```
const { response } = require('express')
const e = require('express')
const express = require('express')
const https = require('https')
const cors = require("cors");
const API_KEY = '3e2d927d4f28b456c6bc662f34350957'
const DEFAULT_UNITS = 'metric'
const app = express()
const port = 8081
const path=require("path")


let publicPath= path.resolve(__dirname,"public")
app.use(express.static(publicPath))
app.use(cors());
app.use(express.static(publicPath))
app.listen(port, () => console.log(`Example app listening on port ${port}!`))

function getlocation(location){
    let data = ''
    let url = `https://api.openweathermap.org/data/2.5/forecast?q=${location}&APPID=${API_KEY}&units=${DEFAULT_UNITS}`

    let p = new Promise((resolve, reject)=>{
        https.get(url, resp=>{
            resp.on("data", (d)=>{
                data +=d
            })
            resp.on("end", ()=>{
                const parsedData = JSON.parse(data)
                resolve(parsedData)
            })
        }).on("error", (e)=>{
            reject(e.message)
        })
    })
    return p
}

function getAirPollution(lat, long){
    let airRawData = ''
    let airurl = `https://api.openweathermap.org/data/2.5/air_pollution/forecast?lat=${lat}&lon=${long}&appid=${API_KEY}`

    let p = new Promise((resolve, reject)=>{
        https.get(airurl, resp=>{

            resp.on("data", (d)=>{
                airRawData+=d
            })
            resp.on("end", ()=>{

                const airData = JSON.parse(airRawData)
                resolve(airData)
            })
        }).on("error", (e)=>{
            reject(e.message)
        })
    })
    return p
}


app.get('/5dayforecast/:location', (req, res)=>{
        getlocation(req.params.location).then(result=>{
            if(result.message){
                res.json({error:result.message})
            }else{
                locationData = {
                    long: result.city.coord.lon,
                    lat: result.city.coord.lat
                }
                weatherData = (result.list).map(element => {
                    if(element.rain){
                        return {
                            date: element.dt_txt,
                            temp: Math.round(element.main.temp),
                            wind: element.wind.speed,
                            rain: element.rain['3h']
                        }
                    }else{
                        return {
                            date: element.dt_txt,
                            temp: Math.round(element.main.temp),
                            wind: element.wind.speed,
                            rain: 0
                        }
                    }

                })
                res.json({weatherData:weatherData, locationData:locationData})
            }
        })
})

app.get('/5dayforecast/:lat/:long', (req, res)=>{
    getAirPollution(req.params.lat, req.params.long).then(result=>{
        if(result.message){
            res.json({error:result.message})
        }else{
            airData = (result.list).map(element=>{
                return{
                    pm25:element.components.pm2_5
                }
            })
            res.json({airData:airData})
        }

    })
})
```