



Node.JS and MongoDB Resit Project Application Guide



Hilal EKİCİ

03/02/2021

1. Introduction

This is a guide of the **GMT 351 – Geospatial Data Management** resit project application. It will explain how to create an application that uses Node.JS, MongoDB and npm.

In addition, this guide will guide you in using the Windows operating system. You have to search for other operating systems yourself.

We will follow these steps to implement this application:

1. To create *Node.js* package
2. To create *Node.js* environment
3. To connect *MongoDB* database
4. To access database through *JavaScript*
5. To display and insert data to database.
6. To filter data according to tree height
7. To sort data according to all inputs: name, latitude, longitude and tree height
8. To delete and update data

Explanation of the Tools

Javascript

JavaScript, sometimes abbreviated as JS, is a high-level, interpreted programming language that lets you implement complex elements on web pages.

It is not the same thing with Java. They are different programming languages.



Node.js

Node.js is run-time environment includes everything you need to execute a program written in JavaScript. It also is a JavaScript runtime built on [Chrome's V8 JavaScript engine](#).

In this project, I will use Node.js as a backend development platform. A backend developer uses Node.js for backend work. The Node.js allows a developer to handle data from the front end and build scalable network applications. It ables to process many simultaneous user requests.

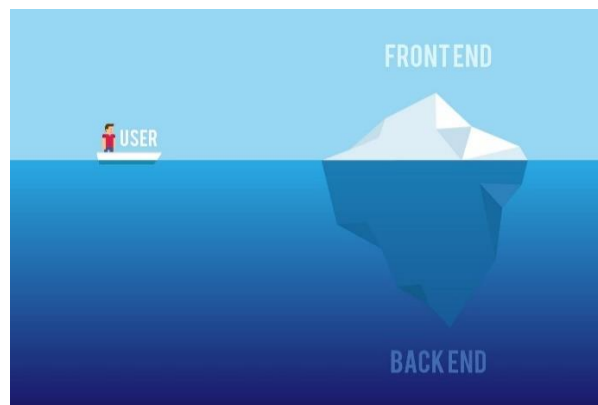
Front-End vs Back-End

Front-End and Back-End are most popular terms used in web development. The terms are very crucial for web development. They are quite different from each other. They both need to communicate and operate effectively with the other as a single unit to improve the website's functionality.

Front-End is the part of a website that user interacts with directly. It also knowns as 'client side' of the application. It includes everything that users experience directly. Text colors, styles, images, buttons, colors, and everything that comes across visually when you open a website.

Back-End means that for web development that occurs at the back end of programs. A back-end developer writes code to help a database and application communicate. Essentially, a back end developer handles what you do not see.

In this image, we can say that, a user only sees small part of an application. Big details are in the backend side.



MongoDB

MongoDB is an object-oriented, simple, dynamic, and scalable NoSQL database. It based on the NoSQL document store model. The data objects are stored as separate documents inside a collection instead of storing the data into the columns and rows of a traditional relational database.

Application Required Steps

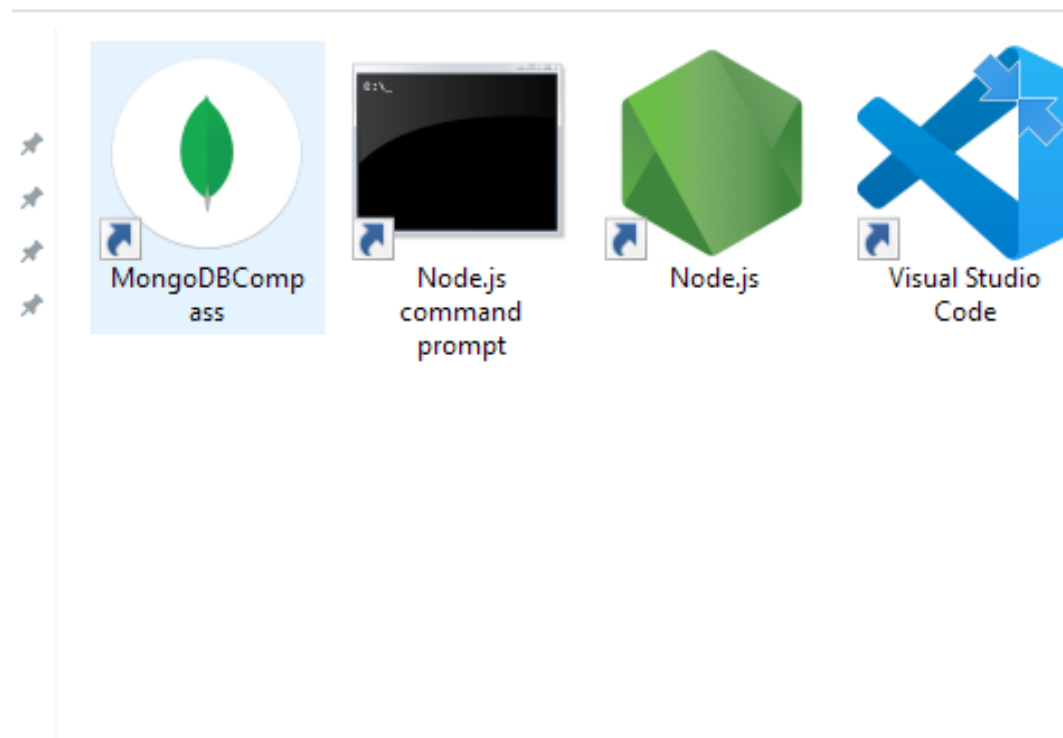
Step 1: Required Tools

The required tools for this application: Visual Studio Code, MongoDB, NodeJS, Npm

You can use another text editor instead of Visual Studio Code

You should install Node.Js, Npm and MongoDB Compass to show the table and data

1. VS Code <https://code.visualstudio.com/>
2. Node.js with Npm <https://nodejs.org/en/download/releases/>
3. MongoDB Compass <https://www.mongodb.com/products/compass>



Step 2: Check for installs

Before implementing the project, we should check whether MongoDB, Node.js and Npm installed on the computer. We should also check from Node.js's own site that Node.js and Npm have compatible versions. This is the website: <https://nodejs.org/en/download/releases/>

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.1198]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\hilar>npm --version
7.4.3

C:\Users\hilar>node --version
v15.7.0

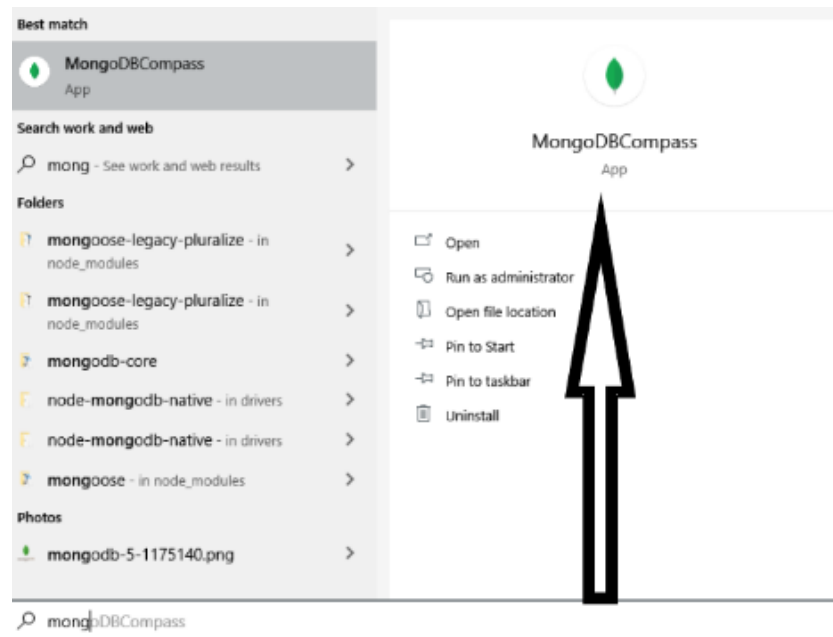
C:\Users\hilar>
```

Version	LTS	Date	V8	npm	NODE_MODULE_VERSION[1]			
Node.js 15.7.0		2021-01-25	8.6.395.17	7.4.3	88		Downloads	Changelog Docs
Node.js 15.6.0		2021-01-14	8.6.395.17	7.4.0	88		Downloads	Changelog Docs
Node.js 15.5.1		2021-01-04	8.6.395.17	7.3.0	88		Downloads	Changelog Docs
Node.js 15.5.0		2020-12-22	8.6.395.17	7.3.0	88		Downloads	Changelog Docs
Node.js 15.4.0		2020-12-09	8.6.395.17	7.0.15	88		Downloads	Changelog Docs
Node.js 15.3.0		2020-11-24	8.6.395.17	7.0.14	88		Downloads	Changelog Docs
Node.js 15.2.1		2020-11-16	8.6.395.17	7.0.8	88		Downloads	Changelog Docs
Node.js 15.2.0		2020-11-10	8.6.395.17	7.0.8	88		Downloads	Changelog Docs
Node.js 15.1.0		2020-11-04	8.6.395.17	7.0.8	88		Downloads	Changelog Docs
Node.js 15.0.1		2020-10-21	8.6.395.17	7.0.3	88		Downloads	Changelog Docs
Node.js 15.0.0		2020-10-20	8.6.395.16	7.0.2	88		Downloads	Changelog Docs
Node.js 14.15.4	Fermium	2021-01-04	8.4.371.19	6.14.10	83		Downloads	Changelog Docs
Node.js 14.15.3	Fermium	2020-12-17	8.4.371.19	6.14.9	83		Downloads	Changelog Docs
Node.js 14.15.2	Fermium	2020-12-15	8.4.371.19	6.14.9	83		Downloads	Changelog Docs
Node.js 14.15.1	Fermium	2020-11-16	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.15.0	Fermium	2020-10-27	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.14.0		2020-10-15	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.13.1		2020-10-07	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.13.0		2020-09-29	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.12.0		2020-09-22	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.11.0		2020-09-15	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.10.1		2020-09-10	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.10.0		2020-09-08	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.9.0		2020-08-27	8.4.371.19	6.14.8	83		Downloads	Changelog Docs
Node.js 14.8.0		2020-08-11	8.4.371.19	6.14.7	83		Downloads	Changelog Docs
Node.js 14.7.0		2020-07-29	8.4.371.19	6.14.7	83		Downloads	Changelog Docs
Node.js 14.6.0		2020-07-20	8.4.371.19	6.14.6	83		Downloads	Changelog Docs
Node.js 14.5.0		2020-06-30	8.3.110.9	6.14.5	83		Downloads	Changelog Docs
Node.js 14.4.0		2020-06-02	8.1.307.31	6.14.5	83		Downloads	Changelog Docs
Node.js 14.3.0		2020-05-19	8.1.307.31	6.14.5	83		Downloads	Changelog Docs

- npm --version or npm -v
- node --version or node --V

When you see if node.js and npm versions are compatible, then this

For **mongoDB**: Start > Search > MongoDB



Step 3: Creating Package

After that, we will create our package for the application. As you show from the screenshot;

- Firstly, we open **cmd** with run as administrator (command line).
- Change your path with **cd** command where you want the project in.
- Create project directory using **mkdir** command.

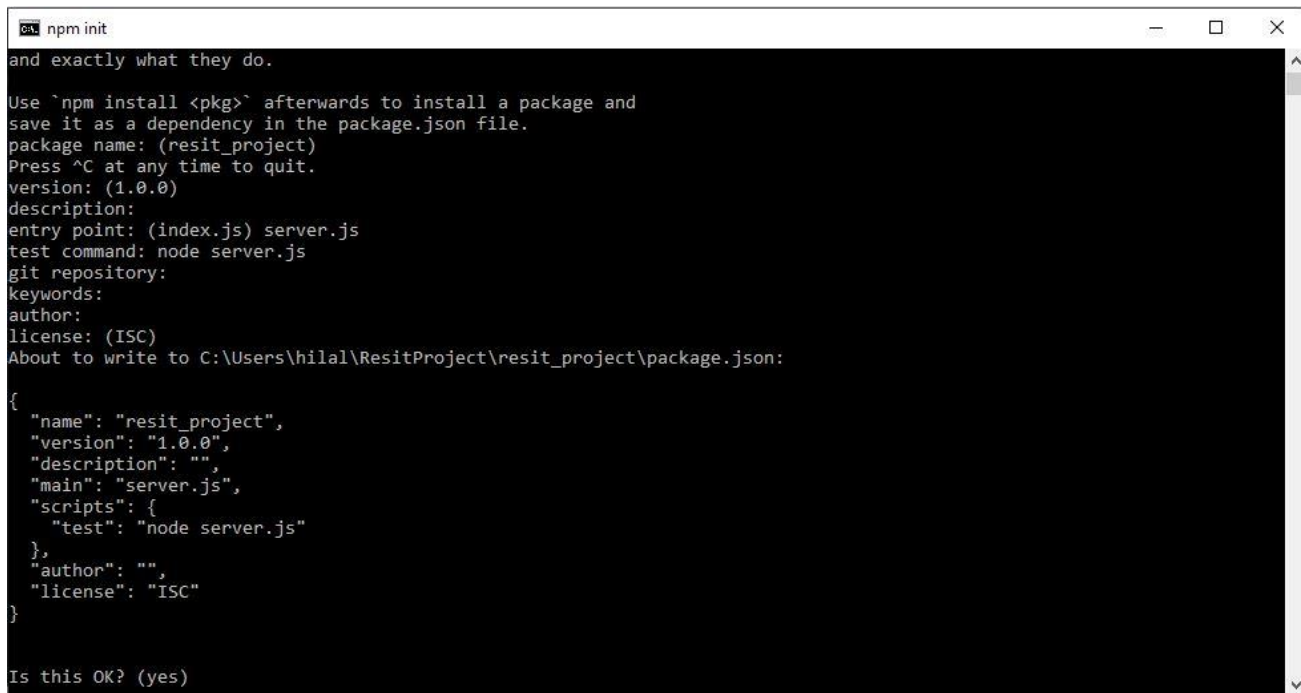
```
npm init
C:\Users\hilal\ResitProject>mkdir resit_project
C:\Users\hilal\ResitProject>cd resit_project
C:\Users\hilal\ResitProject\resit_project>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

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.
package name: (resit_project)

Press ^C at any time to quit.
```

- After that, we should run **npm init** to set up a new **npm** package. This command shows questions to create package.json. You can press enter to skip questions, all of them will default value. Only part is **Entry Point**. The project's entry point (meaning the project's main file), The project's test command (to trigger testing with something like Standard). We set it as **server.js**



```
npm init
and exactly what they do.

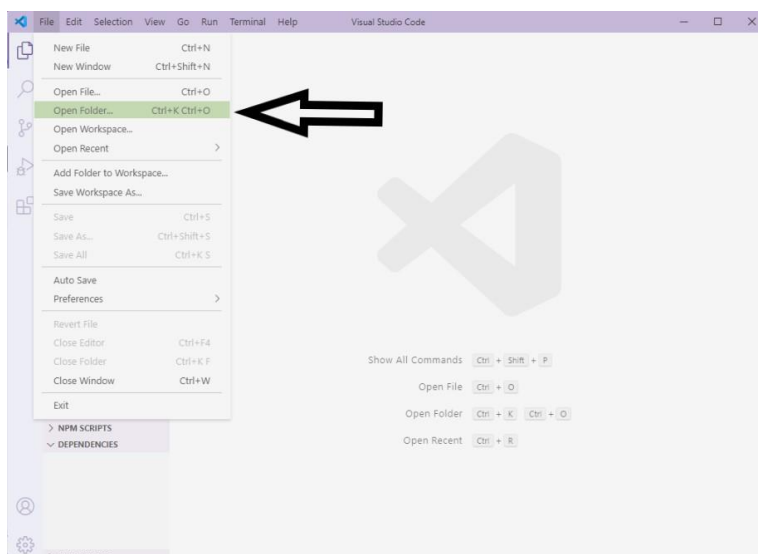
Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
package name: (resit_project)
Press ^C at any time to quit.
version: (1.0.0)
description:
entry point: (index.js) server.js
test command: node server.js
git repository:
keywords:
author:
license: (ISC)
About to write to C:\Users\hilal\ResitProject\resit_project\package.json:

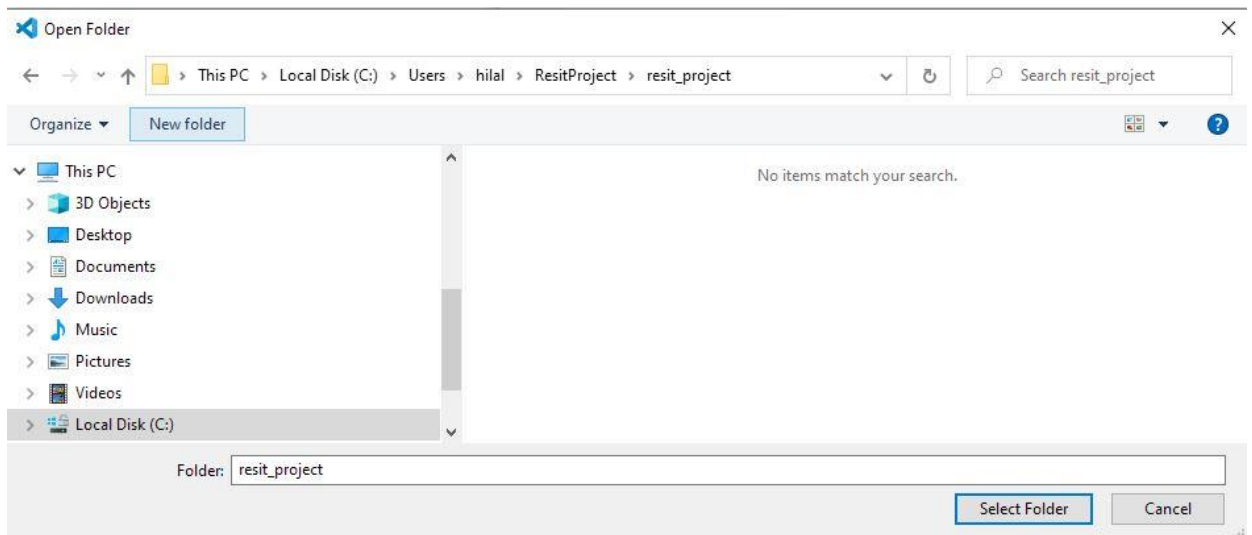
{
  "name": "resit_project",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "node server.js"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes)
```

Step 4: Creating Environment

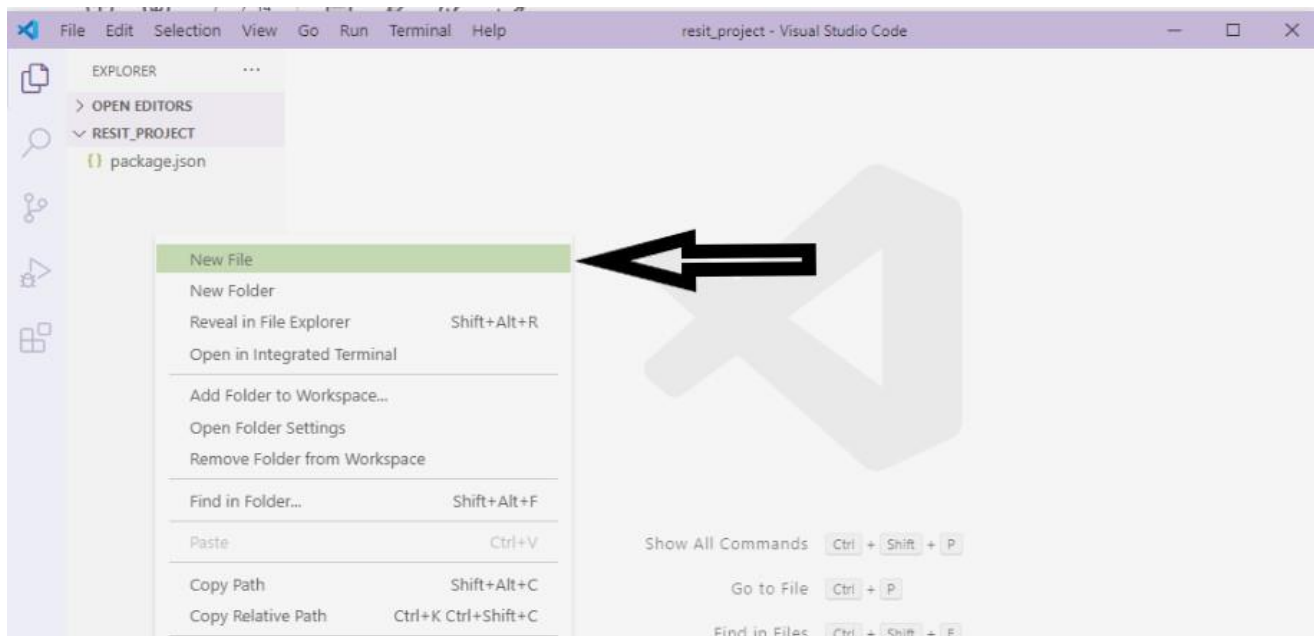
- After, installed VS Code, open it.
- File > Open folder
- Open the folder containing package.json



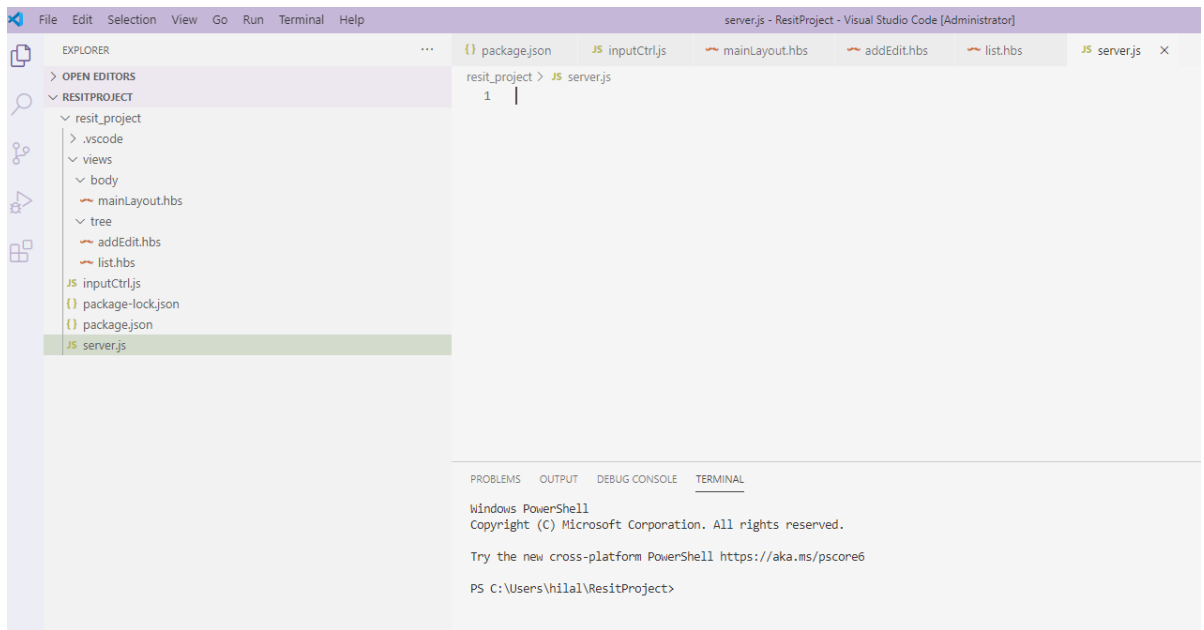


Step 5: Creating Code File

- Right click on VSCode > **“New File”**
- Give name **“server.js”**
- Do same thing for **“inputCtrl.js”, “mainLayout.hbs”, “list.hbs”, “addEdit.lbs”**



After all steps, you should check the application folder:

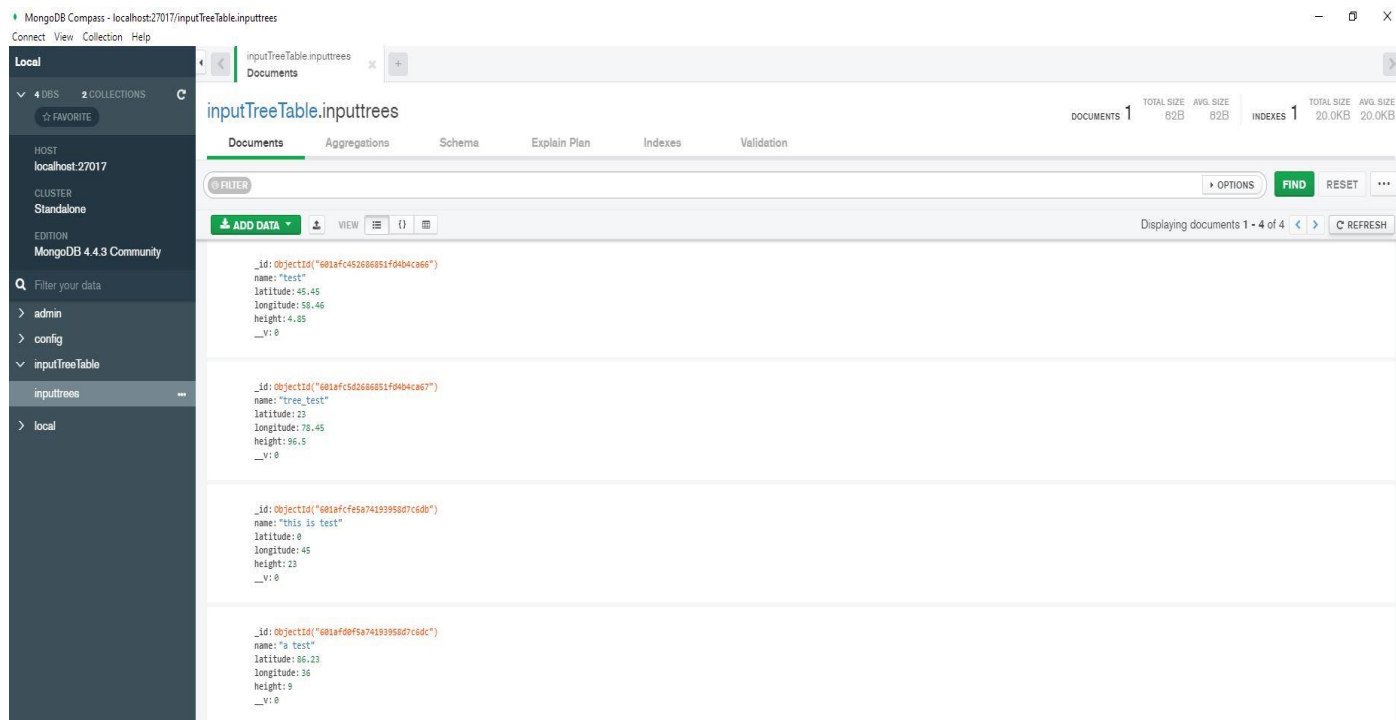


Step 6: Designing Database

In MongoDB, you can handle all database detail from your code files.

```
1  const mongoose = require('mongoose');
2  mongoose.connect('mongodb://localhost:27017/inputTreeTable', { useNewUrlParser: true }, (err) => {
3    if (!err) {
4      console.log('DB connection is successful.')
5    }
6    else {
7      console.log('Error in DB connection : ' + err) |
8    }
9  });
10 var inputTable = new mongoose.Schema({
11   name: {
12     type: String,
13     required: 'This field is required.'
14   },
15   latitude: {
16     type: Number,
17     required: 'This field is required.'
18   },
19   longitude: {
20     type: Number,
21     required: 'This field is required.'
22   },
23   height: {
24     type: Number,
25     required: 'This field is required.'
26   }
27 });
28 mongoose.model('InputTree', inputTable);
```

We use **mongoose** for connecting to the database. I choosed table name: **inputTreeTable**. After that, created a schema with four fields: **name**, **latitude**, **longitude** and **height**. None of them can be empty. These code lines ensures inputs' **validation**. User cannot enter string data for latitude, longitude and height. Beside, user cannot enter number data for name field.



Step 7: Packages

We should install packages now. To install packages, use **npm install** command.

npm install package_name: Install the dependencies in the local node_modules folder.

We need run these commands from terminal:

- **npm install mongoose**

This package is for database operations.

- **npm install express**

Express.js is a Node.js web application server framework, which is specifically designed for building single-page, multi-page, and hybrid web applications..

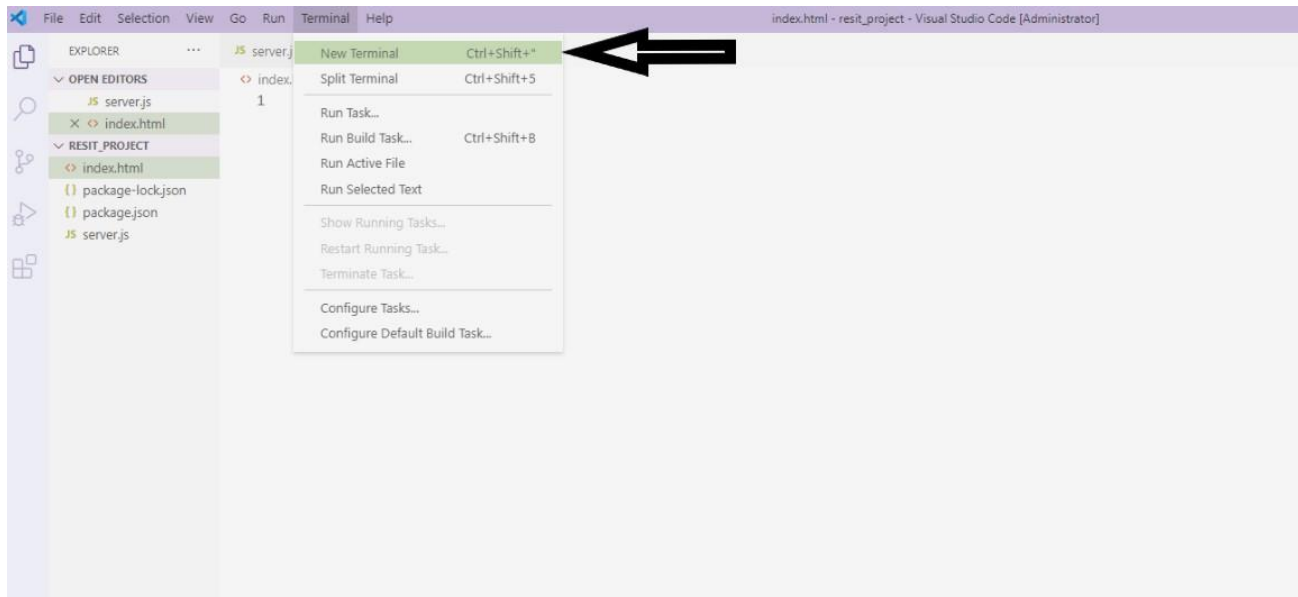
- **npm install express-handlebars**

This package is for user interface part. It Includes Handlebar Helper to manage sections in layouts.

- **npm install body-parser**

Node.js body parsing middleware.

- **Open terminal:** Terminal > New Terminal



- **Type your commands for installing:**

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\hilal\ResitProject\resit_project> npm install express

added 50 packages, and audited 51 packages in 4s

found 0 vulnerabilities
PS C:\Users\hilal\ResitProject\resit_project> █
```

```
PS C:\Users\hilal\ResitProject\resit_project> npm install mongoose

up to date, audited 82 packages in 2s

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

found 0 vulnerabilities
PS C:\Users\hilal\ResitProject\resit_project> █
```

```
PS C:\Users\hilal\ResitProject\resit_project> npm install express-handlebars

added 18 packages, and audited 100 packages in 2s

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

found 0 vulnerabilities
PS C:\Users\hilal\ResitProject\resit_project> █
```

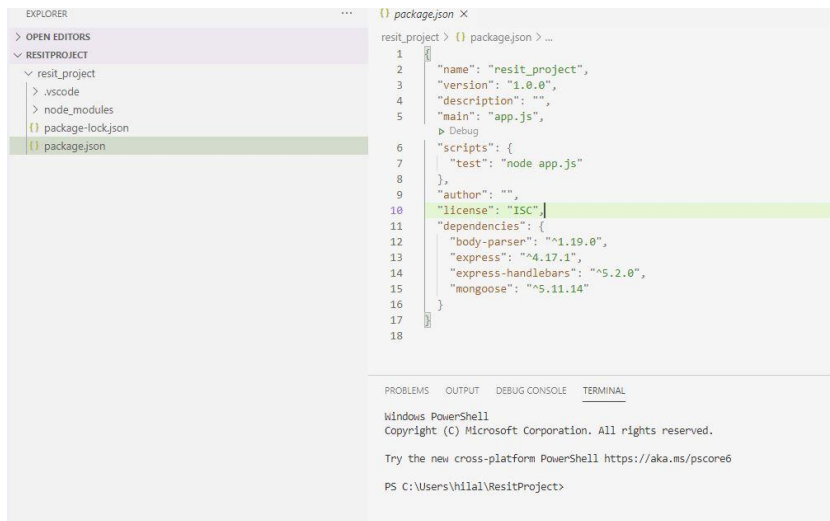
```
PS C:\Users\hilal\ResitProject\resit_project> npm install body-parser

up to date, audited 70 packages in 916ms

1 package is looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\hilal\ResitProject\resit_project> █
```

- **After installing**, check your package.json and node_modules folder if your packages installed.



Step 8: Coding

server.js

https://github.com/HilalEKC/resit_project/blob/main/server.js

```
1 const mongoose = require('mongoose');
2 mongoose.connect('mongodb://localhost:27017/inputTreeTable', { useNewUrlParser: true }, (err) => {
3   if (err) {
4     console.log('DB connection is successful.')
5   }
6   else {
7     console.log('Error in DB connection : ' + err)
8   }
9 });
10 var inputTable = new mongoose.Schema({
11   name: {
12     type: String,
13     required: 'This field is required.'
14   },
15   latitude: {
16     type: Number,
17     required: 'This field is required.'
18   },
19   longitude: {
20     type: Number,
21     required: 'This field is required.'
22   },
23   height: {
24     type: Number,
25     required: 'This field is required.'
26   }
27 });
28 mongoose.model('InputTree', inputTable);
29 const express = require('express');
30 const path = require('path');
31 const exphbs = require('express-handlebars');
32 const bodyparser = require('body-parser');
33 var app = express();
34 app.use(bodyparser.urlencoded({
35   extended: true
36 }));
37 app.use(bodyparser.json());
38 app.set('views', path.join(__dirname, '/views/'));
39 app.engine('hbs', exphbs({ extname: 'hbs', defaultLayout: 'mainLayout', layoutsDir: __dirname + '/views/body/' }));
40 app.set('view engine', 'hbs');
41 app.listen(5000, () => {
42   console.log('Express server started at port : 5000');
43 });
44 const inputCtrl = require('./inputCtrl');
45 app.use('/tree', inputCtrl);
```

list.hbs

https://github.com/HilaleKC/resit_project/blob/main/views/tree/list

.hbs

```
1 <h3 style="text-align: center;">
2   All Data From Database
3 </h3>
4
5 <button style="float:left"><a class="btn" style="width:250px" href="/tree/list">Show All Data</a> </button>
6 <button style="float:right"><a class="btn" style="width:250px" href="/tree">Submit New Data</a> </button>
7 <table class="table">
8   <thead>
9     <tr>
10      <th>Name</th>
11      <th>Latitude</th>
12      <th>Longitude</th>
13      <th>Tree Height</th>
14    </tr>
15  </thead>
16  <tbody>
17    <{{#each list}}>
18      <tr>
19        <td>{{this.name}}</td>
20        <td>{{this.latitude}}</td>
21        <td>{{this.longitude}}</td>
22        <td>{{this.height}}</td>
23        <td>
24          <a href="/tree/{{this._id}}">Edit</a>
25          <a href="/tree/delete/{{this._id}}" onclick="return confirm('Do you want to delete ?');">
26            Delete
27          </a>
28        </td>
29      </tr>
30    </tr>
31    </tbody>
32  </table>
33
34
35 <button><a class="btn btn-block" style="width:250px" href="/tree/api/sortLatitude">Sort Ascending Latitude </a> </button>
36 <button style="float: right;"><a class="btn btn-block" style="width:250px" href="/tree/api/sortName">Sort Ascending Name </a> </button>
37 <button><a class="btn btn-block" style="width:250px" href="/tree/api/sortLongitude">Sort Ascending Longitude </a> </button>
38 <button style="float: right;"><a class="btn btn-block" style="width:250px" href="/tree/api/sortHeight">Sort Ascending Tree Height </a> </button>
39 <button><a class="btn btn-block" style="width:250px" href="/tree/api/filterHeightLt50">Filter Tree Height Less Than 50 </a> </button>
40 <button style="float: right;"><a class="btn btn-block" style="width:250px;" href="/tree/api/filterHeightGt50">Filter Tree Height Greater Than 50 </a> </button>
```

mainLayout.hbs

https://github.com/HilaleKC/resit_project/blob/main/views/body/mainLayout

hbs

```
1
2 <html>
3 <head>
4   <title>Resit Project</title>
5   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPk
6 </head>
7 <body>
8   <div class="row">
9     <div class="col-md-6 offset-md-3" style="background-color: #ffff">
10      <div>
11        <div>
12        </div>
13      </div>
14    </div>
```

addEdit.hbs

https://github.com/HilaleKC/resit_project/blob/main/views/tree/addEdit.hbs

```
1 <h3>{{viewTitle}}</h3>
2 <div class="form-group" style="float: right;">
3   <button><a class="btn" href="/tree/list">Show All Data</a></button>
4 </div>
5 <form action="/tree" method="POST">
6
7   <input type="hidden" name="_id" value="{{tree._id}}">
8   <div class="form-group">
9     <label>Name</label>
10    <input type="text" class="form-control" name="name" placeholder="Name" value="{{tree.name}}">
11    <div class="text-danger">
12      {{tree.nameError}}</div>
13  </div>
14  <div class="form-group">
15    <label>Latitude</label>
16    <input type="text" class="form-control" name="latitude" placeholder="39.865649" value="{{tree.latitude}}">
17    <div class="text-danger">
18      {{tree.latitudeError}}</div>
19  </div>
20
21  <div class="form-group">
22    <label>Longitude</label>
23    <input type="text" class="form-control" name="longitude" placeholder="32.733520" value="{{tree.longitude}}">
24    <div class="text-danger">
25      {{tree.longitudeError}}</div>
26  </div>
27  <div class="form-group">
28    <label>Tree Height</label>
29    <input type="text" class="form-control" name="height" placeholder="4.20" value="{{tree.height}}">
30    <div class="text-danger">
31      {{tree.heightError}}</div>
32  </div>
33
34  <div class="form-group">
35    <button type="submit">Submit</button>
36  </div>
37
38 </form>
```

inputCtrl.js – 1

https://github.com/HilaleKC/resit_project/blob/main/inputCtrl.js

```
1 const express = require('express');
2 var router = express.Router();
3 const mongoose = require('mongoose');
4 const treeInput = mongoose.model('InputTree');
5
6 router.get('/', (req, res) => {
7   res.render("tree/addEdit", {
8     viewTitle: "Input Tree"
9   });
10 });
11
12 router.post('/', (req, res) => {
13   if (req.body._id == '') {
14     insertRecord(req, res);
15   }
16   else {
17     updateRecord(req, res);
18   }
19 });
20
21 function insertRecord(req, res) {
22   var tree = new treeInput();
23   tree.name = req.body.name;
24   tree.latitude = req.body.latitude;
25   tree.longitude = req.body.longitude;
26   tree.height = req.body.height;
27
28   tree.save((err) => {
29     if (err) {
30       res.redirect('tree/list');
31     }
32     else {
33       if (err.name == 'ValidationError') {
34         handleValidationError(err, req.body);
35         res.render("tree/addEdit", {
36           viewTitle: "Input Tree",
37           tree: req.body
38         });
39       }
40     }
41   });
42 }
```

inputCtrl.js - 2

```
43 function updateRecord(req, res) {
44   treeInput.findOneAndUpdate({ _id: req.body._id }, req.body, { new: true }, (err, doc) => {
45     if (!err) {
46       res.redirect('tree/list');
47     }
48     else {
49       if (err.name == 'ValidationError') {
50         handleValidationError(err, req.body);
51         res.render("tree/addEdit", {
52           viewTitle: 'Update Input',
53           tree: req.body
54         });
55       }
56     }
57   });
58 }
59 router.get('/list', (req, res) => {
60   treeInput.find((err, docs) => {
61     res.render("tree/list", {
62       list: docs
63     });
64   });
65 });
66 router.get('/api/data', (req, res) => {
67   treeInput.find((err, docs) => {
68     res.send(docs);
69   });
70 });
71
72 router.get('/api/sortLatitude', (req, res) => {
73
74   treeInput.find({}).sort({latitude: 1}).exec(function(err, docs){
75     res.render("tree/list", {
76       list: docs
77     });
78   })
79 })
80 router.get('/api/sortName', (req, res) => {
81
82   treeInput.find({}).sort({name: 1}).exec(function(err, docs){
83     res.render("tree/list", {
84       list: docs
85     });
86   })
87 });
```

inputCtrl.js - 3

```
88 router.get('/api/sortLongitude', (req, res) => {
89
90   treeInput.find({}).sort({longitude: 1}).exec(function(err, docs){
91     res.render("tree/list", {
92       list: docs
93     });
94   })
95 });
96 router.get('/api/sortHeight', (req, res) => {
97
98   treeInput.find({}).sort({height: 1}).exec(function(err, docs){
99     res.render("tree/list", {
100       list: docs
101     });
102   })
103 });
104 router.get('/api/filterHeightLt50', (req, res) => {
105
106   treeInput.find({'height' : {$lt: 50}}).exec(function(err, docs){
107     res.render("tree/list", {
108       list: docs
109     });
110   })
111 });
112 router.get('/api/filterHeightGt50', (req, res) => {
113
114   treeInput.find({'height' : {$gt: 50}}).exec(function(err, docs){
115     res.render("tree/list", {
116       list: docs
117     });
118   })
119 });
```

inputCtrl.js - 4

```
120 function handleValidationError(err, body) {
121   for (field in err.errors) {
122     switch (err.errors[field].path) {
123       case 'name':
124         body['nameError'] = err.errors[field].message;
125         break;
126       case 'latitude':
127         body['latitudeError'] = err.errors[field].message;
128         break;
129       case 'longitude':
130         body['longitudeError'] = err.errors[field].message;
131         break;
132       case 'height':
133         body['heightError'] = err.errors[field].message;
134         break;
135       default:
136         break;
137     }
138   }
139 }
140 router.get('/:id', (req, res) => {
141   treeInput.findById(req.params.id, (err, doc) => {
142     res.render("tree/addEdit", {
143       viewTitle: "Update Input",
144       tree: doc
145     });
146   });
147 });
148 router.get('/delete/:id', (req, res) => {
149   treeInput.findByIdAndRemove(req.params.id, (err, doc) => {
150     res.redirect('/tree/list');
151   });
152 });
153 module.exports = router;
```

Step 9: Testing

We can test our project with VS Code terminal, browser and MongoDB Compass.

- **Type “node server.js”** to VS Code terminal
- **Open** MongoDB Compass
- **Go** to <http://localhost:5000/tree>

1. First page: submit input

Input Tree

Name

Latitude

Longitude

Tree Height

shows all data in db

[Show All Data](#)

adds new data

2. After submit button clicked

All Data From Database

[Show All Data](#)

[Submit New Data](#)

Name	Latitude	Longitude	Tree Height	
test	45.45	58.46	4.85	Edit Delete
tree_test	23	78.45	96.5	Edit Delete
this is test	0	45	23	Edit Delete
a test	86.23	36	9	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete

[Sort Ascending Latitude](#)

[Sort Ascending Name](#)

[Sort Ascending Longitude](#)

[Sort Ascending Tree Height](#)

[Filter Tree Height Less Than 50](#)

[Filter Tree Height Greater Than 50](#)

3. Sort ascending order by tree height

All Data From Database				
Show All Data			Submit New Data	
Name	Latitude	Longitude	Tree Height	
test	45.45	58.46	4.85	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete
a test	86.23	36	9	Edit Delete
this is test	0	45	23	Edit Delete
tree_test	23	78.45	96.5	Edit Delete
Sort Ascending Latitude			Sort Ascending Name	
Sort Ascending Longitude			Sort Ascending Tree Height	
Filter Tree Height Less Than 50			Filter Tree Height Greater Than 50	

4. Sort ascending order by name

All Data From Database				
Show All Data			Submit New Data	
Name	Latitude	Longitude	Tree Height	
a test	86.23	36	9	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete
test	45.45	58.46	4.85	Edit Delete
this is test	0	45	23	Edit Delete
tree_test	23	78.45	96.5	Edit Delete
Sort Ascending Latitude			Sort Ascending Name	
Sort Ascending Longitude			Sort Ascending Tree Height	
Filter Tree Height Less Than 50			Filter Tree Height Greater Than 50	

4. Sort ascending order by latitude

All Data From Database

Show All Data			Submit New Data	
Name	Latitude	Longitude	Tree Height	
this is test	0	45	23	Edit Delete
tree_test	23	78.45	96.5	Edit Delete
test	45.45	58.46	4.85	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete
a test	86.23	36	9	Edit Delete
Sort Ascending Latitude			Sort Ascending Name	
Sort Ascending Longitude			Sort Ascending Tree Height	
Filter Tree Height Less Than 50			Filter Tree Height Greater Than 50	

5. Sort ascending order by longitude

All Data From Database

Show All Data			Submit New Data	
Name	Latitude	Longitude	Tree Height	
a test	86.23	36	9	Edit Delete
this is test	0	45	23	Edit Delete
test	45.45	58.46	4.85	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete
tree_test	23	78.45	96.5	Edit Delete
Sort Ascending Latitude			Sort Ascending Name	
Sort Ascending Longitude			Sort Ascending Tree Height	
Filter Tree Height Less Than 50			Filter Tree Height Greater Than 50	

6. Filter tree height greater than 50

All Data From Database

Show All Data

Submit New Data

Name	Latitude	Longitude	Tree Height	
tree_test	23	78.45	96.5	Edit Delete

Sort Ascending Latitude

Sort Ascending Longitude

Filter Tree Height Less Than 50

Sort Ascending Name

Sort Ascending Tree Height

Filter Tree Height Greater Than 50

7. Filter tree height less than 50

All Data From Database

Show All Data

Submit New Data

Name	Latitude	Longitude	Tree Height	
test	45.45	58.46	4.85	Edit Delete
this is test	0	45	23	Edit Delete
a test	86.23	36	9	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete

Sort Ascending Latitude

Sort Ascending Longitude

Filter Tree Height Less Than 50

Sort Ascending Name

Sort Ascending Tree Height

Filter Tree Height Greater Than 50

8. Edit data

Update Input

Name

Show All Data

test

Latitude

45.45

Longitude

58.46

Tree Height

4.85

Submit

9. Delete data

localhost:5000/tree/api/filterHeightLt50

localhost:5000 says
Do you want to delete ?

OK

Cancel

Show All Data

Submit New Data

Name	Latitude	Longitude	Tree Height	
test	45.45	58.46	4.85	Edit Delete
this is test	0	45	23	Edit Delete
a test	86.23	36	9	Edit Delete
bi_test	45.4856	63.47856	8.63	Edit Delete

Sort Ascending Latitude

Sort Ascending Longitude

Filter Tree Height Less Than 50

Sort Ascending Name

Sort Ascending Tree Height

Filter Tree Height Greater Than 50

10. Database

inputTreeTable.inputtrees Documents

DOCUMENTS 1 TOTAL SIZE 62B AVG SIZE 62B INDEXES 1 TOTAL SIZE 20.0KB AVG SIZE 20.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

ADD DATA VIEW

Displaying documents 1 - 5 of 5

#	inputtrees	_id ObjectId	name string	latitude Mixed	longitude Mixed	height Mixed	_v Int32
1	601afc452606051f94b4c966		"test"	45.45	58.46	4.85	0
2	601afc452606051f94b4c967		"tree_test"	23	78.45	96.5	0
3	601afc45a74393958d7c6db		"this is test"	0	45	23	0
4	601afc45a74393958d7c6dc		"a test"	86.23	36	9	0
5	60100c65a74393958d7c6dd		"bi_test"	45.4856	63.47856	8.63	0

Notes

- And we can view all data in JSON format using API:

<http://localhost:5000/tree/api/data>

```
[{"_id":"601afc5d2686851fd4b4ca67","name":"tree_test","latitude":23,"longitude":78.45,"height":96.5,"__v":0},
{"_id":"601afcf5a74193958d7c6db","name":"this is test","latitude":0,"longitude":45,"height":23,"__v":0},
{"_id":"601afd0f5a74193958d7c6dc","name":"a test","latitude":86.23,"longitude":36,"height":9,"__v":0},
{"_id":"601b0cde5a74193958d7c6dd","name":"bi_test","latitude":45.4856,"longitude":63.47856,"height":8.63,"__v":0}
]
```

- Here are the all urls:

- <http://localhost:5000/tree/>
- <http://localhost:5000/tree/list>
- <http://localhost:5000/tree/api/sortLatitude>
- <http://localhost:5000/tree/api/sortName>
- <http://localhost:5000/tree/api/sortLongitude>
- <http://localhost:5000/tree/api/sortHeight>
- <http://localhost:5000/tree/api/filterHeightLt50>
- <http://localhost:5000/tree/api/filterHeightGt50>
-

- All steps are done. **Thank you.**