntu@ip-172-31-47-198:~$  curl -s http://169.254.169.254/latest/meta-data/public-hostname
ec2-3-75-177-49.eu-central-1.compute.amazonaws.comubuntu@ip-172-31-47-198:~$
ubuntu@ip-172-31-47-198:~$
ubuntu@ip-172-31-47-198:~$ curl -s http://169.254.169.254/latest/meta-data/public-ipv4
3.75.177.49ubuntu@ip-172-31-47-198:~$
```



Name:

Isbn:

Author:

Pages:

Add

Name Isbn Author Pages

Name:

Isbn:

Author:

Pages:

Add

Name	Isbn	Author	Pages	
Shola Adeniyi	2345	Shollybaba	5	Delete
Shola Adeniyi	2345	Shollybaba	5	Delete