```
/*
 * CSU44000 - Weather API
 *
 * This is a simple Express API which will use the OpenWeather API
 * (https://openweathermap.org/api) to provide a simple 5 day forecast
 * to a web application which for any desired city.
 */
const cors = require('cors');
const express = require('express');
const path = require('path');
const axios = require('axios');

const app = express();
const port = 3000;

/*
 * OpenWeather API Constants
 *
 * 5 Day Forecast - https://openweathermap.org/forecast5
 *
 */
const apiURL = "http://api.openweathermap.org/data/2.5"
const apiKEY = "3e2d927d4f28b456c6bc662f34350957"

let publicPath = path.resolve(__dirname, "public");

app.use(express.static(publicPath));
app.use(cors());

app.get('/', (req, res) => res.send('Hello World!'));

app.get('/forecast/:town', getForecast);

app.listen(port, () => console.log(`Example app listening on port
${port}!`));

function getForecast(req, res) {
    let town = req.params.town;
    console.log(`Generating weather forecast for town ${town}...`);

    let forecastSummary = {};
    let isRain = false;
    let forecastSentiment = null;

    axios.get(`${apiURL}/forecast?q=${town}&APPID=${apiKEY}`)
        .then(response => {
            let weatherData = response.data.list;

            // Loop over OpenWeather API response and extract data for each
day
            for (weatherEntry in weatherData) {

                // Make the date look nicer for front-end
                let date = new Date(response.data.list[weatherEntry].dt *
1000);
                date.setHours(0, 0, 0, 0);
                date = date.toLocaleDateString();

                // First check if there is a date entry for the given date,
if not create one
```

```
                if (!forecastSummary[date]) {
                    forecastSummary[date] = {
                        temperatures: [],
                        windSpeeds: [],
                        rainfallLevels: []
                    }
                }

                // Extract temperature and wind speed data

 forecastSummary[date].temperatures.push(weatherData[weatherEntry].main.temp);

 forecastSummary[date].windSpeeds.push(weatherData[weatherEntry].wind.speed);

                // Check if there is any rain
                if (weatherData[weatherEntry].rain && weatherData[weatherEntry].rain['3h']) {
                    isRain = true;

 forecastSummary[date].rainfallLevels.push(weatherData[weatherEntry].rain['3h']);
                }
            }

            // When finished extracting data, calculate averages
            for (dateEntry in forecastSummary) {
                forecastSummary[dateEntry].averageTemp = convertKelvinToCelsius(getAverage(forecastSummary[dateEntry].temperatures));
                forecastSummary[dateEntry].averageWind = getAverage(forecastSummary[dateEntry].windSpeeds);
                forecastSummary[dateEntry].rainfallLevels = getSum(forecastSummary[dateEntry].rainfallLevels);
                forecastSummary[dateEntry].temperatureRange = getMinMax(forecastSummary[dateEntry].temperatures);
            }

            // Get overall temperature sentiment
            temperatureSummary = getTemperatureSummary(forecastSummary);

            console.log(forecastSummary);
            console.log(isRain);
            console.log(forecastSentiment);

            // Send good response with result
            res.status(200);
            res.json({
                forecastSummary: forecastSummary,
                isRain: isRain,
                temperatureSummary: temperatureSummary
            });
        })
        .catch(error => {
            console.error(error);
            res.status(400);
            res.json({
                error: "Bad Request!"
            });
        })
}
```

```
107
108 // Returns a forecast sentiment (cold, warm, hot) and absolute min and max
109 function getTemperatureSummary(forecastSummary) {
110     let max = 0;
111     let min = forecastSummary[Object.keys(forecastSummary)[0]].averageTemp;
112     let sentiment = null;
113
114     let minMaxObj = {};
115
116     // Loop over every day getting the absolute min and max values
117     for (dateEntry in forecastSummary) {
118         minMaxObj = forecastSummary[dateEntry].temperatureRange;
119
120         // Check if the max on this day is more than current max
121         if (minMaxObj.max >= max)
122             max = minMaxObj.max;
123
124         // Check if the min on this day is more than current max
125         if (minMaxObj.min <= min)
126             min = minMaxObj.min;
127     }
128
129     console.log(`Overall max is ${max}`);
130     console.log(`Overall min is ${min}`);
131
132     if(max >= 20.0)
133         sentiment = "hot";
134
135     else if (max <= 20.0 && min >= 10.0)
136         sentiment = "warm";
137
138     else
139         sentiment = "cold";
140
141     return {
142         sentiment: sentiment,
143         max: max,
144         min: min
145     }
146 }
147
148 // Returns the min and max of an array of values
149 function getMinMax(array) {
150     let max = 0;
151     let min = array[0];
152
153     for (let i = 0; i < array.length; i++) {
154         if (array[i] >= max)
155             max = array[i];
156         else if (array[i] < min)
157             min = array[i];
158     }
159
160     return {
161         min: convertKelvinToCelsius(min),
162         max: convertKelvinToCelsius(max)
163     };
164 }
165
166 // Converts the temperature from Kelvin to Celsius
```

```
function convertKelvinToCelsius(kelvin) {
    if (kelvin < (0)) {
        return 'below absolute zero (0 K)';
    } else {
        let celciusVal = kelvin - 273.15
        return Math.round(celciusVal * 100) / 100;
    }
}

// Returns the sum of an array of values
function getSum(array) {

    if (array.length == 0)
        return 0;

    let total = 0;

    for (let i = 0; i < array.length; i++) {
        total += array[i];
    }

    return Math.round(total * 100) / 100;
}

// Returns the average value of an array of values
function getAverage(array) {
    let total = 0;

    for (let i = 0; i < array.length; i++) {
        total += array[i];
    }
    let avg = total / array.length;

    return Math.round(avg * 100) / 100;
}
```