```
/*
For my assignment, I have used axios to ge the data from openweather API.
The data is stored in the 'data' variable. That data is passed to convert function
which processed and converts the data. Then, the server sends final_data back as
a response
*/

//importing libraries
const express = require('express');
const axios = require('axios');
var moment = require('moment');

const app = express();
const PORT = process.env.PORT || 5000;

app.listen(PORT, () => console.log(`Server started at port ${PORT}`));

API_KEY = '59c3c3690cc24e1ecda415ebe6b31871';

// allowing CORS to open it up for universal JavaScript/browser access
app.use(function (req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");
  next();
});

// The function that gets data from the API. If there is an error,
// (eg. city not found) an error message is returned which is handled
// by the frontend
app.get('/api/data/:location', (req, res) => {

  let url = `https://api.openweathermap.org/data/2.5/forecast?q=${req.params.location}&appid=${API_KEY}&units=metric`

  axios.get(url)
    .then(function (response) {

      var data = response.data;
      let final_data = convert(data);
      res.send(final_data);

    })
    .catch(function (error) {
      res.send(JSON.stringify({ error: "city does not exist" }));
    })


});

// function that takes data from API and converts it to a format usable by frontend

```

```
// the converted response sent back for a get request for city "Dublin,IE"
/*
{
  "city": "Dublin",
  "country": "IE",
  "weather": [
    {
      "date": "Monday, November 9th",
      "max_temp": 13.47,
      "min_temp": 11.34,
      "wind_speed": 4.1,
      "rain": true,
      "total_rain": "0.12",
      "expected_rainfall": {
        "15:00:00": 0.12
      }
    },
    {
      "date": "Tuesday, November 10th",
      "max_temp": 13.37,
      "min_temp": 9.71,
      "wind_speed": 4.51,
      "rain": true,
      "total_rain": "0.32",
      "expected_rainfall": {
        "12:00:00": 0.1,
        "15:00:00": 0.22
      }
    },
    {
      "date": "Wednesday, November 11th",
      "max_temp": 12.74,
      "min_temp": 7.11,
      "wind_speed": 8.92,
      "rain": true,
      "total_rain": "18.48",
      "expected_rainfall": {
        "06:00:00": 0.11,
        "09:00:00": 0.17,
        "12:00:00": 0.81,
        "15:00:00": 3.06,
        "18:00:00": 13.77,
        "21:00:00": 0.56
      }
    },
    {
      "date": "Thursday, November 12th",
      "max_temp": 10.61,
      "min_temp": 4.79,
      "wind_speed": 7.01,
      "rain": true,
      "total_rain": "0.35",
      "expected_rainfall": {
        "21:00:00": 0.35
      }
```

```
      }
    },
    {
      "date": "Friday, November 13th",
      "max_temp": 9.48,
      "min_temp": 6.3,
      "wind_speed": 9.93,
      "rain": true,
      "total_rain": "2.88",
      "expected_rainfall": {
        "00:00:00": 2.88
      }
    },
    {
      "date": "Saturday, November 14th",
      "max_temp": 13.35,
      "min_temp": 9.77,
      "wind_speed": 6.68,
      "rain": true,
      "total_rain": "2.09",
      "expected_rainfall": {
        "09:00:00": 0.37,
        "12:00:00": 1.72
      }
    }
  ],
  "max_in_week": 14,
  "min_in_week": 4,
  "average_temp": 9,
  "packing": "cold",
  "expected_rain": true
}
*/

function convert(data) {

  //the final data object to be sent back
  final_data = {
    city: data.city.name,
    country: data.city.country,
    weather: []
  }

  //getting unique dates
  var list_of_dates = [];

  for (date of data["list"]) {
    list_of_dates.push(date["dt_txt"].slice(0, 10))
  }

  var unique_dates = [... new Set(list_of_dates)]

  //for temperature
  var global_min = Infinity;
  var global_max = -Infinity;
```

```
  //making an object per date
  for (i = 0; i < unique_dates.length; i++) {

    // the local object that will get appended to weather list
    // in final data
    let weather_object = {}

    var cur_date = unique_dates[i];
    let cur_temp_min = [];
    let cur_temp_max = [];
    let wind_speed = [];
    let total_rain = 0
    var weather_type = null
    var rain_object = {}


    var chance_of_rain = false

    for (main_data of data["list"]) {
      if (cur_date === main_data["dt_txt"].slice(0, 10)) {

        cur_temp_min.push(main_data["main"]["temp_min"])
        cur_temp_max.push(main_data["main"]["temp_max"])
        weather_type = main_data.weather[0].main

        if (weather_type === "Rain") {
          chance_of_rain = true;
          rain_object[main_data["dt_txt"].slice(11, 19)] = main_data["rain"]
["3h"]
          total_rain += main_data["rain"]["3h"]
        }

        wind_speed.push(main_data["wind"]["speed"])

      }

    }


    min_temp = Math.min(...cur_temp_min)
    max_temp = Math.max(...cur_temp_max)

    weather_object["date"] = moment(cur_date, "YYYY-MM-DD").format("dddd,
MMMM Do")
    weather_object["max_temp"] = max_temp;
    weather_object["min_temp"] = min_temp;
    weather_object["wind_speed"] = Math.max(...wind_speed);

    weather_object["rain"] = chance_of_rain
    weather_object["total_rain"] = total_rain.toFixed(2);

    if (chance_of_rain) {
      weather_object["expected_rainfall"] = rain_object;
    }
```

```
    final_data.weather.push(weather_object)

    if (min_temp < global_min) {
      global_min = min_temp;
    }

    if (max_temp > global_max) {
      global_max = max_temp;
    }

  }

  //approximating results
  final_data["max_in_week"] = Math.ceil(global_max);
  final_data["min_in_week"] = Math.floor(global_min);

  //packing is dependent on average weather
  let avg_temp = (Math.ceil(global_max) + Math.floor(global_min)) / 2;

  final_data["average_temp"] = avg_temp;

  if (avg_temp > -10 && avg_temp <= 10) {
    final_data["packing"] = "cold";
  } else if (avg_temp > 10 && avg_temp <= 20) {
    final_data["packing"] = "warm";
  } else if (avg_temp > 20) {
    final_data["packing"] = "hot";
  } else {
    final_data["packing"] = "extreme_cold";
  }

  final_data["expected_rain"] = false;

  for (data of final_data.weather) {
    if (data.rain) {
      final_data["expected_rain"] = true;
    }
  }

  return final_data
}
```