Server

```
//set up for path
const express = require('express')
const app = express()
const port = 3000
const path = require("path")
const cors = require("cors")
let publicPath = path.resolve(__dirname, "public")

app.use(cors())
app.use(express.static(publicPath))
app.listen(port, () => console.log(`Seeking out and listening on... ${port}!`))

//set up the API call
const fetch = require("node-fetch")
const { json } = require('express')
const API_KEY = "3f3cf662c8e9191033ff4958995ef4db"
//API_KEY = process.env.API_KEY;
//debugging
//console.log(API_KEY)

//Pollution API Call
app.get('/air_pollution/:lon/:lat', pollutionData)
async function pollutionData(req, res) {
  let lon = req.params.lon
  let lat = req.params.lat

fetch(`http://api.openweathermap.org/data/2.5/air_pollution/forecast?lat=${lat}&lon=${lo
n}&appid=${API_KEY}`)
  .then(res => res.json())
  .then(json => {
    let result = json
    res.send(result)
  })
}

//Collective Weather API Call
app.get('/forecast/:city', weatherData)
async function weatherData(req, res) {
  let city = req.params.city

fetch(`http://api.openweathermap.org/data/2.5/forecast?q=${city}&appid=${API_KEY}&uni
ts=metric`)
  .then(res => res.json())
  .then(json => {
    //debug to check data call
    //console.log(json)
```

```
        let result = json
        res.send(result)
    })
}
```

Client

```
<link rel="stylesheet" href="style.css">
<div id="app">
    <h1>Hey there!🤗 </h1>
    <h2>Insert the location you would like to find the weather out about!</h2>
    <div>The ouput will be the average of the following 4 day forecast wtih some advice on how to prepare for the expected weather!💡 </div>
        <br>
        <input v-model="location">
        <br>
        <button v-on:click="httpsGet">Get Weather</button>
        <br>
        <br>
        <br>
        <div class="table">
            <table border="5">
                <thead>
                    <tr>
                        <th>Temperature (°C)</th>
                        <th>Wind Speed (m/s)</th>
                        <th>Rainfall (mm)</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>{{ tempResult }}</td>
                        <td>{{ windResult }}</td>
                        <td>{{ rainResult }}</td>
                    </tr>
                </tbody>
                </table>
            <p>{{ umbrella }}</p>
            <p>{{ temp4packing }}</p>
            <p>{{ PM2_5 }}</p>
        </div>
</div>
<script type="module">
    import { createApp } from "https://unpkg.com/vue@3/dist/vue.esm-browser.js"
    createApp({
        data() {
            return {
                location: null,
            umbrella: null,
            temp4packing: null,
            tempResult: null,
            rainResult: null,
            windResult: null,
            PM2_5: null,
          }
        },
        methods: {
          httpsGet(clicked) {
            if (clicked) {
              //call
              fetch(`/forecast/${this.location}`)
              .then((response) => response.json())
              .then((weatherData) => {

                //pollution call, location co-ords fixed to two decimals
                var lon = weatherData["city"]["coord"]["lon"].toFixed(2);
                var lat = weatherData["city"]["coord"]["lat"].toFixed(2);

                fetch(`/air_pollution/${lon}/${lat}`)
                .then((response) => response.json())
                .then((pollutionData) => {

                  // traverse through and find if hum>10
                  var pollutionBoolean = false;
                  for (var count3 = 0; count3 < pollutionData["list"].length; count3++) {
                    if (pollutionData["list"][count3]["components"]["pm2_5"] >= 10) {
                      pollutionBoolean = true;
                    }
                  }
                  //state result
                  if (pollutionBoolean == true) {
                    this.PM2_5  = `It appears that in ${weatherData["city"]["name"]}, the air is
polluted. PM2_5 exceeds a level of 10. For your safety wear a face covering.`;
                  } else {
                    this.PM2_5  = `It appears that in ${weatherData["city"]["name"]} the air is
not polluted. PM2_5 is below the level of 10.  It is not necessary to wear a face covering.`
                  }

                });

                //create values for data
                var temp = 0, rain = 0, wind = 0;
                var umbrellaBoolean = false;

                for (var count = 0; count < 32; count++) {
```

```
                if ("rain" in weatherData["list"][count]) {
                    umbrellaBoolean = true;
                    rain += weatherData["list"][count]["rain"]["3h"];
                }
                temp += weatherData["list"][count]["main"]["temp"];
                wind += weatherData["list"][count]["wind"]["speed"];
            }
            if (umbrellaBoolean = true) {
                this.umbrella = "Looks like rain, you should bring an umbrella!";
            } else {
                this.umbrella = "No rain forecasted!";
            }

            var forecastArr = [];
            var count2 = 0;
            while (forecastArr.length !== 3 && count2 < 32) {
                var locTemp = weatherData["list"][count2]["main"]["temp"];
                if (locTemp < 12 ) {
                    forecastArr.push("Cold🥶");
                }
                else if (locTemp >= 12 && locTemp <= 25) {
                    forecastArr.push("Mild😬");
                }
                else {
                    forecastArr.push("Hot🥵🔥");
                }
                count2++;
            }

            // final results, decimal points adjustment and hourly adjustment
            this.temp4packing = `It is forecasted to be
${forecastArr[0].charAt(0).toUpperCase() + forecastArr[0].slice(1)}, pack the correct
clothes!`;
            this.tempResult = (temp/32).toFixed(1);
            this.rainResult = (rain/32).toFixed(1);
            this.windResult = (wind/32).toFixed(1);

          });
        }
      }
    }
  }).mount("#app")
</script>
```

Styling

```
/* :root {
  -- colour-primary: #020202;
} */
#app {
  /* font-family: arial;
  font-size: 24px;
  margin: 25px;
  width: 350px;
  height: 200px;
  outline: dashed 1px black; */

  display: flex;
  align-items: center;
  flex-direction: column;
 }
.table {
  align-items: center;
  vertical-align: bottom;
}
body{
  background-color: rgb(121, 138, 135);
}
```