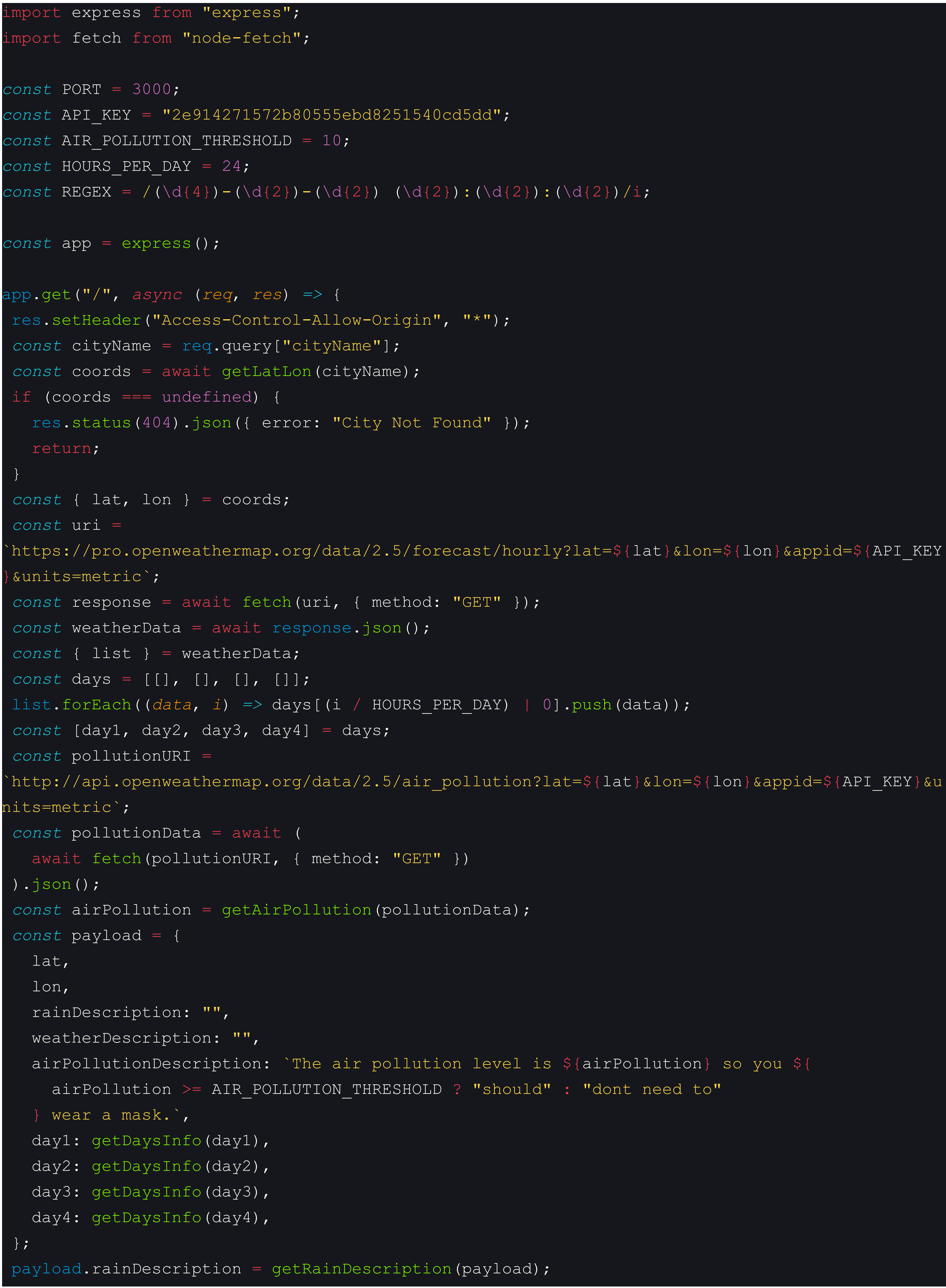
Server Code:



```
import express from "express";
import fetch from "node-fetch";

const PORT = 3000;
const API_KEY = "2e914271572b80555ebd8251540cd5dd";
const AIR_POLLUTION_THRESHOLD = 10;
const HOURS_PER_DAY = 24;
const REGEX = /(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})/i;

const app = express();

app.get("/", async (req, res) => {
 res.setHeader("Access-Control-Allow-Origin", "*");
 const cityName = req.query["cityName"];
 const coords = await getLatLon(cityName);
 if (coords === undefined) {
   res.status(404).json({ error: "City Not Found" });
   return;
 }
 const { lat, lon } = coords;
 const uri =
`https://pro.openweathermap.org/data/2.5/forecast/hourly?lat=${lat}&lon=${lon}&appid=${API_KEY
}&units=metric`;
 const response = await fetch(uri, { method: "GET" });
 const weatherData = await response.json();
 const { list } = weatherData;
 const days = [[], [], [], []];
 list.forEach((data, i) => days[(i / HOURS_PER_DAY) | 0].push(data));
 const [day1, day2, day3, day4] = days;
 const pollutionURI =
`http://api.openweathermap.org/data/2.5/air_pollution?lat=${lat}&lon=${lon}&appid=${API_KEY}&u
nits=metric`;
 const pollutionData = await (
   await fetch(pollutionURI, { method: "GET" })
 ).json();
 const airPollution = getAirPollution(pollutionData);
 const payload = {
   lat,
   lon,
   rainDescription: "",
   weatherDescription: "",
   airPollutionDescription: `The air pollution level is ${airPollution} so you ${
     airPollution >= AIR_POLLUTION_THRESHOLD ? "should" : "dont need to"
   } wear a mask.`,
   day1: getDaysInfo(day1),
   day2: getDaysInfo(day2),
   day3: getDaysInfo(day3),
   day4: getDaysInfo(day4),
 };
 payload.rainDescription = getRainDescription(payload);
 payload.weatherDescription = getWeatherDescription(payload);
 res.json(payload);
});

app.listen(PORT);

const getLatLon = async (cityName) => {
 const url =
`http://api.openweathermap.org/geo/1.0/direct?q=${cityName}&appid=${API_KEY}&units=metric`;
 const res = await fetch(url, { method: "GET" });
 const body = await res.json();
 if (body.length === 0) return undefined;
 const { lat, lon } = body[0];
 return { lat, lon };
};

const getRainData = (hourlyData) =>
 hourlyData.map(({ rain }) => (rain ? rain["1h"] : 0));

const getTempData = (hourlyData) =>
 hourlyData.map(({ main }) => main.feels_like.toFixed(2));

const getWindspeedData = (hourlyData) =>
 hourlyData.map(({ wind: { speed } }) => speed);

const getTodaysAverageTemperature = (hourlyData) => {
 const avgMin =
   hourlyData.reduce((prev, { main }) => prev + main.temp_min, 0) /
   hourlyData.length;
 const avgMax =
   hourlyData.reduce((prev, { main }) => prev + main.temp_max, 0) /
   hourlyData.length;
 return (avgMax + avgMin) / 2;
};

const getAirPollution = (data) => data.list[0].components.pm2_5;

const getWeekDay = (day) => {
 const date = day[0].dt_txt;
 const times = REGEX.exec(date)?.slice(1).map(Number);
 times[1]--;
 return new Date(...times).toLocaleDateString(undefined, { weekday: "long" });
};

const getDaysInfo = (day) => {
 const rain = getRainData(day);
 const temps = getTempData(day);
 const avgTemp = getTodaysAverageTemperature(day);
 const windSpeeds = getWindspeedData(day);
 const weekday = getWeekDay(day);
 const tempDescription =
   avgTemp <= 12 ? "Cold" : avgTemp <= 24 ? "Mild" : "Hot";
 return { rain, temps, avgTemp, windSpeeds, tempDescription, weekday };
};

const stringify = (input) => {
 if (input.length === 0) return "";
 if (input.length === 1) return input[0];
 const res = input.slice(0, input.length - 2).join(", ");
 return `${res}${res ? ", " : ""}${input.at(-2)} and ${input.at(-1)}`;
};

const getRainDescription = ({ day1, day2, day3, day4 }) => {
 const days = [day1, day2, day3, day4];
 const rains = days.map(({ rain }) => rain);
 const weekdays = days.map(({ weekday }) => weekday);
 const rainDays = weekdays.filter((_, i) => rains[i].some((n) => n > 0));
 if (rainDays.length !== 0) {
   return `It's going to rain on ${stringify(
     rainDays
   )} so you should pack an umbrella.`;
 } else return "It's not going to rain so you don't need to pack an umbrella.";
};

const getTempDays = (weekdays, days, tempDescription) =>
 weekdays.filter((_, i) => days[i].tempDescription === tempDescription);

const getTempDescription = (days, temp) =>
 days.length ? `${temp.toLowerCase()} on ${stringify(days)}` : "";

const getWeatherDescription = ({ day1, day2, day3, day4 }) => {
 const days = [day1, day2, day3, day4];
 const weekdays = days.map(({ weekday }) => weekday);
 const coldDays = getTempDays(weekdays, days, "Cold");
 const mildDays = getTempDays(weekdays, days, "Mild");
 const hotDays = getTempDays(weekdays, days, "Hot");
 const coldDesc = getTempDescription(coldDays, "Cold");
 const mildDesc = getTempDescription(mildDays, "Mild");
 const hotDesc = getTempDescription(hotDays, "Hot");
 const descs = [coldDesc, mildDesc, hotDesc];
 const finalDescs = descs.filter((s) => !!s);
 const packingConditions = ["cold", "mild", "hot"].filter(
   (_, i) => !!descs[i]
 );
 return `It's going to be ${stringify(
   finalDescs
 )} so should pack for ${stringify(packingConditions)} weather.`;
};
```

Front End Code:

```html
<!DOCTYPE html>
<html lang="en">

<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <link rel="stylesheet" href="style.css">
 <title>Weather App</title>
</head>

<body>
 <script type="module">
   import { createApp, reactive, ref } from 'https://unpkg.com/vue@3/dist/vue.esm-browser.js'

   createApp({
     setup() {
       const weatherData = ref(undefined);
       const src = ref(undefined)
       const onEnter = async ({ target: { value: cityName } }) => {
         const response = await (await fetch(`http://localhost:3000?cityName=${cityName}`,
           { method: "GET", "Content-Type": "application/json" })).json()
         weatherData.value = response;
         if (!response.error) {
           const { lat, lon } = weatherData.value;
           src.value =
`https://maps.google.com/maps?q=${lat},${lon}&t=&z=13&ie=UTF8&iwloc=&output=embed`
         }
       }
       const times = [...new Array(24).keys()]
         .map(time => `${(time + "").padStart(2, "0")}:00`);
       return { weatherData, oninput, onEnter, times, src }
     },
   }).mount('#app')
 </script>

 <div id="app">
   <h1>Enter the City Name</h1>
   <div class="input">
     <input placeholder="City Name" @input="oninput" type="text" @keyup.enter="onEnter" />
   </div>
   <div v-if=weatherData style="display: none;" :id=`weather-data`>
     <div id="error-message" v-if="weatherData.error">
       Uhm, that city doesn't exist. Please try again
     </div>
     <div v-else id="weather-info">
       <ul>
         <li>{{ weatherData.rainDescription }}</li>
         <li>{{ weatherData.weatherDescription }}</li>
         <li>{{ weatherData.airPollutionDescription }}</li>
```

```
      </ul>
      <div class="weather-table">
        <div>
          <h3>{{ weatherData.day1.weekday }}</h3>
          <div class="table">
            <div class="label">Rain</div>
            <div class="row-1" v-for="rain in weatherData.day1.rain">{{ rain }}</div>
            <div class="label">Temperature</div>
            <div class="row-2" v-for="temp in weatherData.day1.temps">{{ temp }}</div>
            <div class="label">Wind Speed</div>
            <div class="row-3" v-for="speed in weatherData.day1.windSpeeds">{{ speed }}</div>
            <div class="label-last">Time</div>
            <div class="time row-4" v-for="time in times">{{ time }}</div>
          </div>
        </div>
        <div>
          <h3>{{ weatherData.day2.weekday }}</h3>
          <div class="table">
            <div class="label">Rain</div>
            <div class="row-1" v-for="rain in weatherData.day2.rain">{{ rain }}</div>
            <div class="label">Temperature</div>
            <div class="row-2" v-for="temp in weatherData.day2.temps">{{ temp }}</div>
            <div class="label">Wind Speed</div>
            <div class="row-3" v-for="speed in weatherData.day2.windSpeeds">{{ speed }}</div>
            <div class="label-last">Time</div>
            <div class="time row-4" v-for="time in times">{{ time }}</div>
          </div>
        </div>
        <div>
          <h3>{{ weatherData.day3.weekday }}</h3>
          <div class="table">
            <div class="label">Rain</div>
            <div class="row-1" v-for="rain in weatherData.day3.rain">{{ rain }}</div>
            <div class="label">Temperature</div>
            <div class="row-2" v-for="temp in weatherData.day3.temps">{{ temp }}</div>
            <div class="label">Wind Speed</div>
            <div class="row-3" v-for="speed in weatherData.day3.windSpeeds">{{ speed }}</div>
            <div class="label-last">Time</div>
            <div class="time row-4" v-for="time in times">{{ time }}</div>
          </div>
        </div>
        <div>
          <h3>{{ weatherData.day4.weekday }}</h3>
          <div class="table">
            <div class="label">Rain</div>
            <div class="row-1" v-for="rain in weatherData.day4.rain">{{ rain }}</div>
            <div class="label">Temperature</div>
            <div class="row-2" v-for="temp in weatherData.day4.temps">{{ temp }}</div>
            <div class="label">Wind Speed</div>
            <div class="row-3" v-for="speed in weatherData.day4.windSpeeds">{{ speed }}</div>
            <div class="label-last">Time</div>
            <div class="time row-4" v-for="time in times">{{ time }}</div>
```

```html
          </div>
        </div>
      </div>
      <div id="map" style="width: 100%;">
        <div class="mapouter">
          <div class="gmap_canvas">
            <iframe width="747" height="360px" id="gmap_canvas" :src="src" frameborder="0"
scrolling="no"
              marginheight="0" marginwidth="0">
            </iframe>
            <a href="https://putlocker-is.org"></a><br>
            <a href="https://www.embedgooglemap.net"></a>
          </div>
        </div>
      </div>
    </div>
  </div>
 </div>
</body>

</html>
```