MONGO DB SETUP

- Depending on your system , install MongoDB in your system
- Go to the installation folder and type below command to start your db server, also give path to store your db data
- mongodb-osx-x86_64-4.0.2/bin/mongod --dbpath YOUR_DB_PATH
- We may use MONGODB COMPASS or any other workbench to access the mongo db in UI
- We may set credentials and port details of our db
- We may create a database using the workbench

NODEJS SETUP

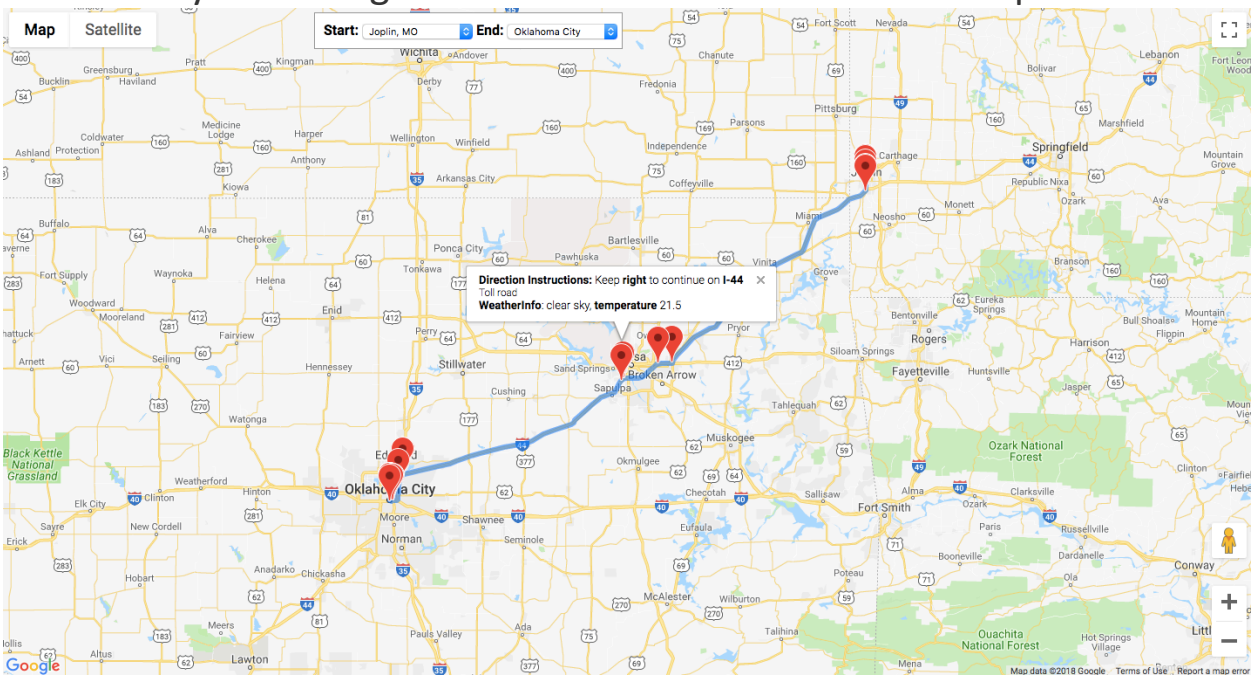
- Depending on your system , install Node.js in your system
- Move to NodeJs folder in the project directory, make sure you have all the dependencies installed, you can follow commands to install dependencies
  // installs express ,mongoose, body-parser dependencies and saves them in package.json file.
- npm I express mongoose body-parser —save
  //install nodemon to make the server start automatically for every change in the Nodejs project.
- npm install -g nodemon

- npm i cors —save //cross app sharing purpose
- To start the server use below command inside NodeJs folder
- nodemon index.js

CLIENT SETUP AND RUN:

- Once you are sure that the mongoDb and NodeJs servers are running properly ,move to client folder , open index.html in any of your browser to start the application

- Give src , destination from the dropdown inputs , the routes along with the way points are displayed.
- Click on any route to get the weather information for that point



Project file details

Client : client folder contains following files

1) Index.html // contains the structure of the web page

2) MapsFunctionality.js // contains the functionality to effect changes on user input

3) mystyles.css. // contains the styling details of our web page

NodeJS: NodeJs folder contains the following files

1) routesController.js // contains the functionality to serve the requests received from the client and send response to the client

2) DirectionsInfo.js // contains the DirectionsInfo object model details

3) WeatherInfo.js //contains the WeatherInfo object model details

4) Package.json. // contains all the dependencies used in this project

5) db.js. // contains functionality to set up connection between backend server and database

6) index.js // handles your app startup, routing and other functions of the application and does require other modules to add functionality