server.js

```
const { error } = require('console')
const express = require('express')
require('dotenv').config()
const app = express()
const port = 3000
const path=require("path")
let publicPath= path.resolve(__dirname,"public")
app.use(express.static(publicPath))
app.get('/forecast/:city', fetchWeather)
app.get('/air_pollution/forecast/:lat/:lon', getAirPollution)
app.listen(port, () => console.log(`Example app listening on port ${port}!`))



const apiKey = process.env.MY_WEATHER_API_KEY;

// fetching openweathermap API
function fetchWeather (request, response)
{
   const city_name = request.params.city;

fetch('https://api.openweathermap.org/data/2.5/forecast?q='+city_name+'&units=metric&
appid='+apiKey)
   .then(response => response.json())
   .then(json => {
      console.log(json);
      let result = json;
      response.send(result);
   })

}

function getAirPollution (request, response)
{
   const lat = request.params.lat;
   const lon = request.params.lon;
   console.log(request)
   console.log(lon)

fetch('http://api.openweathermap.org/data/2.5/air_pollution/forecast?lat='+lat+'&lon='+lon
+'&appid='+apiKey)
   .then(response => response.json())
   .then(json => {
      console.log(json);
      let result = json;
      response.send(result);
```

```
    })
}
```

styles.css

```
body {

  background-color: lightblue;
  color: black;
  margin-right: auto;
  margin-left: auto;


  }
  h1 {
    color: #023E8A;
    font-family: Georgia, 'Times New Roman', Times, serif ;
    justify-content: center;
  }

  .weather-activities {
    justify-content: right;
  }



table, th, td {
  border:1px solid black;
  margin-left: auto;
  margin-right: auto;
}

.table-container {
  display: flex; /* Display table and image side by side */
  align-items: center; /* Vertically center the content */
}

/* Style for the image */
.custom-image {

  height: auto; /* Maintain the aspect ratio */
  margin-left: 20px; /* Add spacing between the table and image */
  border: 5px solid #555;
}
```

index.html

```
<!-- development version, includes helpful console warnings -->
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<link
  rel="stylesheet"
  href="//cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.4.1/semantic.min.css"
/>
<link rel="stylesheet" href="styles.css">
<div id="app">

<h1>Welcome to the Best Weather App</h1>
You'll never be caught out in the rain, that's not a bold claim!
<br>

<div class="ui icon input" id="button_city">
<input v-model="city" placeholder="search for a city"> <br><br>
<i class="search icon"></i>
</div>

<div class="ui icon input" id="button_country">
<input v-model="country" placeholder="add the country code"> <br><br>
<i class="search icon"></i>
</div>

<button class="ui primary button" v-on:click="GetWeatherForecast" tabindex="0">Show me
the weather!</button>

<span v-if="history.length>0">

<body>


<hr>

<div v-if="raining > 0"> We predict spells of rain - remember to pack an umbrella!</div>

<div v-if="hot > 0"> Hot - pack for temperatures of 23 degrees and higher on some
days</div>
<div v-if="mild > 0"> Mild - it will be between 13 and 23 degrees on some days</div>
<div v-if="cold > 0"> Cold - it will be below 13 on some days</div>

<div v-if="pollution_mask > 0"> Wear a mask as the PM2.5 level is bad (above 10)</div>
<div v-if="pollution_mask == 0">No need to wear a mask, PM2.5 level (fine particles matter)
is good (below 10)</div>
```

```
<h2>The weather for {{city}}, {{country}}</h2>

<div class="table-container">
<table class="ui very basic collapsing celled table">
 <thead>
  <tr><th>Day</th>
  <th>Forecast</th>
 </tr></thead>
 <tbody>
  <tr>
   <td>
    <h4 class="ui image header">
     <div class="content">
      Day 1
     </div>
    </div>
   </h4></td>
   <td>
    {{sortedWeatherDescriptions[0]}}

    <!-- Unique Feature: Recommending activities based on the weather forecast for each
day -->
    <div class="weather-activities">
     Recommended Activity Based on Weather:<br>
  <span v-if="sortedWeatherDescriptions[0].includes('rain') ||
sortedWeatherDescriptions[0].includes('hail')">Go to the cinema or play games
inside</span>
  <span v-else-if="sortedWeatherDescriptions[0].includes('cloud') ||
sortedWeatherDescriptions[0].includes('overcast')">Go for a walk<br>No need for
suncream!</span>
  <span v-else-if="sortedWeatherDescriptions[0].includes('sun') ||
sortedWeatherDescriptions[0].includes('clear')">Go to the beach and get ice-
cream<br>Remember suncream!</span>
  <span v-else>No specific recommendation for this weather<br>Consult authorities</span>
    </div>
    </div>
   </td>
  </tr>
  <tr>
   <td>
    <h4 class="ui image header">

     <div class="content">
      Day 2
     </div>
    </div>
```

```
    </h4></td>
    <td>
      {{sortedWeatherDescriptions[1]}}
      <div class="weather-activities">
       Recommended Activity Based on Weather:<br>
  <span v-if="sortedWeatherDescriptions[1].includes('rain') ||
sortedWeatherDescriptions[1].includes('hail')">Go to the cinema or play games
inside</span>
  <span v-else-if="sortedWeatherDescriptions[1].includes('cloud') ||
sortedWeatherDescriptions[1].includes('overcast')">Go for a walk <br>No need for
suncream!</span>
  <span v-else-if="sortedWeatherDescriptions[1].includes('sun') ||
sortedWeatherDescriptions[1].includes('clear')">Go to the beach and get ice-
cream<br>Remember suncream!</span>
  <span v-else>No specific recommendation for this weather<br>Consult authorities</span>
      </div>
    </td>
  </tr>
  <tr>
    <td>
      <h4 class="ui image header">

        <div class="content">
          Day 3
        </div>
      </div>
    </h4></td>
    <td>
     {{sortedWeatherDescriptions[2]}}
      <div class="weather-activities">
       Recommended Activity Based on Weather:<br>
  <span v-if="sortedWeatherDescriptions[2].includes('rain') ||
sortedWeatherDescriptions[2].includes('hail')">Go to the cinema or play games
inside</span>
  <span v-else-if="sortedWeatherDescriptions[2].includes('cloud') ||
sortedWeatherDescriptions[2].includes('overcast')">Go for a walk <br>No need for
suncream!</span>
  <span v-else-if="sortedWeatherDescriptions[2].includes('sun') ||
sortedWeatherDescriptions[2].includes('clear')">Go to the beach and get ice-
cream<br>Remember suncream!</span>
  <span v-else>No specific recommendation for this weather<br>Consult authorities</span>
      </div>
    </td>
  </tr>
  <tr>
    <td>
      <h4 class="ui image header">
```

```
        <div class="content">
          Day 4
        </div>
       </div>
     </h4></td>
     <td>
       {{sortedWeatherDescriptions[3]}}
       <div class="weather-activities">
        Recommended Activity Based on Weather:<br>
   <span v-if="sortedWeatherDescriptions[3].includes('rain') ||
sortedWeatherDescriptions[3].includes('hail')">Go to the cinema or play games
inside</span>
   <span v-else-if="sortedWeatherDescriptions[3].includes('cloud') ||
sortedWeatherDescriptions[3].includes('overcast')">Go for a walk <br>No need for
suncream!</span>
   <span v-else-if="sortedWeatherDescriptions[3].includes('sun') ||
sortedWeatherDescriptions[3].includes('clear')">Go to the beach and get ice-
cream<br>Remember suncream!</span>
   <span v-else>No specific recommendation for this weather<br>Consult authorities</span>
      </div>

     </td>
   </tr>
   <tr>
     <td>
       <h4 class="ui image header">
         <div class="content">
           Day 5
         </div>
       </div>
     </h4></td>
     <td>
       {{sortedWeatherDescriptions[4]}}
       <div class="weather-activities">
        Recommended Activity Based on Weather:<br>
   <span v-if="sortedWeatherDescriptions[4].includes('rain') ||
sortedWeatherDescriptions[4].includes('hail')">Go to the cinema or play games
inside</span>
   <span v-else-if="sortedWeatherDescriptions[4].includes('cloud') ||
sortedWeatherDescriptions[4].includes('overcast')">Go for a walk <br>No need for
suncream!</span>
   <span v-else-if="sortedWeatherDescriptions[4].includes('sun') ||
sortedWeatherDescriptions[4].includes('clear')">Go to the beach and get ice-
cream<br>Remember suncream!</span>
   <span v-else>No specific recommendation for this weather<br>Consult authorities</span>
      </div>
```

```
    </td>
   </tr>
  </tbody>
</table>
<img
src="https://t3.ftcdn.net/jpg/02/11/52/42/360_F_211524227_Ett8aboQvVnROAFtqu3S1pW
99Y3Th9vm.jpg" alt="Your Image" class="custom-image">
</div>


<table style="width:50%" class="metric-table">
          <tr>
             <th> Metric</th>
             <th> Day 1</th>
             <th> Day 2</th>
             <th> Day 3</th>
             <th> Day 4</th>
             <th> Day 5</th>

          </tr>
          <tr>
             <td>Average Daily Temperature (°C) </td>
             <td>{{temps[0]}}</td>
             <td>{{temps[1]}}</td>
             <td>{{temps[2]}}</td>
             <td>{{temps[3]}}</td>
             <td>{{temps[4]}}</td>
          </tr>
          <tr>
             <td>Average Daily Windspeed (nautical m/h)</td>
             <td>{{winds[0]}}</td>
             <td>{{winds[1]}}</td>
             <td>{{winds[2]}}</td>
             <td>{{winds[3]}}</td>
             <td>{{winds[4]}}</td>
          </tr>
          <tr>
             <td> Average Daily Humidity (g/m^3) </td>
            <td>{{humids[0]}}</td>
            <td>{{humids[1]}}</td>
            <td>{{humids[2]}}</td>
            <td>{{humids[3]}}</td>
            <td>{{humids[4]}}</td>
          </tr>
          <tr>
          <td> Daily Rainfall (mm) </td>
             <td>{{rainfall[0]}}</td>
```

```
            <td>{{rainfall[1]}}</td>
            <td>{{rainfall[2]}}</td>
            <td>{{rainfall[3]}}</td>
            <td>{{rainfall[4]}}</td>
          </tr>
        <td> Daily Concentration of PM2.5 (µg/m3) </td>
            <td>{{tablePollution[0]}}</td>
            <td>{{tablePollution[1]}}</td>
            <td>{{tablePollution[2]}}</td>
            <td>{{tablePollution[3]}}</td>
            <td>{{tablePollution[4]}}</td>
          </tr>

      </table>

    </body>

  </div>

</div>
<script>
  var app = new Vue({
    el:'#app',
    data:{
    city: '',
    country: '',
    descriptions: [],
    forecastList: [],
    sortedWeatherDescriptions: [],
    DayOne: [],
    DayTwo: [],
    DayThree: [],
    DayFour: [],
    DayFive: [],
    packForWeather: [],
    raining: -1,
    hot: -1,
    mild: -1,
    cold: -1,
    temps : [],
    winds: [],
    humids: [],
    rainfall: [],
    rainfallMeasure: [],
    Pollution: [],
    pollution_mask: 0,
    tablePollution: [],
```

```
    history: [],
    weather: []},
    methods:{
    GetWeatherForecast : getForecast, sortWeatherDescriptions,
    packingForWeather, getPollution, getRainfall, checkPollution }
  })


  function getForecast (){
        console.log("function getWeather API called")
        let prom = fetch("forecast/"+this.city + ((this.country == "") ? "" : ",") + this.country)
        prom.then( response => response.json())
        .then (response =>
        {
          let forecast = response;
          console.log(forecast);
          let longitude = response["city"]["coord"]["lon"];
          let latitude = response["city"]["coord"]["lat"];

          console.log(longitude, "this is longitude");
          console.log(latitude, "this is lat");

            let weatherList = response["list"]
            console.log(weatherList)
            app.rainfallMeasure = response["list"]
            app.packForWeather = response.list //make weather forecast hold list array of
weather reports \
            console.log(app.packForWeather)
            app.history.push(response.result);
            sortWeatherDescriptions(weatherList);

            packingForWeather();

            console.log(app.rainfallMeasure)
            getRainfall(weatherList);
            getPollution(latitude, longitude);

    })}

    function sortWeatherDescriptions (weatherList)
    {
      let i = 0;
      while (i <= 4)
      {
        app.sortedWeatherDescriptions.push(weatherList[i]["weather"][0]["description"]);
         i++;
      }
```

```
    }

    function packingForWeather ()
    {
      let k = 0;
    //find out if raining over next 5 days
          let i=0;

          while((i < app.packForWeather.length) && (app.raining==-1)){
            let temp = app.packForWeather[i].weather[0].main ;
            if(temp === "Rain"){
               app.raining = 1;
               }

            i++;
          }

          // cold, warm or hot temperature for user notification
          let j=0;
          let minTemp = app.packForWeather[0]["main"]["temp"] //set an initial min
          let maxTemp = app.packForWeather[1]["main"]["temp"] //set an initial max
          for( j=0; j < app.packForWeather.length; j++)
          {
            if (app.packForWeather[j]["main"]["temp"] < minTemp) minTemp =
app.packForWeather[j]["main"]["temp"];
            if (app.packForWeather[j]["main"]["temp"] > maxTemp) maxTemp =
app.packForWeather[j]["main"]["temp"];
          }

          if(minTemp < 13) app.cold = 1;
          else if(minTemp <= 23) app.mild = 1;
          if(minTemp > 23) app.hot = 1;

          i = 3 //start at reading 3 for day 0, increase by 8 readings to get the next midday
reading
          j = 0
          let tempEachDay = []
          for(i = 3; i < app.packForWeather.length; i+=8)
          {
            tempEachDay[j]= ((app.packForWeather[i]["main"]["temp"] +
app.packForWeather[i+1]["main"]["temp"])/2).toFixed(2)
            j++
          }
          app.temps = tempEachDay



          i=3
```

```
            j=0
            let windEachDay = []
            for(i = 3; i < app.packForWeather.length; i+=8)
            {
               windEachDay[j]= ((app.packForWeather[i]["wind"]["speed"] +
app.packForWeather[i+1]["wind"]["speed"])/2).toFixed(2)
               j++
            }
            app.winds = windEachDay

            i=3
            j=0
            let humidEachDay = []
            for(i = 3; i < app.packForWeather.length; i+=8)
            {
               humidEachDay[j]= ((app.packForWeather[i]["main"]["humidity"] +
app.packForWeather[i+1]["main"]["humidity"])/2).toFixed(2)
               j++
            }
            app.humids=humidEachDay

      }

      function getPollution (latitude, longitude)
      {
       console.log("function getPollution API called")
       console.log(latitude)
       console.log(longitude)
       let prom = fetch("air_pollution/forecast/"+latitude+"/"+longitude)
       prom.then(response => response.json())
       .then( response =>
       {
         console.log(response)
         let pollutionList = response
         console.log(pollutionList)
         app.Pollution = response.list
         console.log(app.Pollution)
         //console.log(app.Pollution["list"][0]["main"]["aqi"]) // aqi
         app.history.push(response.result);
       checkPollution();
      })}

      function checkPollution ()
      {
       let i = 3;
       let j = 0;
       let k = 0;
```

```
        let pollutionLevelByDay = [];
        let warning = false;

        console.log(app.Pollution[0]["components"]["pm2_5"])
        for(i = 3; i < app.Pollution.length; i+=8)
            {
               pollutionLevelByDay[j]= ((app.Pollution[i]["components"]["pm2_5"] +
app.Pollution[i+1]["components"]["pm2_5"])/2).toFixed(2)
               j++
            }
            console.log("pollution: " ,pollutionLevelByDay)
            app.tablePollution = pollutionLevelByDay


            for ( k = 0; k < 5; k++)
            {
              if (pollutionLevelByDay[k] > 10)
              {
                warning = true;
              }
            }
          // Set the indicator to wear a mask using Vue directives
          app.pollution_mask = warning ? 1 : 0;

      }

      function getRainfall (weatherList)
      {

          app.rainfall = [];

          for (let i = 0; i < 5 ; i++) {
            let totalRain = 0;
            let count = 0;

              for (let j = i * 8; j < (i + 1) * 8; j++) { // uodates every 3 hours, increase by 8 hours to
get next reading
                if (typeof app.rainfallMeasure[j] === 'object') {
                  if (app.rainfallMeasure[j]["rain"] && app.rainfallMeasure[j]["rain"]['3h']) {
                    // Check if the "rain" field is present and has a value
                    totalRain += app.rainfallMeasure[j]["rain"]['3h'];
                    count++;
                  }
                }
              }

            // Calculate the average daily rainfall
```

```
            if (count > 0) {
              app.rainfall.push((totalRain / count).toFixed(2)); // Assuming 8 data points per day
            } else {
              app.rainfall.push(0);
            }
          }
        }


</script>
```

package.json

```
{
  "name": "assignment-1",
  "version": "1.0.0",
  "description": "Internet Apps As1",
  "main": "server.js",
  "scripts": {
    "start": "node server.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "dotenv": "^16.3.1",
    "express": "^4.18.2",
    "gulp": "^4.0.2",
    "semantic-ui-vue": "^0.11.0"
  }
}
```

.env

MY_WEATHER_API_KEY = 21cb73a0f86d4380fd6caa950a89b744