Tampereen ammattikorkeakoulu

Datankäsittely Ilmatieteen laitoksen tarjoamalle avoimen datan palvelulle

Jere Aho

OPINNÄYTETYÖ
Toukokuu 2020

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

AHO, JERE:
Datankäsittely Ilmatieteen laitoksen tarjoamalle avoimen datan palvelulle

Opinnäytetyö 19 sivua, joista liitteitä 1 sivua
Toukokuu 2020

Työn tavoitteena oli toteuttaa ohjelmointirajapinta Tampereen ammattikorkeakoululle, joka on mukana Study and Test of Smart Control and Storage of Energy for Nearly Zero Energy Buildings - smart case nzeb -tutkimushankkeessa.

Rajapinta muuntaa Ilmatieteen laitoksen tarjoaman avoimen datan koneluettavaan muotoon, jotta sitä voidaan hyödyntää Tampereen ammattikorkeakoulun projektiryhmän toteuttamassa python sovelluksessa. Rajapinnan tarkoituksena on olla jatkuvasti ajossa virtuaalikoneella, jotta sitä voidaan kutsua milloin vain.

Asiasanat: ohjelmisto rajapinta

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Software Engineering

AHO, JERE:
Data handling for open data provided by Finnish Meteorological Institute

Bachelor's thesis 19 pages, appendices 1 pages
May 2020

The goal of this thesis was to build an application programming interface for Tampere University of Applied Sciences, which is collaborating in the Study and Test of Smart Control and Storage of Energy for Nearly Zero Energy Buildings - smart case nzeb -study.

The purpose of the program build for this thesis is to modify data provided by Finnish Meteorological Institute into machine readable format so it may be used by a project group within Tampere University of Applied Sciences. The program is made to be always online on a virtual machine., so it may be accessed at any time.

Key words: application programming interface

SISÄLLYS

LYHENTEET JA TERMIT

API	Application programming interface
cURL	Client URL
IP address	Internet Protocol address
I/O	Input / Output
JSON	JavaScript Object Notation
TAMK	Tampereen ammattikorkeakoulu
URL	Uniform Resource Locator
XML	Extensible Markup Language

1 JOHDANTO

Tämä opinnäytetyö on tehty osaksi Tampereen ammattikorkeakoulun Nzeb-hanketta. Hanke hyödyntää ilmatieteen laitoksen tarjoamaa avointa dataa, joka on tarjottuna XML muotoisena. Hankkeeseen haluttiin yksinkertainen rajapinta, jota käyttämällä saadaan muunnettua tarjottu data JSON muotoon.

Työn tavoitteena oli valmistaa kevyt ja yksinkertainen rajapinta, jolle voidaan lähettää python sovelluksesta pyyntö, joka lähetetään edelleen ilmatieteen laitokselle, jonka palautus muokataan JSON-muotoiseksi dataksi ja tämä palautetaan myöhempää käyttöä varten python sovellukselle. Ilmatieteen laitoksen tarjoamassa datassa on hankkeen käyttötarkoitukseen ylimääräistä dataa, joka karsitaan rajapinnan avulla pois.

Työ on juuriaan myöten riippuvainen Ilmatieteen laitoksen tarjoamasta rajapinnasta, jos tämän palvelun malli muuttuu, vaatii työ luultavasti muokkausta toiminnan jatkamiseksi. Työn valmistushetkellä avoimen datan palvelu oli ilmainen eikä se vaatinut API-avainta taikka muuta rekisteröitymistä. Avoimeen dataan käsiksi pääsemiseen vaadittiin API-avain vielä lokakuuhun 2019 asti, joten tilanteen pysymisenä samana ei ole takeita. (Ilmatieteen laitos: Avoin data muutosloki 2019.)

2 PALVELUISTA JA HANKKEESTA

2.1 TAMKin hankkeesta

Tampereen ammattikorkeakoulu on osana Business Finlandin rahoittamaa Study and Test of Smart Control and Storage of Energy for Nearly Zero Energy Buildings - smart case nzeb -tutkimushanketta. Hankkeessa tutkitaan mahdollisuutta vaikuttaa kiinteistöjen resurssitehokkuuden optimoimiseen tekoälyn avulla. Tätä varten kerätään dataa ulkoilman olosuhteista ja tilojen käytöstä, jotta voidaan ennustaa ja ohjata energian käyttöä kiinteistöissä. Tämän datan pohjalta selvitetään mittausten välisiä riippuvuussuhteita ja mahdollisuuksia hyödyntää tekoälypohjaisia mallinnusratkaisuja rakennuksen toiminnan ennakoinnissa sekä energiatehokkaan toiminnan optimoinnissa (nzeb-projektin kuvaus 2020).

Tämän työn osuus oli valmistaa rajapinta, joka muuntaa ilmatieteen laitoksen tarjoaman säädatan projektissa olevan tekoälyalgoritmin vaatimaan muotoon. Tämä vaati XML muotoisen datan parsimista, jotta siitä saatiin muodostettua järkevästi käsiteltävää JSON muotoista dataa.

2.2 Avoimesta datasta

Ilmatieteen laitoksen avoimen datan verkkopalvelun kautta voi hakea, katsella ja ladata laitoksen tuottamia tietoaineistoja koneluettavassa muodossa maksutta. Verkkopalvelun tekninen toteutus noudattaa Inspire-direktiivin vaatimusmäärittelyjä. Sisällön osalta toteutus on Inspire-direktiivissä määriteltyä laajempi. Avointa dataa jaetaan hakupalvelussa Open Geospatial Consortium –standardien mukaisesti (Ilmatieteen laitoksen avoin data ja lähdekoodi 2019).

Palvelu mahdollistaa datan keräämisen yli viideltäkymmeneltä sääasemalta. Palvelulle voi lähettää useita eritapaisia kyselyitä, jotka palauttavat erilaista tietoa, muun muassa voi kysellä viimeisimpiä lentosääviestejä taikka ulkoilman radioaktiivisten aineiden pitoisuuksia. Suurinta osaa datasta voi kysellä joko havaintoarvoina taikka ennusteina.

Palvelun käyttöönotossa käyttäjä hyväksyy Creative Commons -lisenssin, tässä on määriteltynä rajoitteita käyttäjän kyselymääristä erilaisiin palveluihin. Nämä rajat ovat kuitenkin asetettu hyvin yläkanttiin, koska palveluun saa tehdä vuorokauden aikana 20 000 kyselyä. Työssä käytetty kysely palauttaa tuntikohtaisia säähavaintoja, joten tämä raja ei tavallisessa käytössä tule vastaan.

2.3 Työssä käytetty kysely

Tässä työssä on käytetty kyselyä, joka palauttaa tuntikohtaisia säähavaintoja suomen havaintoasemilta. Kyselyn palauttamat parametrit ja niiden selitteet ovat listattuna taulukossa (TAULUKKO 1).

TAULUKKO 1. Kyselyn palauttamat parametrit

Parametri	**Selite**
TA_PT1H_AVG	Lämpötila keskiarvo
TA_PT1H_MAX	Lämpötila maksimi
TA_PT1H_MIN	Lämpötila minimi
RH_PT1H_AVG	Suhteellisen kosteuden keskiarvo
WS_PT1H_AVG	Tuulen nopeuden keskiarvo
WS_PT1H_MAX	Tuulen nopeuden 10 minuutin keskiarvojen maksimi
WS_PT1H_MIN	Tuulen nopeuden 10 minuutin keskiarvojen minimi
WD_PT1H_AVG	Tuulen suunnan keskiarvo
PRA_PT1H_ACC	Sademäärä
PRI_PT1H_MAX	Sateen intensiteetin maksimi
PA_PT1H_AVG	Ilmanpaineen keskiarvo
WAWA_PT1H_RANK	Merkittävin sääkoodi

Kysely vaatii käyttäjää antamaan hakuehtona vähintään yhden paikkamääreen. Työssä rakennetulle rajapinnalle voi antaa kyselyparametreinä paikka-arvon sekä ajankohdan alun ja lopun, jolta dataa halutaan. Kyselylle voi antaa myös muita hakuehtoja, mutta tämän työn vaatimuksena oli saada data paikka- ja aikakohtaisesti. Taulukossa 1 näkyvissä parametreissa käytettyjä lyhenteitä ei löytynyt selitettynä Ilmatieteen laitoksen sivulta, eikä referenssiä minkä standardin mukaan nämä ovat nimetty, joten ne ovat raportin kirjoittajan päättelemiä arvoja, täten ne eivät välttämättä ole paikkaansa pitäviä. Niiden nimeämisessä on käytetty referenssinä kyselyn selosteessa kerrottuja arvoja.

3 KÄYTETYISTÄ TEKNIIKOISTA

3.1 Tekniikoiden valinta

Rajapinnan valmistukseen käytetyt tekniikat valittiin muutaman eri tekniikan testauksen jälkeen. Testausten perusteelta valitut tekniikat olivat sovelluksen valmistamiseen sopivia, ne mahdollistivat kevyen ohjelman rakentamisen ja helpon kutsumisen sovelluksen ulkopuolelta. Sovelluksen olisi voinut toteuttaa usealla eri tavalla, mutta työssä päädyttiin Express web applikaatio runkorakenteeseen ja Node.js ympäristöön, koska nämä olivat työn tekijälle entuudestaan tuttuja.

Ilmatieteen laitos mainitsee sivuillaan MetOLib JavaScript kirjaston, joka sisältää implementaatioita API-olioluokista, joilla on mahdollista suorittaa kyselyitä avoimeen dataan. MetOLib mahdollistaa myös XML datan parsimisen JSON dataksi.

Tässä työssä ei hyödynnetty MetOLib kirjastoa, koska se on turhan raskas ohjelmiston tarkoitukseen ja se sisältää useita ylimääräisiä toimintoja. MetOLib kirjastoa ei myöskään ollut päivitetty vastaamaan Ilmatieteen laitoksen muutosta API-avainten tarpeellisuuteen ja tästä syystä MetOLib kirjastoon olisi täytynyt tehdä muutoksia sen toimivuuden varmistamiseksi. Muutokset vaatisivat myös useita testejä toimivuuden takaamiseksi.

3.2 Node.js

Node.js on asynkroninen, tapahtumapohjainen JavaScript ajoympäristö, joka on suunniteltu käytettäväksi skaalattavien verkkosovellusten rakentamiseen. Node.js mahdollistaa useiden samanaikaisten yhteyksien käsittelyn. Kun yhteys Node.js applikaatioon luodaan, se laukaisee takaisinkutsun, mutta kun applikaatioon ei tehdä yhteyskutsuja, se menee automaattisesti lepotilaan. Tämä eroaa nykyajan yleisemmästä samanaikaisuus mallista, jossa hyödynnetään käyttöjärjestelmän säikeitä. Säie-pohjainen verkkotoiminta on suhteellisen tehotonta ja hankalaa käyttää. Tämä ei tarkoita että Node.js ei pysty hyödyntämään useita

säikeitä. Node.js on mahdollista hyödyntää lapsiprosesseja tämän suorittamiseen.

Node.js käyttäjien ei myöskään tarvitse huolehtia prosessin lukkiutumisesta, koska sen ajosta ei aiheudu prosessilukkoja. Lähes mikään funktio Node.js käyttää I/O siirräntää, jotka yleensä aiheuttavat prosessilukkiutumisen. Tästä syystä skaalautuvia järjestelmiä on järkevää kehittää Node.js käyttämällä (About Node.js 2020).

Työssä käytettiin työkaluna Node.js, koska se ja Express yhteisenä pakettina mahdollistaa yksinkertaisen ja kevyen rajapinnan rakentamisen, jonka voi asentaa ongelmitta erilaisille järjestelmille. Työn valmistusvaiheessa asennusta testattiin sekä Windows 10 käyttöjärjestelmälle sekä virtuaaliselle linux ubuntu käyttöjärjestelmälle. Molemmissa tapauksissa asennus onnistui ongelmitta.

3.3 Express

Express on web ohjelmistokehys Node.js pohjaisiin ohjelmistoihin. Se yksinkertaistaa kyselyiden URL polkujen käsittelyn ja yleisten web applikaatioasetusten asettamisen (Express/Node introduction 2020).

Työssä käytettiin Express web runkorakennetta koska se mahdollistaa minimalistisella asentelulla toimivan polkukäsittelyn. Muitakin runkorakenteita on mahdollista käyttää Node.js kanssa, mutta Express on näistä suosituin. Se on avoimen lähdekoodin sovellus, tästä syystä suureen osaan mahdollisista ongelmista löytyy internetistä ratkaisu.

3.4 Docker

Docker on työkalu, joka on suunniteltu yksinkertaistamaan applikaatioiden luomisen, käyttöönoton ja ajamisen kontteja käyttäen. Konttien avulla ohjelmakehittäjä voi pakata applikaation kaikkien sen tarvitsemien osien kanssa, kuten lisäkirjastot

ja riippuvuudet, ja ottaa se uudelleen käyttöön uudessa ympäristössä yhtenä pakettina. Näin ohjelmakehittäjä voi olla varma, että applikaatio toimii millä vain linux-pohjaisella laitteella huolimatta kustomoiduista asetuksista, joita uudella alustalla saattaa olla.

Dockeria voisi kuvailla virtuaalikoneena, sillä erolla että Docker ei tee uutta käyttöjärjestelmää, vaan mahdollistaa saman linux kernel version käytön applikaatiolle (What is Docker 2020).

Työssä valmistettu rajapinta on asennettuna Docker konttiin TAMK:in linux virtuaalikoneelle, josta sitä on helppo kutsua ja se on mahdollista siirtää ongelmitta, jos tälle on tarvetta.

3.5 API

API eli ohjelmointirajapinta määrittelee miten järjestelmät voivat keskustella keskenään. Sillä määritellään työkalut, protokollat ja rutiinit, joilla applikaatio kehitetään. Web kehityksessä ohjelmointirajapinta määritellään usein teknisinä tietospesifikaatioina, miten sivuun voi ottaa yhteyden ja millä formaatilla tähän vastataan.

Tässä työssä hyödynnetään ilmatieteen laitoksen ohjelmointirajapintaa pyytämällä tältä dataa. Avoimen datan palvelun rajapinnassa on määritelty tietyt päätepisteet eri kyselyille ja tämän perusteella kyselyn lähettäjälle palautetaan tälle päätepisteelle määritelty palautedata.

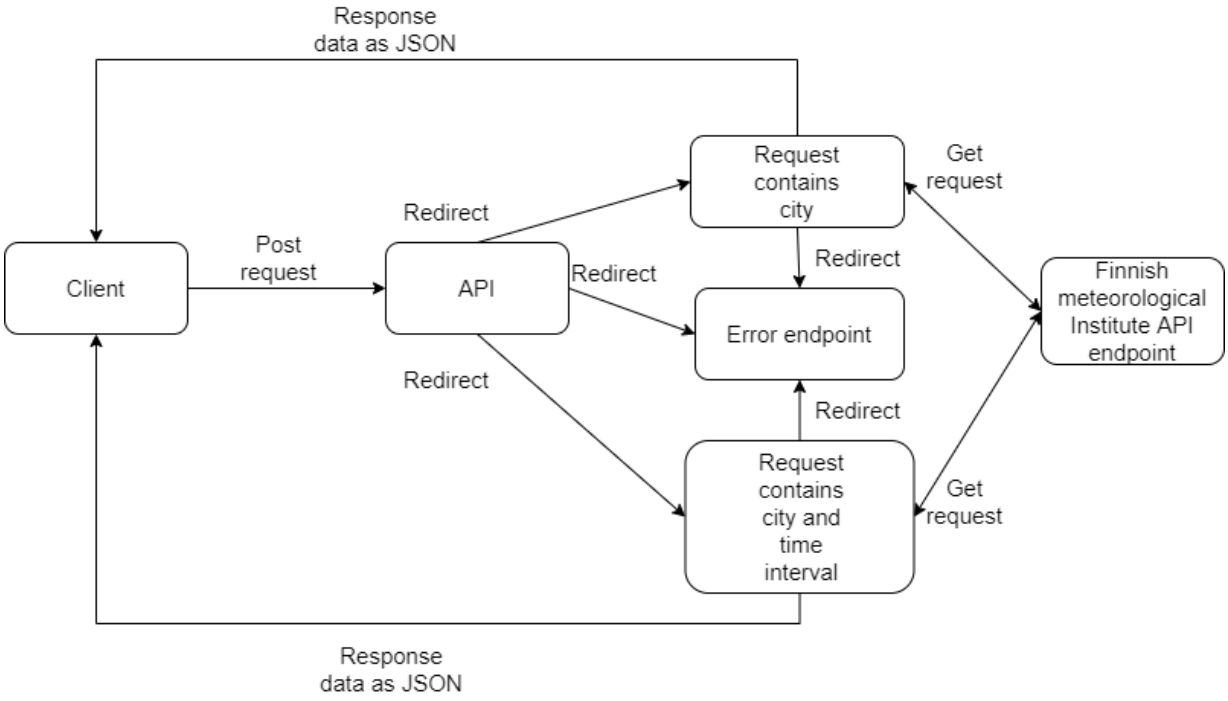
Tässä työssä valmistettu ohjelmointirajapinta toimii kyselyn lähettäjän ja avoimen datan palvelun rajapinnan välissä. Työn rajapinta vaatii kyselijältä samat tiedot kuin avoimen datan palvelu. Nämä tiedot saatuaan se lähettää kyselyn avoimen datan palvelun rajapinnalle, ottaa palautteesta relevantit tiedot talteen JSON muotoon ja palauttaa tämän alkuperäiselle kyselyn lähettäjälle.

4 OHJELMOINTIRAJAPINNAN TOIMINTA

Työssä valmistettu rajapinta on suunniteltu olemaan jatkuvasti ajossa alustalla jolle se asennetaan. Tämän työn tapauksessa se laitettiin ajoon TAMKin linux pohjaiselle virtuaalikoneelle, josta sitä voidaan kutsua milloin vain. TAMKin virtuaalikoneella ohjelman suorittaminen mahdollistaa myös pääsyn hallinnan rajapinnalle. Tämä on tärkeää koska ilmatieteen laitos on asettanut rajat kutsujen kysyntätaajuudelle ja ilmatieteen laitos saattaa musta listata kyselyrajan ylittävät IP-osoitteet.

4.1 Ohjelmointirajapinnan rakenne

Työssä valmistettu ohjelmointirajapinta käsittelee pyynnöt kuvion (KUVIO 1) mukaisesti. Käyttäjän post-pyynnöstä käsitellään sen sisältämät parametrit ja niiden perusteella pyyntö uudelleenohjataan rajapinnan eri päätepisteisiin. Jos pyynnössä oli parametrina vain kaupunki, rajapinta ohjaa kyselyn päätepisteeseen, joka lähettää uuden pyynnön Ilmatieteen laitoksen avoimen datan palvelulle. Saatuaan palautteen avoimen datan palvelulta tämä palaute käsitellään ja palautetaan alkuperäisen pyynnön lähettäjälle. Tapauksessa, jossa pyynnön parametreissa on annettu kohdekaupunki sekä aikaväli, jolta data halutaan, ohjaa rajapinta pyynnön eri päätepisteeseen. Jos pyynnössä ei ole annettuna parametreissä kohdekaupunkia taikka aikaa, tai toisen parametrin ollessa virheellinen, ohjaa rajapinta pyynnön virhe päätepisteeseen. Virhe päätepisteeseen ohjataan myös tapauksissa, joissa Ilmatieteen laitos palauttaa odottamatonta dataa. Tämä usein johtuu avoimen datan palvelun palvelukatkosta.



KUVIO 1. Ohjelmointirajapinnan rakenne

4.2 Toteutettava kysely

Rajapinnalle voidaan lähettää kuvan (KUVA *1*) mukainen kysely, jossa on määriteltynä kaupunki ja aikaväli jolta dataa halutaan. Rajapinta hyväksyy myös kyselyn, jossa on määriteltynä ainoastaan kohdekaupunki, jolloin se palauttaa kyseisen kaupungin säädatan 24 tuntia pyyntöhetkestä taaksepäin. Kuvassa (KUVA 1) on nähtävissä kyselyn lopussa -L merkki, tämä vaaditaan cURL pyyntöön, koska se mahdollistaa pyynnön uudelleenohjauksen. Tämä on tarpeellista koska ohjelmointirajapinta uudelleenohjaa kyselyssä annettujen parametrien perusteella eri päätepisteeseen. Jos kyselystä puuttuu -L, tekee rajapinta kyselyn Ilmatieteen laitokselle, mutta palaute ei saavu alkuperäiselle kyselijälle.

```
$ curl -d 'location=helsinki&starttime=2018-02-27T00:00:00Z&endtime=2018-02-28T00:00:00Z'
 http://localhost:5000/query -L
```

KUVA 1. Rajapinnalle lähetetty kysely

Ilmatieteen laitoksen latauspalvelu käyttää WFS 2.0-palvelua ja se toteuttaa WFS Simple Profile:n. Tämä tarkoittaa, että kyselyt tulee tehdä tallennettujen kyselyiden avulla (Latauspalvelun pikaohje 2020).

4.3 Datan parsiminen

Rajapinnalle lähetetystä cURL pyynnöstä poimitaan paikka- ja aikatiedot, joista muodostetaan uusi kysely, joka lähetetään ilmatieteen laitokselle. Jos kyselyn paikka- ja aikaparametrit olivat hyväksyttäviä, palauttaa ilmatieteen laitoksen avoin data - palvelu rajapinnalle kuvan (KUVA *2*) mukaista XML-dataa.

```
▼<wfs:member>
  ▼<BsWfs:BsWfsElement gml:id="BsWfsElement.1.1.1">
    ▼<BsWfs:Location>
      ▼<gml:Point gml:id="BsWfsElementP.1.1.1" srsDimension="2" srsName="http://www.opengis.net/def/crs/EPSG/0/4258">
          <gml:pos>61.50124 23.76478 </gml:pos>
        </gml:Point>
      </BsWfs:Location>
      <BsWfs:Time>2018-02-27T00:00:00Z</BsWfs:Time>
      <BsWfs:ParameterName>TA_PT1H_AVG</BsWfs:ParameterName>
      <BsWfs:ParameterValue>-15.4</BsWfs:ParameterValue>
    </BsWfs:BsWfsElement>
  </wfs:member>
▼<wfs:member>
  ▼<BsWfs:BsWfsElement gml:id="BsWfsElement.1.1.2">
    ▼<BsWfs:Location>
      ▼<gml:Point gml:id="BsWfsElementP.1.1.2" srsDimension="2" srsName="http://www.opengis.net/def/crs/EPSG/0/4258">
          <gml:pos>61.50124 23.76478 </gml:pos>
        </gml:Point>
      </BsWfs:Location>
      <BsWfs:Time>2018-02-27T00:00:00Z</BsWfs:Time>
      <BsWfs:ParameterName>TA_PT1H_MAX</BsWfs:ParameterName>
      <BsWfs:ParameterValue>-15.0</BsWfs:ParameterValue>
    </BsWfs:BsWfsElement>
  </wfs:member>
```

KUVA 2 Ilmatieteen laitoksen tarjoama avoin data

Tämä data sisältää loppukäyttöön tarvittavia arvoja XML -tageissa pos, Time, ParameterName ja ParameterValue. Ohjelma muuntaa datan ensin string muotoiseksi, joka sitten muunnetaan JSON muotoon. Ohjelman datan parsinta osio näkyy kuvassa (KUVA 3). Kaikki tämän kyselyn palauttamat parametrinimet ja niiden selvennykset ovat näkyvillä taulukossa (TAULUKKO 1).

```
.then(res => res.text()) // takes response xml and makes it into string text
.then(bodyText => {
    // parses text xml into json
    parser.parseString(bodyText, function (err, result) {
        // remove unneccessary outer 'tags' from json
        var weatherData = result['wfs:FeatureCollection']['wfs:member'];

        var objArray = [];
        // Looping data, making it more palatable, and adding new objects to array
        for (var i = 0; i < weatherData.length; i++) {
            //prepare object to take parameters
            var obj = {
                latitude: '',
                longtitude: '',
                time: '',
                parameterName: '',
                parameterValue: ''
            };

            // temporary variables to get data out of useless arrays
            var tempTime, tempParName, tempParVal;

            // 'Mining' the parameters from the data
            var singleObservation = weatherData[i]["BsWfs:BsWfsElement"]; // takes one parameter from one station
            //time, name and value data handling

            tempTime = singleObservation[0]["BsWfs:Time"];
            tempParName = singleObservation[0]["BsWfs:ParameterName"];
            tempParVal = singleObservation[0]["BsWfs:ParameterValue"];
            //latlong related data handling

            var sOLocation = singleObservation[0]["BsWfs:Location"] // latlong is located deeper in the data
            var point = sOLocation[0]["gml:Point"]// latlong is located deeper in the data
            var latlong = point[0]["gml:pos"]// latlong is located deeper in the data
            latlong = '' + latlong; // makes the variable into string so substring works
            var lat = latlong.substring(0, latlong.indexOf(' ')); // latitude and longitude are within same string
            var lon = latlong.substring(latlong.indexOf(' '), latlong.lenght); // substring takes tem apart
            lon = lon.split(' ').join(''); // removes empty spaces from string

            // adding data to obj
            obj.latitude = lat;
            obj.longtitude = lon;
            obj.time = tempTime[0];
            obj.parameterName = tempParName[0];
            obj.parameterValue = tempParVal[0];
            // push obj to object array      You, a few seconds ago • Uncommitted changes
            objArray.push(obj);
        }
```

KUVA 3. Datan parsinta

Ohjelma poistaa haetusta datasta ylimääräiset tagit, jonka jälkeen data on mahdollista pyörittää luupissa ja poimia kaikki data riippumatta sen sisältämien elementtien määrästä. Datan elementtien määrä vaihtelee pyydetyn aikavälin muuttuessa, jonka takia tämä on tarpeellista. Ohjelmassa on luupin sisällä luotu JavaScript objekti, johon poimitaan halutut parametrit alkuperäisestä datasta.

XML datan tagit ovat JSON muodossa taulukoita, jotka puretaan erillisiin muuttujiin, joista sitten poimitaan parametrit. Kun tämä on tehty yhdelle datan parametrille, esimerkiksi lämpötilalle, tämän datan sisältävä JavaScript objekti lisätään

tietokenttäjonoon. Tämä toistetaan niin monta kertaa, kun alkuperäisessä tietokenttäjonossa on elementtejä.

Rajapinta sitten palauttaa kyselyn pyytäjälle kuvan (KUVA 4) mukaisen tietokenttäjonon, joka sisältää JavaScript objekteja.


```
"objArray": [{
    "latitude": "60.17523",
    "longtitude": "24.94459",
    "time": "2020-05-05T10:00:00Z",
    "parameterName": "TA_PT1H_AVG",
    "parameterValue": "13.2"
}, {
```


KUVA 4. Palautettu JSON data

4.4 Virhetilanteet

Ohjelman ajossa yleisimmät virhetilanteet ovat joko pyynnön parametreissa ilmenneet virheet, esimerkiksi jos ohjelmalle annetaan virheellinen kaupungin nimi taikka aikaväli, joka päättyy tulevaisuuteen, taikka aikaväli jolle ilmatieteen laitoksen tietokannassa ei ole dataa.

Toinen yleinen virhetilanne riippuu ilmatieteen laitoksen palvelujen ylläpidosta. Jos avoimen datan palvelut ovat poissa toiminnasta, ei ohjelma voi tätä tilannetta korjata.

Molemmissa tilanteissa rajapinta ohjaa pyynnön error -palautteeseen. Tämä palaute on molemmissa tapauksissa sama, joten siitä ei pysty päättelemään mikä aiheutti epäonnistuneen datan haun. Tosin ongelma on helppo tarkistaa joko omasta syötteestä taikka tarkistamalla ilmatieteen laitoksen nettisivut.

5 POHDINTA

Rajapinnan toiminnallisuuden tekemiseen ei itsessään kulunut paljoa aikaa, mutta oikean kyselyn löytämiseen ja usean version testaamiseen kului suurin osa projektiin käytetystä ajasta. Ilmatieteen laitoksen avoimen datan palvelussa on useita kymmeniä erilaisia kyselyitä, eikä näiden parsiminen toiminut samalla ohjelmalla. Työtä aloitettiin myös aluksi tekemään python- pohjaisilla ratkaisuilla, koska se oli taustatutkimuksen mukaan sopiva sellaiseen parsimiseen, jota tässä työssä suoritettiin. Tämä aika meni projektin valmistumisen kannalta hukkaan, mutta tästä on mahdollista oppia tekemään työt tutuilla tekniikoilla jollei toisista tekniikoista ole selviä hyötyjä.

Rajapintaan olisi mahdollista tehdä jatkokehitystä, jotta sillä voisi suorittaa useampia kyselyitä ilmatieteen laitoksen avoimen datan palveluun. Myös virheenkäsittely voisi ilmoittaa selkeästi käytössä ilmenneen virheen luonteen joko virhekoodilla taikka selkeällä virheviestillä.

LÄHTEET

Ilmatieteen laitos. Avoin data muutosloki. Luettu 23.4.2020. https://www.ilmatieteenlaitos.fi/avoin-data-muutosloki

Ilmatieteen laitos. Ilmatieteen laitoksen avoin data ja lähdekoodi. Luettu 21.4.2020. https://www.ilmatieteenlaitos.fi/avoin-data

Ilmatieteen laitos. Latauspalvelun pikaohje. Luettu 11.5.2020. https://www.ilmatieteenlaitos.fi/latauspalvelun-pikaohje

MDN web docs. Express/Node introduction. Luettu 27.4.2020. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction

Nodejs. About Node.js. Luettu 27.4.2020. https://nodejs.org/en/about/

Opensource. What is Docker. Luettu 13.5.2020. https://opensource.com/resources/what-docker

Tuni. Nzeb-projektin kuvausta. Luettu 17.4.2020. https://www.tuni.fi/fi/tutkimus/study-and-test-smart-control-and-storage-energy-nearly-zero-energy-buildings-smart-case

LIITTEET

Liite 1. Ohjelmointirajapinnan käyttö

Rajapinta on käytössä TAMKin linux virtuaalikoneella osoitteessa https://nzeb-weather-fetcher.tamk.cloud/. Siihen on mahdollista lähettää kyselyitä esimerkiksi kyselyllä curl -d 'location=helsinki&starttime=2018-02-27T00:00:00Z&endtime=2018-02-28T00:00:00Z' https://nzeb-weather-fetcher.tamk.cloud/ -L, joka palauttaa säädatan helsingistä aikaväliltä 27.2.2018 – 28.2.2018.