Rx Savings Solutions Backend Developer Code Challenge

Submitted by : Pallavi Desai

This document contains

Objective

- Create a very basic API
- The API will receive a latitude/longitude, using the provided list in pharmacies.csv the API will calculate the closest pharmacy to the input latitude/longitude.
- The API will return the closest pharmacy (with name and address) in a consumable response packet.
- The API will include the distance (in miles) between the input latitude/longitude and the selected closest pharmacy in the

response packet.

Technologies used

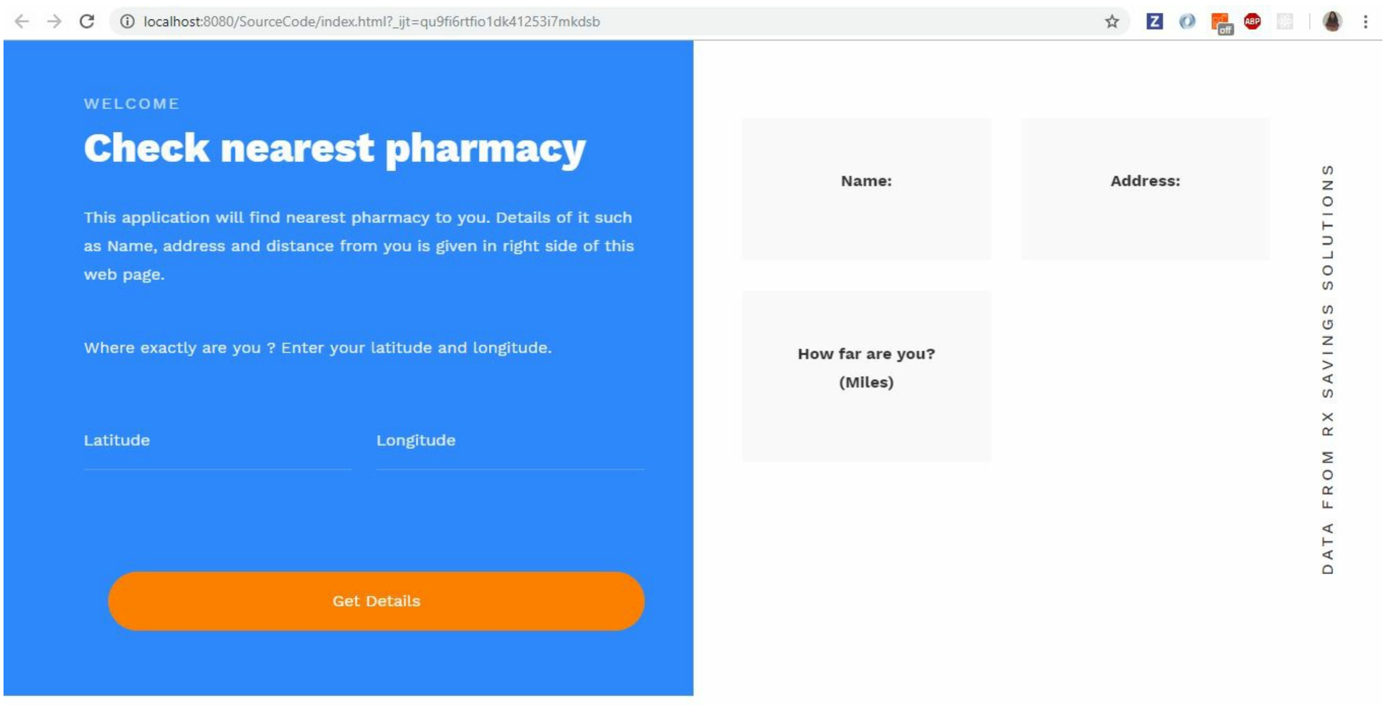
- Front end : HTML5, CSS3, AngularJS 1.4.4, Bootstrap,
- Database Used : MongoDB
- Back end : NodeJS 6.4.1

How to execute application

- Save source code and navigate until you find package.json
- Open command prompt to this path and type “npm install”
- This will install all dependencies. It may take a while, be patient.
- Run server.js. Now your computer will be listening to port 5000.
- Run index.html in your browser.
- Input latitude and longitude and press “Find Pharmacy”.
- Web app will give you nearest pharmacy.

Input/ Output

Input : This web application receive current latitude and longitude of user.



localhost:8080/SourceCode/index.html?_ijt=qu9fi6rtfio1dk41253i7mkdsb

WELCOME

Check nearest pharmacy

This application will find nearest pharmacy to you. Details of it such as Name, address and distance from you is given in right side of this web page.

Where exactly are you ? Enter your latitude and longitude.

Latitude

Longitude

Get Details

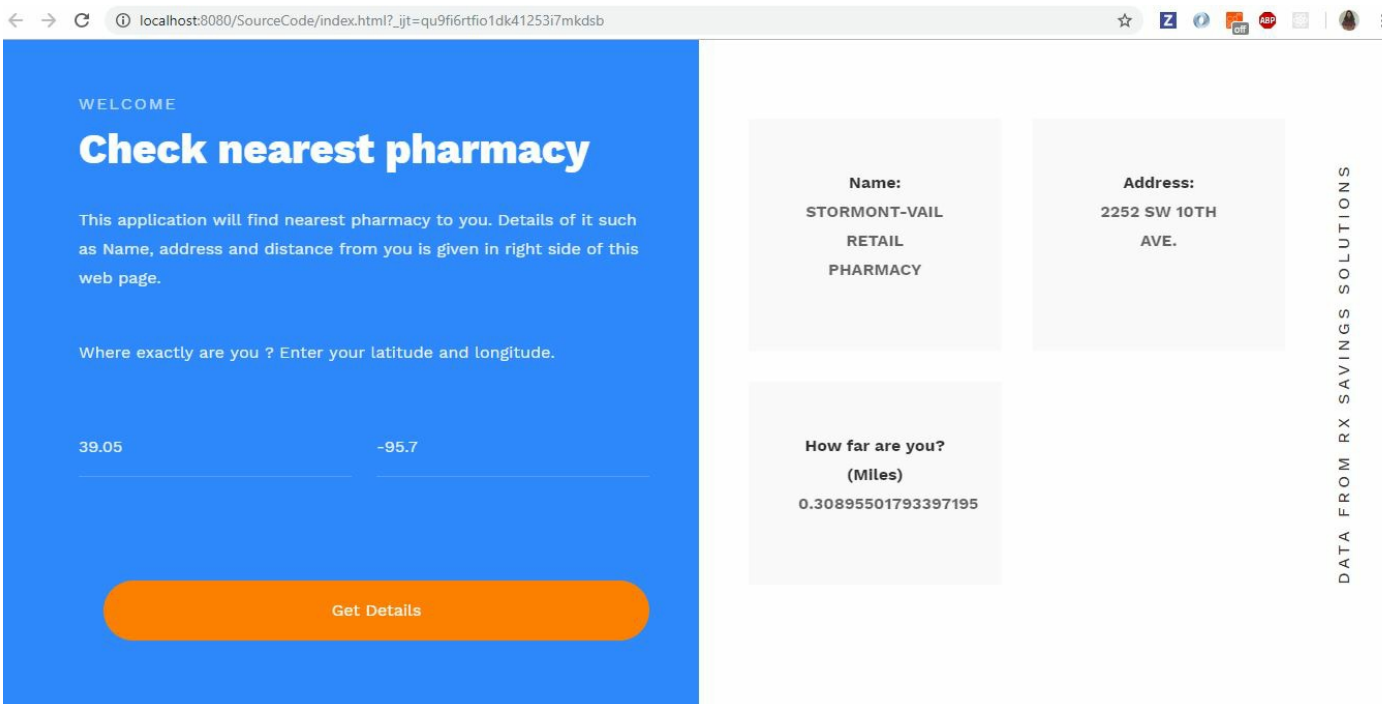
Name:

Address:

How far are you? (Miles)

DATA FROM RX SAVINGS SOLUTIONS

Output : This web application include the distance (in miles) between the input user latitude and longitude and the selected closest pharmacy in the response packet.


localhost:8080/SourceCode/index.html?_ijt=qu9fi6rtfio1dk41253i7mkdsb

WELCOME

Check nearest pharmacy

This application will find nearest pharmacy to you. Details of it such as Name, address and distance from you is given in right side of this web page.

Where exactly are you ? Enter your latitude and longitude.

39.05 -95.7

Get Details

Name: STORMONT-VAIL RETAIL PHARMACY

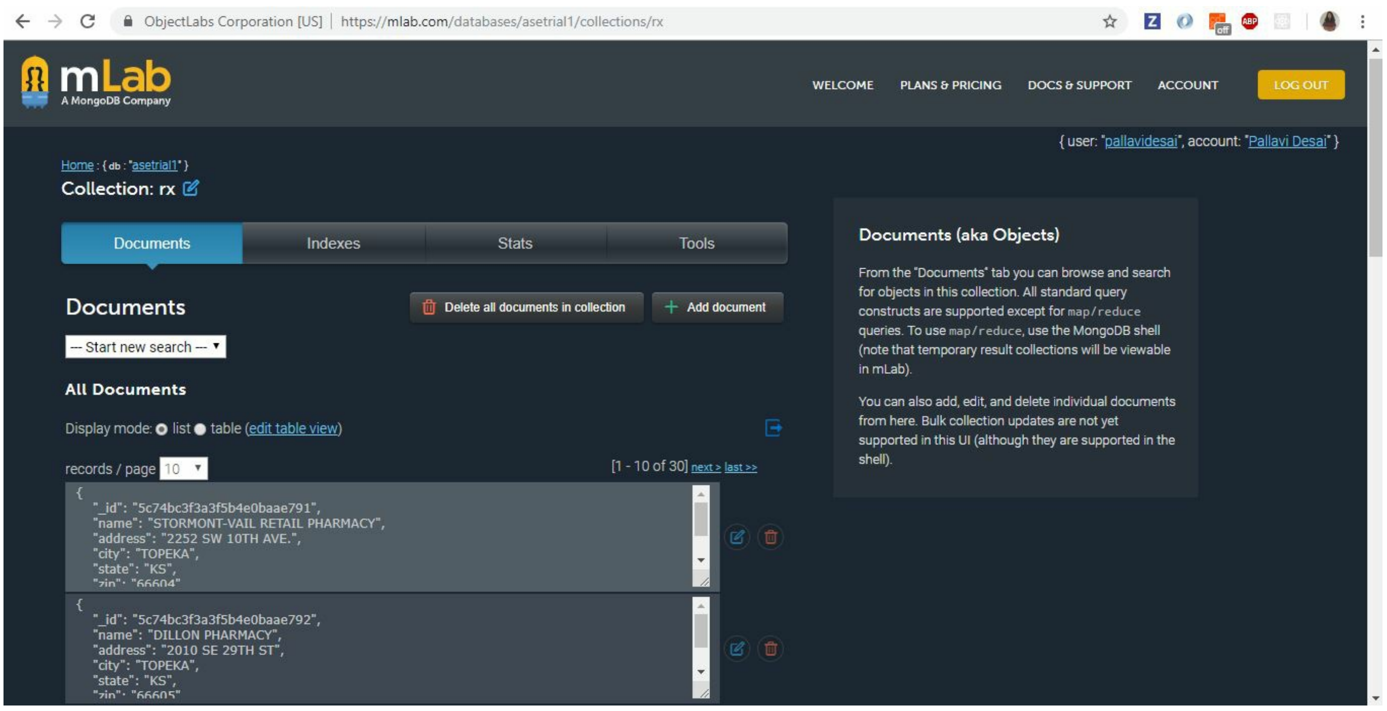
Address: 2252 SW 10TH AVE.

How far are you? (Miles) 0.30895501793397195

DATA FROM RX SAVINGS SOLUTIONS

Approach

Step 1 : Imported given CSV file into my mlab.


ObjectLabs Corporation [US] | https://mlab.com/databases/asetrial1/collections/rx

{ user: "pallavidesai", account: "Pallavi Desai" }

Home : { db : "asetrial1" }

Collection: rx

Documents

All Documents

Display mode: list table (edit table view)

records / page 10 [1 - 10 of 30] next > last >>

```
{
    "_id": "5c74bc3f3a3f5b4e0baae791",
    "name": "STORMONT-VAIL RETAIL PHARMACY",
    "address": "2252 SW 10TH AVE.",
    "city": "TOPEKA",
    "state": "KS",
    "zip": "66604"
```

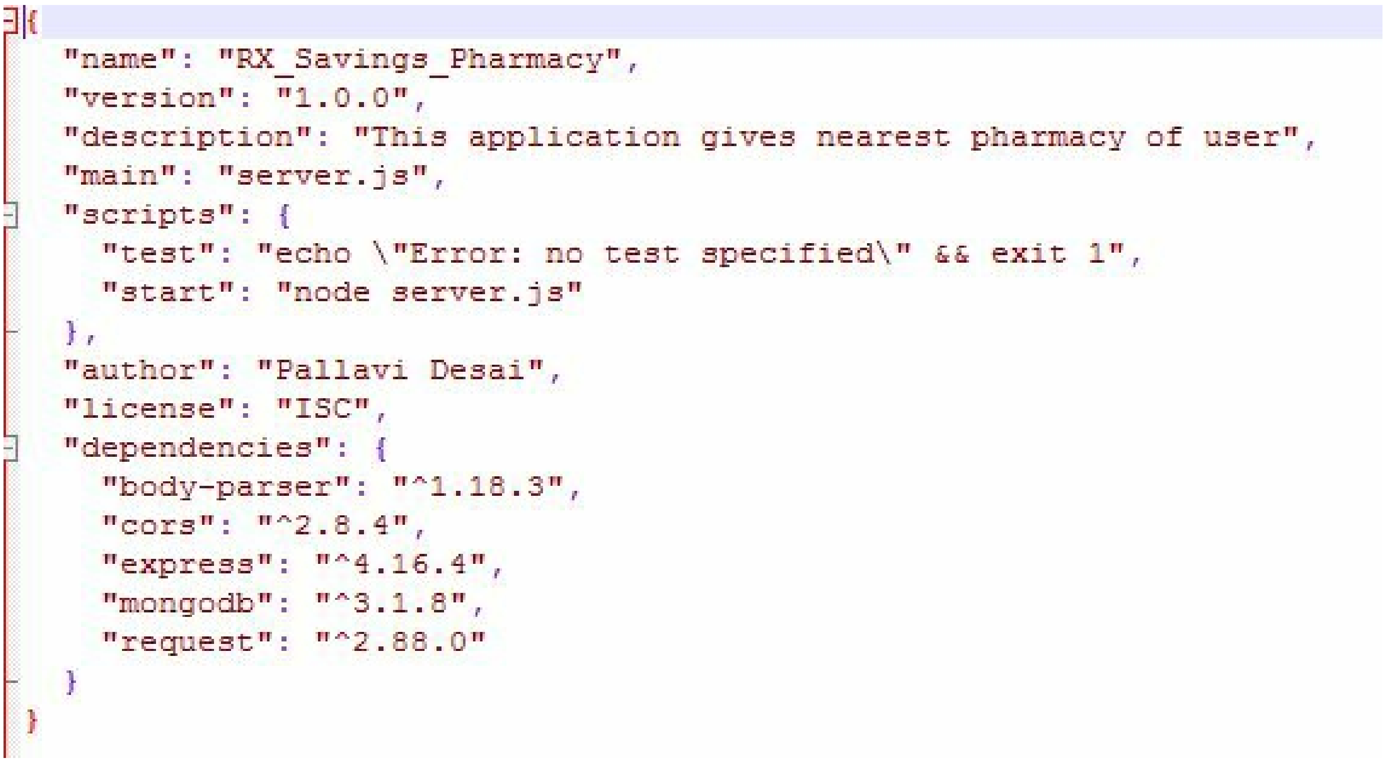
```
{
    "_id": "5c74bc3f3a3f5b4e0baae792",
    "name": "DILLON PHARMACY",
    "address": "2010 SE 29TH ST",
    "city": "TOPEKA",
    "state": "KS",
    "zip": "66605"
```

Documents (aka Objects)

From the "Documents" tab you can browse and search for objects in this collection. All standard query constructs are supported except for map/reduce queries. To use map/reduce, use the MongoDB shell (note that temporary result collections will be viewable in mLab).

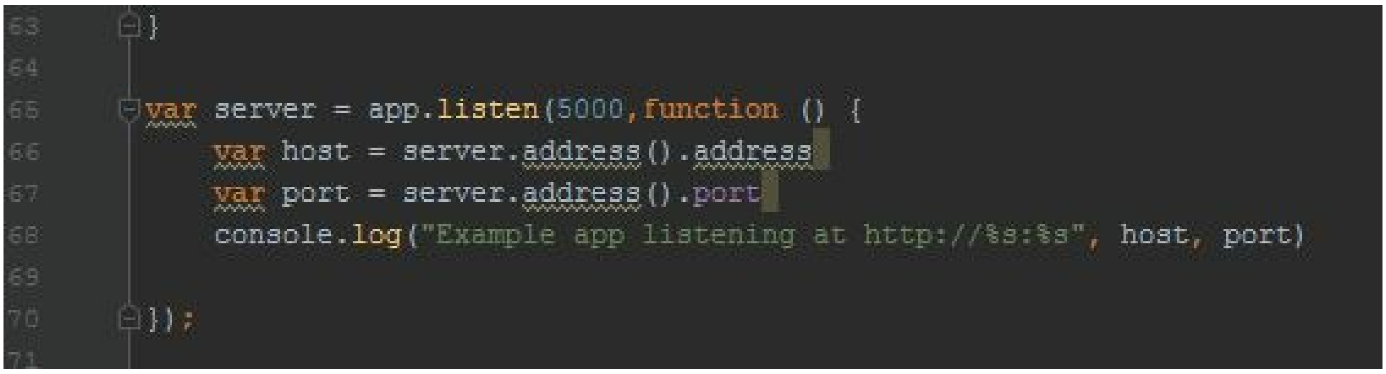
You can also add, edit, and delete individual documents from here. Bulk collection updates are not yet supported in this UI (although they are supported in the shell).

Step 2 : Created package.json file and installed it.This will install all dependencies



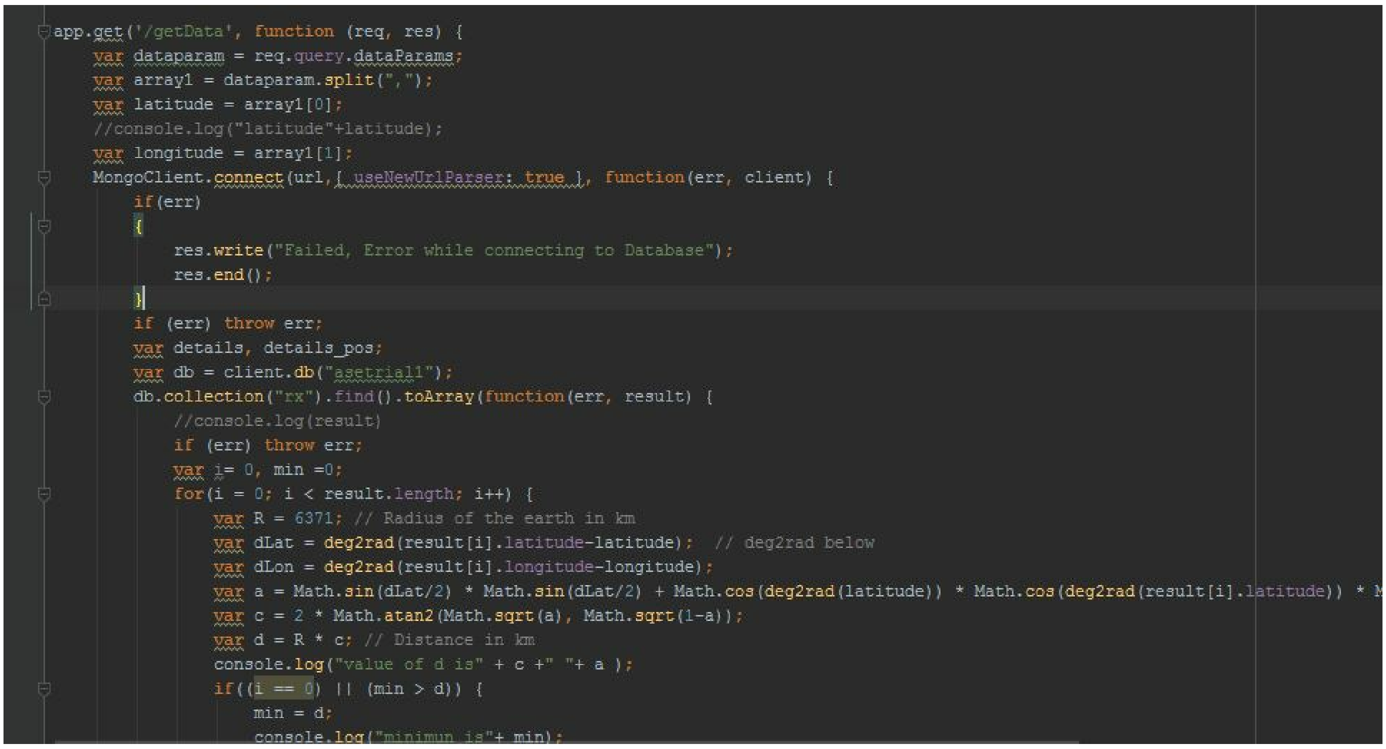
```json
{
  "name": "RX_Savings_Pharmacy",
  "version": "1.0.0",
  "description": "This application gives nearest pharmacy of user",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "Pallavi Desai",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.18.3",
    "cors": "^2.8.4",
    "express": "^4.16.4",
    "mongodb": "^3.1.8",
    "request": "^2.88.0"
  }
}
```

Step 3 : Created server.js and started express server and start listening to port 5000



```javascript
}

var server = app.listen(5000,function () {
    var host = server.address().address
    var port = server.address().port
    console.log("Example app listening at http://%s:%s", host, port)

});
```

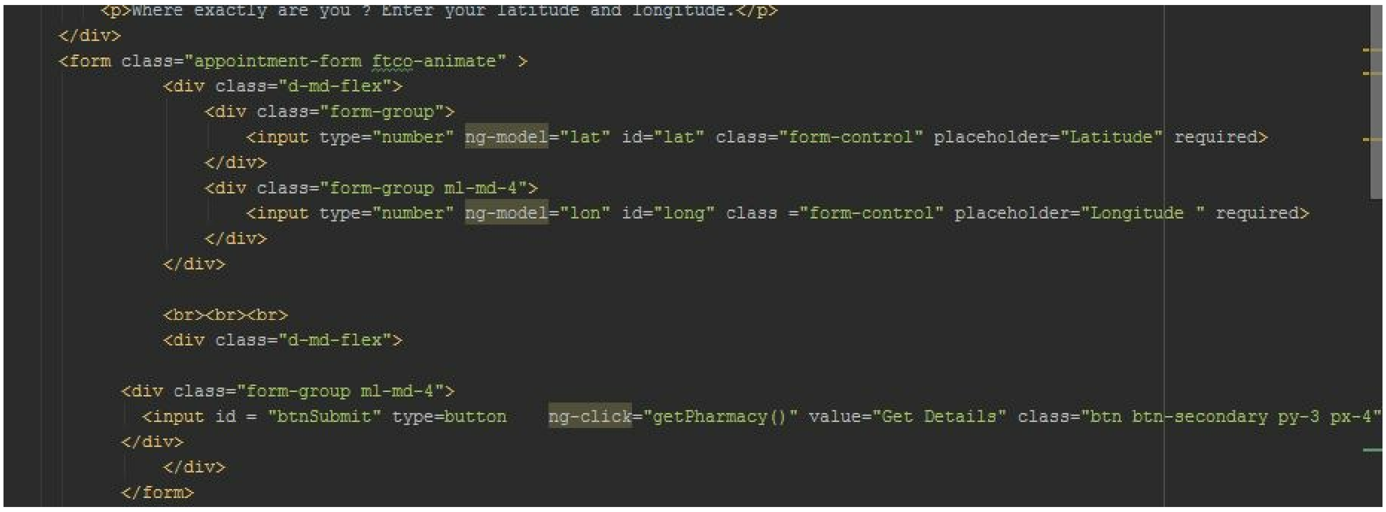
Step 4 : Set request handlers using .get(). The callback accepts one parameter. This contains latitude and longitude separated by comma. The res is the response body and it will return json data which contains names, address and distance in miles of pharmacy.



```
app.get('/getData', function (req, res) {
    var dataparam = req.query.dataParams;
    var array1 = dataparam.split(",");
    var latitude = array1[0];
    //console.log("latitude"+latitude);
    var longitude = array1[1];
    MongoClient.connect(url,{ useNewUrlParser: true }, function(err, client) {
        if(err)
        {
            res.write("Failed, Error while connecting to Database");
            res.end();
        }
        if (err) throw err;
        var details, details_pos;
        var db = client.db("asetrial1");
        db.collection("rx").find().toArray(function(err, result) {
            //console.log(result)
            if (err) throw err;
            var i= 0, min =0;
            for(i = 0; i < result.length; i++) {
                var R = 6371; // Radius of the earth in km
                var dLat = deg2rad(result[i].latitude-latitude);  // deg2rad below
                var dLon = deg2rad(result[i].longitude-longitude);
                var a = Math.sin(dLat/2) * Math.sin(dLat/2) + Math.cos(deg2rad(latitude)) * Math.cos(deg2rad(result[i].latitude)) * M
                var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
                var d = R * c; // Distance in km
                console.log("value of d is" + c +" "+ a );
                if((i == 0) || (min > d)) {
                    min = d;
                    console.log("minimun is"+ min);
```

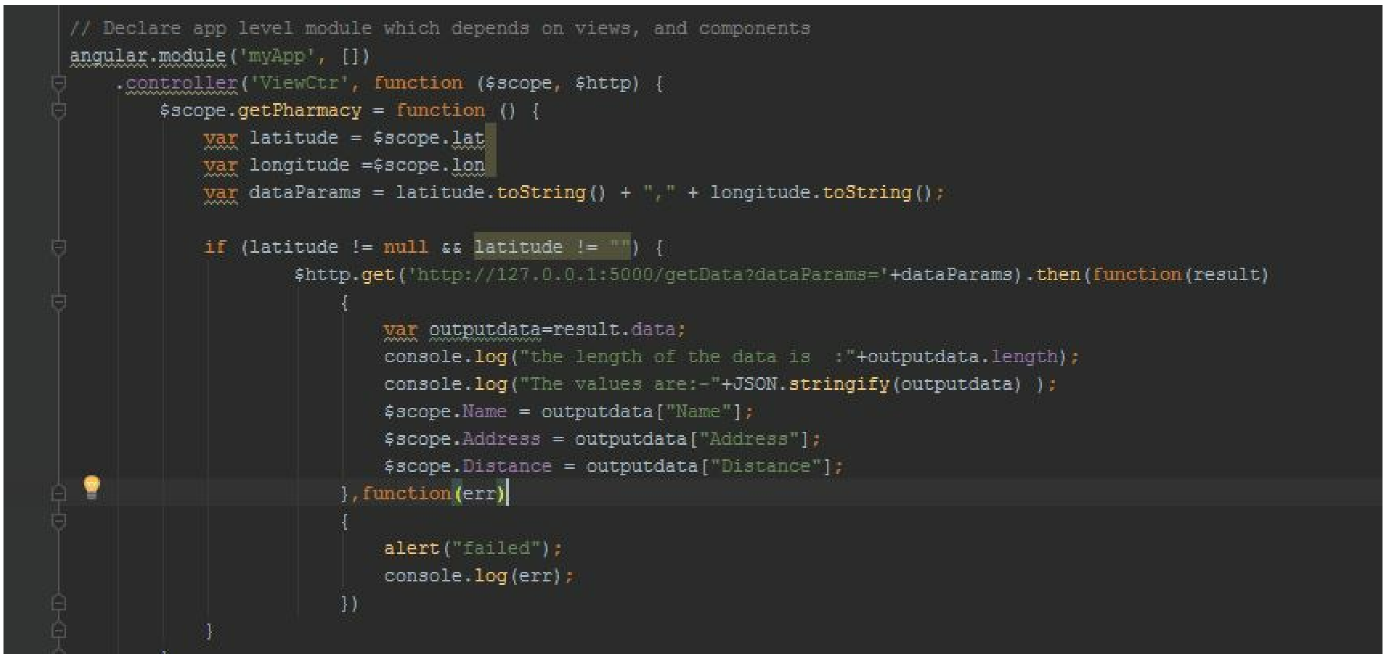
Step 6 : Calculate distance using Haversine formula.

Step 5 : Create front end using HTML5, CSS3 and Bootstrap. And initialize angular app and module.



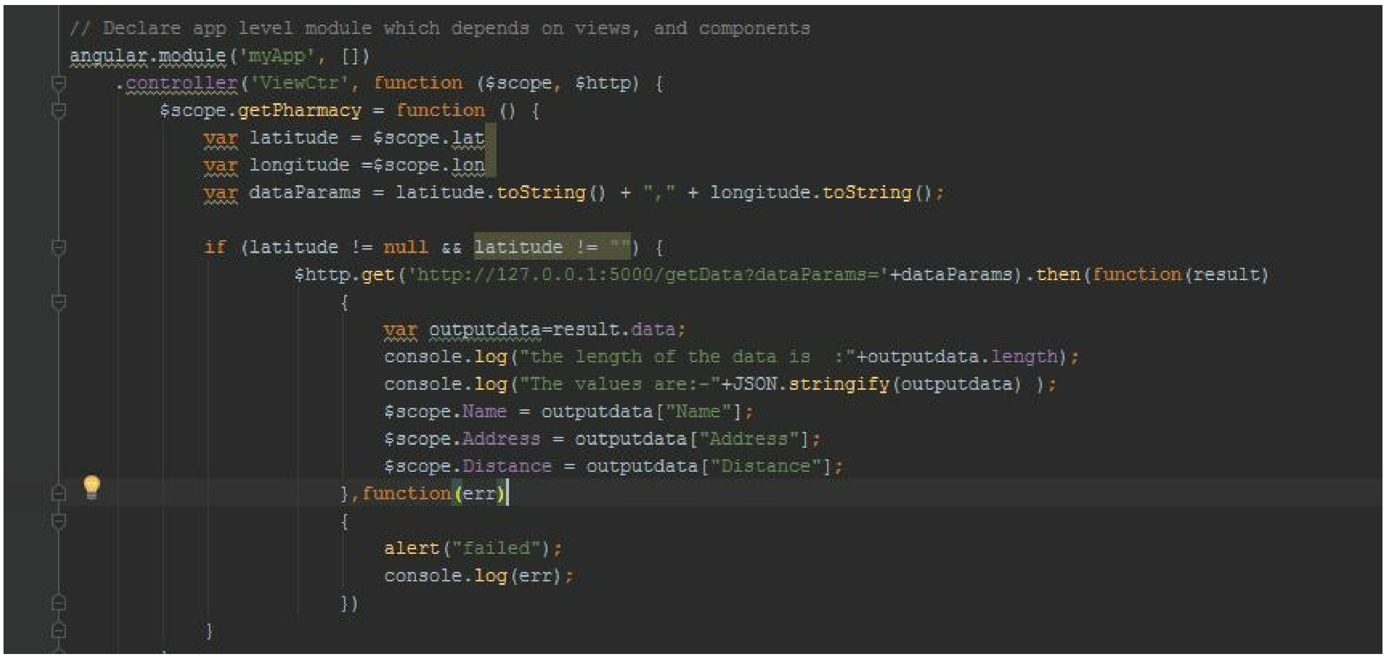
```
        <p>Where exactly are you ? Enter your latitude and longitude.</p>
    </div>
    <form class="appointment-form ftco-animate" >
            <div class="d-md-flex">
                <div class="form-group">
                    <input type="number" ng-model="lat" id="lat" class="form-control" placeholder="Latitude" required>
                </div>
                <div class="form-group ml-md-4">
                    <input type="number" ng-model="lon" id="long" class ="form-control" placeholder="Longitude " required>
                </div>
            </div>

            <br><br><br>
            <div class="d-md-flex">

        <div class="form-group ml-md-4">
          <input id = "btnSubmit" type=button    ng-click="getPharmacy()" value="Get Details" class="btn btn-secondary py-3 px-4"
        </div>
            </div>
        </form>
```

Step 6 : Get longitude and latitude values from text box using ng-model and pass it to API.

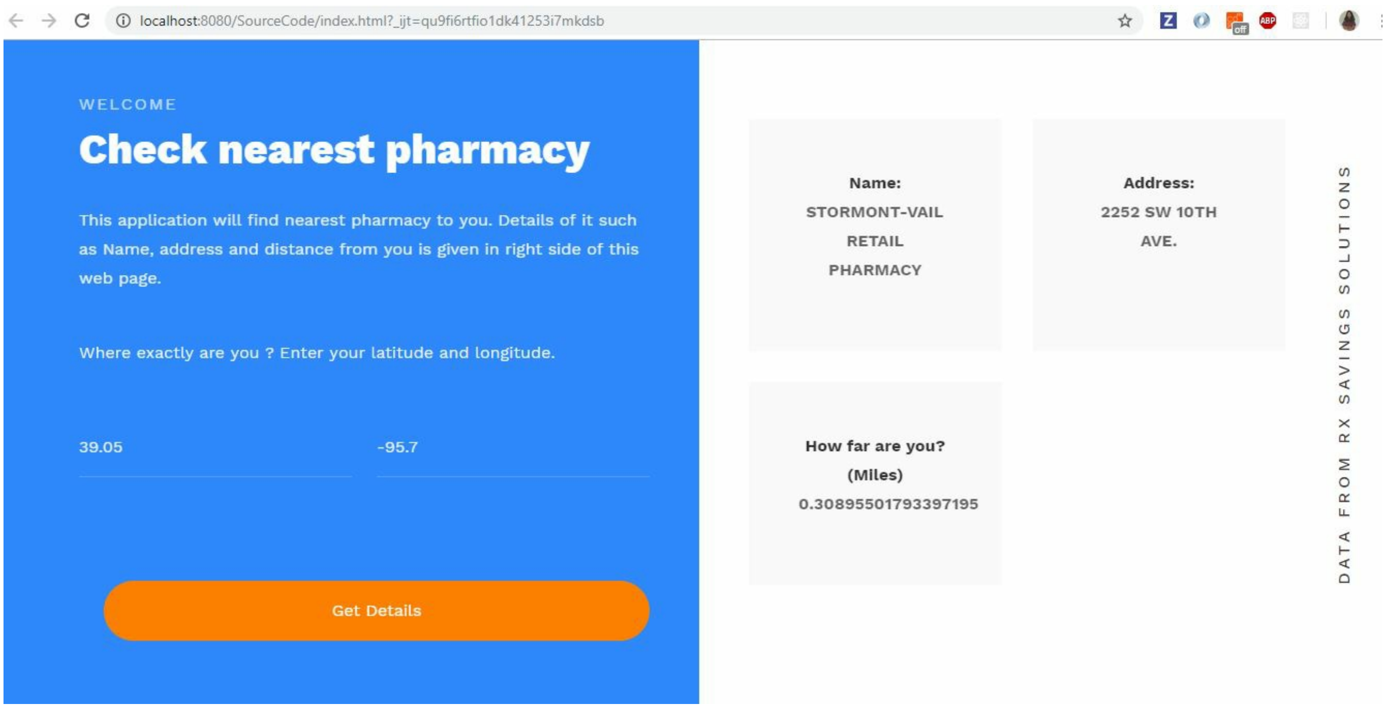


```
// Declare app level module which depends on views, and components
angular.module('myApp', [])
    .controller('ViewCtr', function ($scope, $http) {
        $scope.getPharmacy = function () {
            var latitude = $scope.lat
            var longitude =$scope.lon
            var dataParams = latitude.toString() + "," + longitude.toString();

            if (latitude != null && latitude != "") {
                    $http.get('http://127.0.0.1:5000/getData?dataParams='+dataParams).then(function(result)
                        {
                            var outputdata=result.data;
                            console.log("the length of the data is  :"+outputdata.length);
                            console.log("The values are:-"+JSON.stringify(outputdata) );
                            $scope.Name = outputdata["Name"];
                            $scope.Address = outputdata["Address"];
                            $scope.Distance = outputdata["Distance"];
                        },function(err)
                        {
                            alert("failed");
                            console.log(err);
                        })
            }
```

Step 7: API will return JSON. I then display it in view using {}



```
// Declare app level module which depends on views, and components
angular.module('myApp', [])
    .controller('ViewCtr', function ($scope, $http) {
        $scope.getPharmacy = function () {
            var latitude = $scope.lat
            var longitude =$scope.lon
            var dataParams = latitude.toString() + "," + longitude.toString();

            if (latitude != null && latitude != "") {
                    $http.get('http://127.0.0.1:5000/getData?dataParams='+dataParams).then(function(result)
                        {
                            var outputdata=result.data;
                            console.log("the length of the data is  :"+outputdata.length);
                            console.log("The values are:-"+JSON.stringify(outputdata) );
                            $scope.Name = outputdata["Name"];
                            $scope.Address = outputdata["Address"];
                            $scope.Distance = outputdata["Distance"];
                        },function(err)
                        {
                            alert("failed");
                            console.log(err);
                        })
            }
```


localhost:8080/SourceCode/index.html?_ijt=qu9fi6rtfio1dk41253i7mkdsb

WELCOME

Check nearest pharmacy

This application will find nearest pharmacy to you. Details of it such as Name, address and distance from you is given in right side of this web page.

Where exactly are you ? Enter your latitude and longitude.

39.05 -95.7

Get Details

Name: STORMONT-VAIL RETAIL PHARMACY

Address: 2252 SW 10TH AVE.

How far are you? (Miles) 0.30895501793397195

DATA FROM RX SAVINGS SOLUTIONS

Testing

- Did manual testing.
- Calculated manually using Haversine formula.
- Compared with the result of API.
- Performed 10 tests and all tests passes.

Improvements

- I could include google maps to give directions to user.
- I could check for valid latitude and longitude.
- I could use server-less architecture for scalability. Because number of concurrent requests our API can handle is now limited by processing power of machine where it is deployed. If we run it on cloud you can scale it to handle increasing load.

References:

- https://docs.mlab.com/connecting/
- https://hackernoon.com/19-things-i-learnt-reading-the-nodejs-docs-8a2dcc7f307f
- https://www.w3schools.com/angular/angular_modules.asp
- https://stackoverflow.com/questions/27928/calculate-distance-between-two-latitude-longitude-points-haversine-formula