Tampereen ammattikorkeakoulu

Web-sovelluksen kehittäminen Node-RED-ohjelmointityökalulla

Antti Kivineva

OPINNÄYTETYÖ
Toukokuu 2020

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

KIVINEVA, ANTTI:
Web-sovelluksen kehittäminen Node-RED-ohjelmointityökalulla

Opinnäytetyö 45 sivua, joista liitteitä 8 sivua
Toukokuu 2020

Toiminnallinen opinnäytetyö toteutettiin yhteistyössä Tampereen ammattikorkeakoulun kanssa. Tavoitteena oli lisätä tietoa Node-RED-ohjelmointityökalun käyttömahdollisuuksista ja soveltuvuudesta web-sovelluksen kehittämistyössä. Opinnäytetyön tarkoituksena oli luoda web-sovellus Node-RED-ohjelmointityökalulla. Opinnäytetyön tuotoksen tehtävänä oli kuvata prosessia siitä, kuinka Node-RED-ohjelmointityökalu soveltuu web-sovelluksen kehittämistyöhön.

Opinnäytetyön teoriaosuus koottiin ajankohtaisen kansallisen ja kansainvälisen tutkimustiedon pohjalta. Teoriaosuudessa käsitellään perinteistä web-ohjelmointia, tietovuo-ohjelmointia sekä erityisesti Node-RED-ohjelmointityökalua. Teoriaosan pohjalta tuotettiin web-sovellus käyttäen Node-RED-ohjelmointityökalua. Web-sovelluksena luotiin käyttäjän valitseman kaupungin säätietoja tarjoava palvelu.

Opinnäytetyö nostaa esille Node-RED-ohjelmointityökalun käyttömahdollisuuksia web-sovelluksen kehittämisessä käytännönläheisessä sekä helposti ymmärrettävässä muodossa. Opinnäytetyö lisää tietoa Node-RED-ohjelmointityökalun hyödyistä ja haasteista web-sovelluksen kehittämisessä. Lisätutkimusta tarvittaisiin Node-RED-ohjelmointityökalun käytettävyydestä eri konteksteissa sekä tietovuo-ohjelmoinnista yleisesti. Tutkittaessa Node-RED-ohjelmointityökalun hyödynnettävyyttä web-sovelluksen kehityksessä olisi jatkossa optimaalista pyrkiä käyttämään erilaisia moderneja käyttöliittymäkehittämiseen tarkoitettuja viitekehyksiä sekä tehdä lisätutkimusta web-sovelluksen kehittämisen tuotannollisesta näkökulmasta.

Asiasanat: node-red, web-ohjelmointi, tietovuo-ohjelmointi, web-sovellus

ABSTRACT

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

KIVINEVA, ANTTI:
Web application development with Node-RED programming tool

Bachelor's thesis 45 pages, appendices 8 pages
May 2020

This functional study was conducted in cooperation with Tampere University of Applied Sciences. The objective of this study was to increase information about the range of possibilities and applicability of Node-RED programming tool in web application development. The purpose of this study was to create a web application using Node-RED programming tool. The objective of the web application was to describe the process of how Node-RED programming tool suits web application development.

The theory of this study was gathered from relevant national and international researches. The theory part of the study covers traditional web programming and flow-based programming as well as Node-RED programming tool. A web application serving weather information of city chosen by the user was developed as the purpose of this study. The weather application was based on the theory of the study.

The thesis discusses the range of possibilities and applicability of Node-RED programming tool in web application development in practical and understandable manner. The thesis increases information about the benefits and challenges of developing web applications using Node-RED programming tool. Further research is needed on both usability of Node-RED programming tool in various contexts and flow-based programming in general. Utilizing modern frontend frameworks would be optimal in future researches on applicability of Node-RED in web development. In addition, further research is needed on web application development from productional perspective when using Node-RED programming tool.

Key words: node-red, web programming, flow-based programming, web application

SISÄLLYS

1 JOHDANTO 5
2 OPINNÄYTETYÖN TAVOITE, TARKOITUS JA TEHTÄVÄ 6
3 PERINTEINEN WEB-OHJELMOINTI 7
3.1 Palvelinohjelma 7
3.2 Käyttöliittymä 7
3.3 Ohjelmointirajapinta 8
4 TIETOVUO-OHJELMOINTI 10
4.1 Node-RED 10
5 OPINNÄYTETYÖN TOTEUTUS 13
5.1 Metodologia 13
5.2 Opinnäytetyöprosessin kuvaus 14
5.3 Web-sovelluksen kuvaus ja arviointi 14
5.3.1 Web-sovelluksen suunnittelu 15
5.3.2 Web-sovelluksen toteutus 16
5.3.3 Web-sovelluksen arviointi 25
6 POHDINTA 28
6.1 Node-RED-ohjelmointityökalun hyödynnettävyys web-sovelluksen kehittämisessä 28
6.2 Opinnäytetyön arviointi 30
6.3 Opinnäytetyön hyödynnettävyys sovelluskehittäjän näkökulmasta 32
6.4 Opinnäytetyön jatkokehittämisehdotukset 33
LÄHTEET 35
LIITTEET 38
Liite 1. Sääsovellus JSON-tiedostona 38

1 JOHDANTO

Node-RED on alun perin IBM:n vuonna 2013 kehittämä visuaalinen ohjelmointityökalu, joka pohjautuu J. Paul Morrisonin 1970-luvulla kehittämään tapaan kuvata sovelluksen käyttäytymistä laatikoiden verkostoina. Node-RED-ohjelmointityökalussa data kulkee näiden solmuiksi (node) kutsuttujen laatikoiden läpi. Tämä visuaalisen ohjelmoinnin tapa antaa ohjelmoijalle valmiudet nopeaan ja selkeään tiedon hakuun sekä sen käsittelyyn ja esittämiseen halutulla tavalla. (Node-RED n.d.a) Tässä opinnäytetyössä käsitellään Node-RED-ohjelmointityökalun käytettävyyttä web-sovelluksen kehittämisessä.

Opinnäytetyö tehtiin yhteistyössä Tampereen ammattikorkeakoulun kanssa. Yhteistyötahon toiveena oli saada käytännönläheinen ja lukijaa ohjaava raportti. Opinnäytetyön metodiksi valikoitui siten toiminnallinen eli tuotokseen painottuva opinnäytetyö. Opinnäytetyön tavoitteena on lisätä tietoa Node-RED-ohjelmointityökalun käyttömahdollisuuksista ja soveltuvuudesta web-sovelluksen kehittämistyössä. Opinnäytetyön tarkoituksena on luoda web-sovellus Node-RED-ohjelmointityökalulla. Opinnäytetyön tuotoksen tehtävänä on kuvata prosessia siitä, kuinka Node-RED-ohjelmointityökalu soveltuu web-sovelluksen kehittämistyöhön.

Ohjelmistotekniikan opetussuunnitelmaan ei juurikaan sisälly opetusta koskien visuaalista ohjelmointia tai tietovuo-ohjelmointia, joten aiheen pariin päätyminen on opiskelijoiden omien tarpeiden ja mielenkiinnon varassa. Tämä opinnäytetyö tuo käytännönläheisessä ja helposti ymmärrettävässä muodossa esille Node-RED-ohjelmointityökalun käyttömahdollisuuksia web-sovelluksen kehittämisessä.

2 OPINNÄYTETYÖN TAVOITE, TARKOITUS JA TEHTÄVÄ

Opinnäytetyön tavoitteena on lisätä tietoa Node-RED-ohjelmointityökalun käyttömahdollisuuksista ja soveltuvuudesta web-sovelluksen kehittämistyössä. Opinnäytetyön tarkoituksena on luoda web-sovellus Node-RED-ohjelmointityökalulla. Opinnäytetyön tuotoksen tehtävänä on kuvata prosessia siitä, kuinka Node-RED-ohjelmointityökalu soveltuu web-sovelluksen kehittämistyöhön.

3 PERINTEINEN WEB-OHJELMOINTI

Web-ohjelmoinnilla tarkoitetaan verkkoselaimessa toimivien sovelluksien kehittämistä. Perinteisesti web-sovellus koostuu niin kutsutusta frontend-osasta eli käyttöliittymästä, backend-osasta eli palvelinohjelmasta sekä jostakin dataa tarjoavasta ja säilyttävästä osasta, joka on useimmiten tietokanta. (Kaipiainen 2016, 16.) Yleisimmin nämä osat kommunikoivat toistensa kanssa asiakas-palvelin-mallissa (client-server) (Fielding 2000, 45). Palvelin ja käyttöliittymä kommunikoivat web-sovelluksessa keskenään yleensä HTTP-protokollan (Hypertext Transfer Protocol) välityksellä. Käyttöliittymä eli asiakas lähettää palvelimelle tiettyyn osoitteeseen datan hakua tai tallentamista käsitteleviä pyyntöjä, joihin palvelin vastaa. Näille pyynnöille on määritelty tyyppi, kuten POST (tiedon tallentaminen), PUT (tiedon päivittäminen), GET (tiedon hakeminen) tai DELETE (tiedon poistaminen). (Kaipiainen 2016, 16.)

3.1 Palvelinohjelma

Web-sovelluksen backend-osa eli palvelinohjelma tarkoittaa palvelimen päässä suoritettavaa pyyntöjen käsittelyä. Fieldingin (2000, 45) mukaan Andrews (1991) kuvailee palvelinohjelman olevan päättymätön prosessi, joka odottaa pyyntöjä asiakkaalta ja pyynnön saatuaan vastaa näihin kutsuihin. Yleensä palvelinohjelmia on yksi, joka palvelee useampia asiakkaita. Tavallisesti palvelinohjelma toimii rajapintana asiakkaan ja tietokannan välillä. Sen tehtäviin kuuluvat esimerkiksi datan haku ja tallentaminen tietokantaan sekä datan muuttaminen käyttöliittymälle sopivaan muotoon. (Kaipiainen 2016, 16.) Palvelinohjelman kehitykseen käytettäviin teknologioihin kuuluvat muun muassa Java, Python ja Node.js.

3.2 Käyttöliittymä

Frontend-osa on se web-sovelluksen käyttäjälle näkyvä osa, jonka kanssa käyttäjä kommunikoi muodostaen asiakkaan puolen (client side) web-sovelluksesta. Käytännössä tämä tarkoittaa graafista käyttöliittymää. (Wales 2020.) Yleensä tätä web-sovelluksen osaa käyttävät verkkoselaimet (Sinisalo 2017). Frontend-

kehitykseen voidaan mieltää kuuluvaksi kaikki web-sovelluksen käyttöliittymän suunnitteluun ja toteutukseen liittyvät aspektit. Perinteisiä frontend-kieliä ovat HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) ja JavaScript. (Wales 2020.) HTML:llä määritellään web-sovelluksen rakenne, CSS:llä määritellään web-sovelluksen tyylit, teemat ja esitystavat ja JavaScript:lla hallitaan eri elementtien käyttäytymistä ja tehdään web-sovelluksesta interaktiivinen käyttäjän kanssa (Kolowich 2020). Näiden pääkielien lisäksi käyttöliittymäkehitykseen liittyvät vahvasti erilaiset käyttöliittymäkehitykseen tarkoitetut viitekehykset ja kirjastot kuten Bootstrap, AngularJS, jQuery ja Ajax. Viitekehyksien ja kirjastojen tarkoitus on tarjota kehittäjille laadukasta sisältöä hyödyllisessä ja aikaa säästävässä muodossa. (Wales 2020.)

3.3 Ohjelmointirajapinta

API (Application Programming Interface) tarkoittaa sovellusohjelmointirajapintaa. Se on ohjelmistojen välinen rajapinta, joka määrittelee miten sovellukset kommunikoivat keskenään, mitä tietoa API tarjoaa missäkin muodossa ja missä tietoa on saatavilla. API:t auttavat tarjoamaan palveluntarjoajien tarkasti rajoitettuja palveluita, dataa ja toiminnallisuuksia sovelluskehittäjien käyttöön. (Brajesh 2017.)

API:ja on olemassa käyttöjärjestelmille, sovelluksille sekä internetille (web API). Nykyään kuitenkin, kuten tässäkin opinnäytetyössä, API:ista puhuttaessa viitataan yleisesti web API:hin, jotka on tehty käyttäen REST-tyyliä (Representational State Transfer). (Brajesh 2017.) REST on ohjelmistoarkkitehtuurinen tyyli, joka vaalii komponenttien skaalautuvuutta, niiden itsenäisyyttä sekä rajapintojen yhdenmukaisuutta. REST:ssa käytetään asiakas-palvelin-mallia, jossa yhdenmukainen kaikille asiakkaille tarjottava palvelinohjelma pidetään tilattomana eli sen ei tarvitse tietää asiakkaiden tilasta mitään. Sama pätee myös toiseen suuntaan – asiakkaan session tila eriytetään palvelimen tilasta. REST:ssa asiakkaan toistuviin pyyntöihin vastattu data voidaan tallentaa välimuistiin tehokkuuden ja skaalautuvuuden parantamiseksi. (Fielding 2000.) Maailmanlaajuisesti suosittuja API:ja ovat muun muassa lentotietoja tarjoava Skyscanner Flight Search sekä tämänkin opinnäytetyön tuotoksessa käytetty säätietoja tarjoava

Open Weather Map. Kotimaisia API:ja löytyy esimerkiksi digi- ja väestötietoviraston vastaamasta avoindata.fi-palvelusta, joka on tarkoitettu kaikelle kotimaisen avoimen datan julkaisemiselle ja hyödyntämiselle.

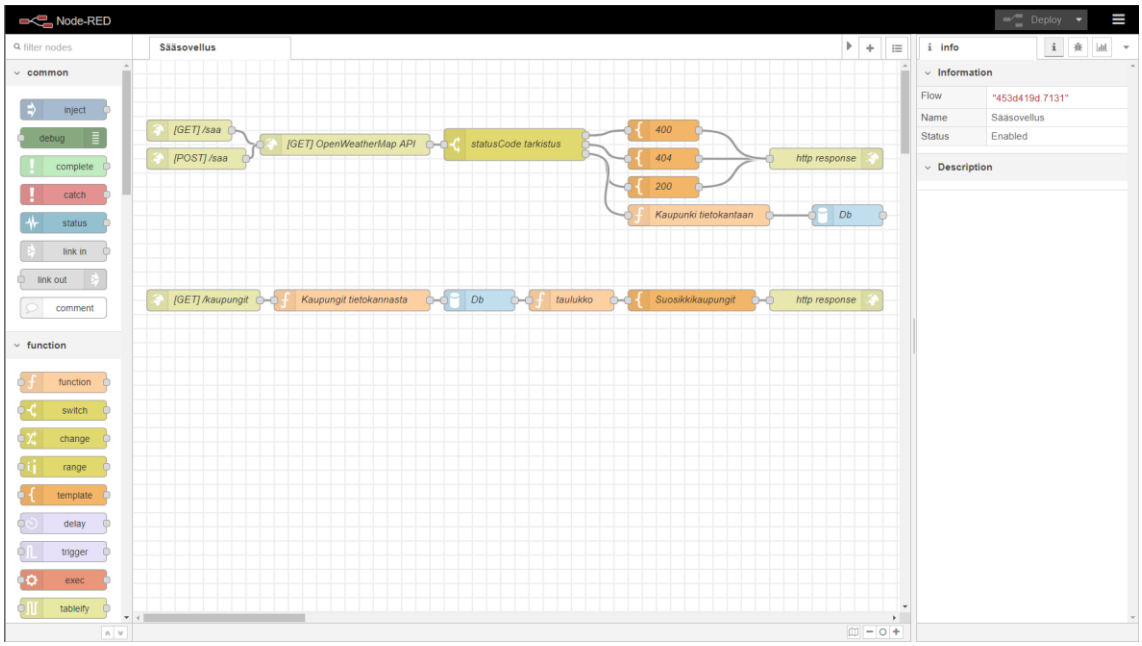
4 TIETOVUO-OHJELMOINTI

Tietovuo-ohjelmointi (flow-based programming) on J. Paul Morrisonin 1960-luvun lopulla kehittämä ohjelmoinnin suuntaus. Sen mukaan sovellukset ovat niin kutsuttujen mustien laatikoiden muodostamien prosessien verkostoja. Nämä prosessit kommunikoivat keskenään ennalta määritettyjä reittejä pitkin kulkevien datapakettien avulla. Reittejä voidaan muuttaa prosesseihin koskematta, jolloin samoilla prosesseilla voidaan kehittää erilaisia sovelluksia. Näin ollen tietovuo-ohjelmointia voidaan pitää komponenttiohjelmointina (component-oriented programming). Suuntauksen on todettu muun muassa nopeuttavan sovelluskehitystä ja parantavan sovelluksien ylläpidettävyyttä, uudelleenkäytettävyyttä, suorituskykyä sekä prototyyppien nopeaa kehittämistä. (Morrison n.d.) Tässä opinnäytetyössä käsitellään tietovuo-ohjelmointiin perustuvaa Node-RED-ohjelmointityökalua.

4.1 Node-RED

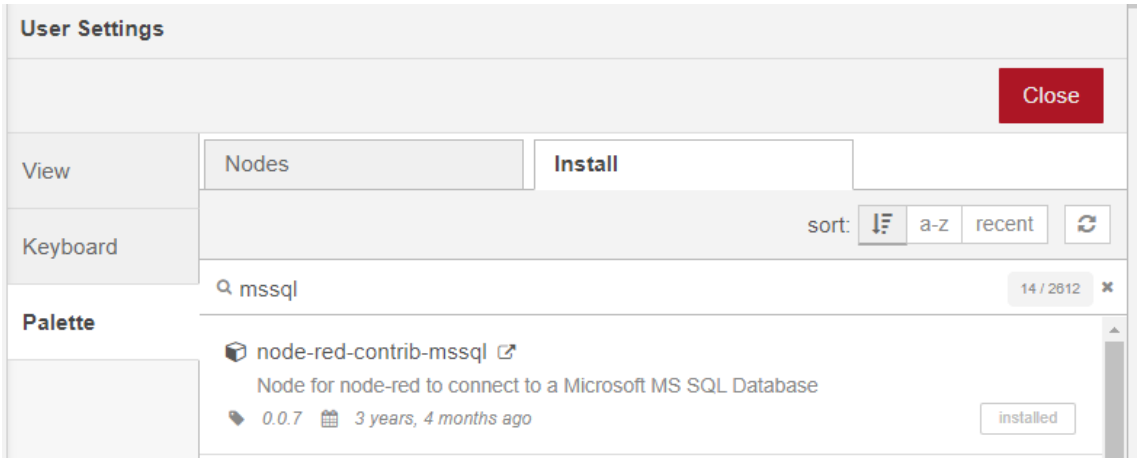
Node-RED on alun perin IBM:n vuonna 2013 kehittämä visuaalinen ohjelmointityökalu, joka pohjautuu J. Paul Morrisonin kehittämän tietovuo-ohjelmoinnin tapaan kuvata sovelluksen käyttäytymistä laatikoiden verkostoina. Nykyään Node-RED on avoimen lähdekoodin projekti osana OpenJS Foundationia. (Node-RED n.d.a) OpenJS Foundation on avoimen lähdekoodin JavaScript-projekteja hallinnoiva säätiö, jonka projekteihin kuuluvat muun muassa jQuery, Node.js, Dojo, Appium, Electron sekä webpack (OpenJS Foundation n.d.). Node-RED-ohjelmointityökalussa data kulkee näiden solmuiksi (node) kutsuttujen laatikoiden läpi. Jokaisella solmulla on tarkkaan määritelty tarkoitus. Solmulle annetaan dataa, solmu käsittelee dataa halutulla tavalla ja tämän jälkeen data siirtyy seuraavaan solmuun. Toteutusta pystyy tarkastelemaan ja ymmärtämään karkealla tasolla kuitenkaan ymmärtämättä jokaista solmujen sisältämää koodiriviä. Tämä malli antaa hyvät edellytykset toteutuksien visuaaliseen esittämiseen ja kehittämiseen sekä laajentaa Node-RED-ohjelmointityökalun potentiaalista käyttäjäkuntaa. (Node-RED n.d.a)

Node-RED tarjoaa selainpohjaisen visuaalisen editorin sovellusten kehittämiseksi (kuva 1). Node-RED-ohjelmointityökalussa sovelluksen logiikka hahmotetaan kuvassa 1 keskellä näkyvässä ruudukossa eli flow-näkymässä. Flow-näkymiä voi olla useissa eri välilehdissä ja ne koostuvat solmuista, jotka ovat Node-RED-komponentteja. Solmuja vedetään kuvassa 1 vasemmalla näkyvästä palkista flow-näkymään. Solmujen väliset yhteydet määritellään vetämällä viivoja solmujen sisään- ja ulostulopisteiden välillä. Flow-näkymän solmuilla ei ole yhteistä tilaa – solmujen ainoa tapa kommunikoida keskenään on lähettää dataa solmujen välillä. Solmujen sisäiset määritykset sekä editorin yleiset asetukset asetetaan kuvassa 1 oikealla näkyvässä palkissa. Lisäksi tähän palkkiin saa näkyville muun muassa sovellusten ajonaikaiset viestit testausta sekä virheiden etsimistä varten.



KUVA 1. Node-RED-ohjelmointityökalun editori-ikkuna selaimessa

Flow-näkymiä voidaan jakaa tuomalla ja viemällä niitä editori-ikkunoihin JSON-tiedostoina. Lisäksi Node-RED sisältää tuhansia käyttäjäyhteisön tekemiä solmuja sisällään pitävän kirjaston. Näitä solmuja lataamalla käyttäjä voi laajentaa omaa solmupalettiaan tarpeidensa mukaan (Node-RED n.d.a). Kuvassa 2 esitetään Node-RED:n solmujen latausominaisuutta esimerkkihaun avulla. Solmuja voi myös kehittää itse JavaScript-kielellä.



KUVA 2. Solmupalettien latausominaisuus esimerkkihaulla "mssql"

Node-RED suunniteltiin alun perin teolliseen IoT-tarpeeseen (Internet of Things) ohjelmointityökaluksi, jolla pystyttäisiin yksinkertaistamaan systeemien ja sensoreiden yhdistämistä toisiinsa sekä kehittämään konseptitodistuksia (proof of concept) pikaisesti. Nykyään Node-RED:iä työstävän ydinryhmän lisäksi Node-RED:llä on kattava käyttäjä- ja kehittäjäyhteisö ja siitä on tullut laajemminkin hyödynnettävä työkalu. (O'Leary 2016; Rodger 2016.) IoT-sovellusten lisäksi Node-RED:iä voi hyödyntää muun muassa web-sovelluksissa, sosiaalisen median sovelluksissa sekä erilaisten tietoteknisten tehtävien hallinnassa. Node-RED on nopeasti kehittyvä teknologia, jota edelleenkehitetään jatkuvasti. (Rodger 2016.)

5 OPINNÄYTETYÖN TOTEUTUS

5.1 Metodologia

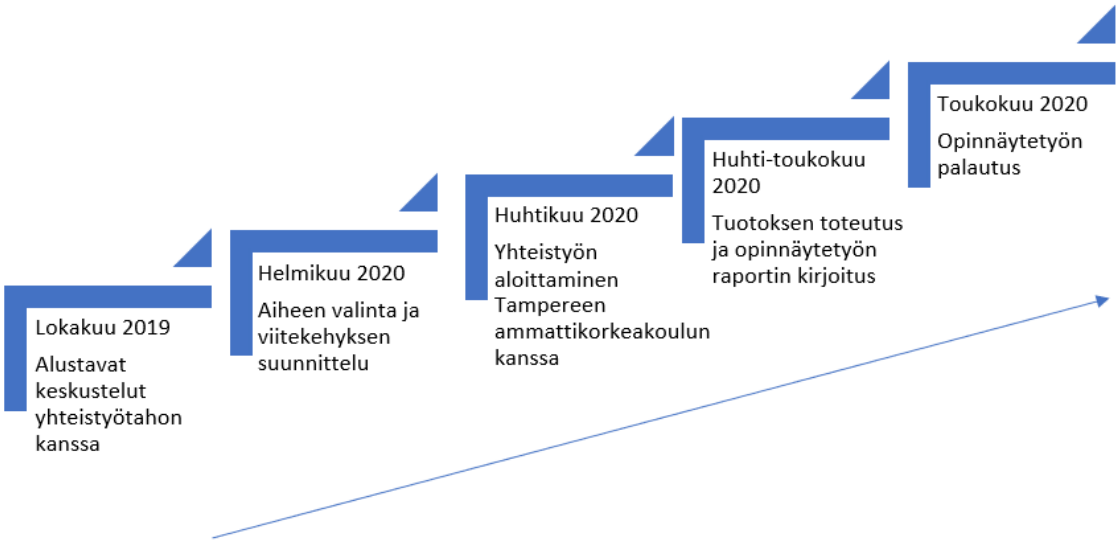
Opinnäytetyön metodi on toiminnallinen opinnäytetyö. Toiminnallisen opinnäytetyön tavoitteena on luoda tuotos, kuten web-sovellus, joka on hyödynnettävissä tuotokseen liittyvällä alalla (Vilkka & Airaksinen 2004, 6, 14). Tuotoksen lisäksi opinnäytetyön suorittamisen vaatimuksena on laatia opinnäytetyöraportti, jonka on oltava perustellusti jäsennelty tehtävänannon mukainen kokonaisuus. Toiminnallisessa opinnäytetyössä yhdistyy siten käytäntö, teoria sekä tutkiva ote raportoinnissa ja työskentelyssä. (Vilkka & Airaksinen 2004, 6–8, 14–19.) Opinnäytetyön tuotoksen ja raportin osalta on huomioitu yllä mainitut toiminnallisen opinnäytetyön kriteerit.

Toiminnallisen opinnäytetyön toteutustavaksi valitaan parhaiten opinnäytetyön kohderyhmää palveleva muoto, jonka kriteereitä ovat informatiivisuus, johdonmukaisuus, asiasisällön sopivuus kohderyhmälle sekä käytettävyys kyseisessä kohderyhmässä. Lisäksi opinnäytetyö rakentuu jo tunnetun tiedon ja opitun taidon päälle eli opiskelija pääsee syventämään omaa ammattitaitoaan asiantuntijuudeksi sekä kykenee ratkaisemaan ammatillisen kentän haasteita. (Vilkka & Airaksinen 2004, 17, 65.) Työelämälähtöinen opinnäytetyö ammattikorkeakoulussa tukee työelämän asiantuntijuuden kehittymistä sekä ammatillista kasvua (Rissanen 2003, 242). Opinnäytetyö on kirjoitettu olettaen, että kohderyhmänä ovat ohjelmistotekniikan alan opiskelijat ja muut alan asiantuntijat, jolloin lukijalla voidaan olettaa olevan tietty lähtötietotaso raporttia lukiessaan.

Opinnäytetyössä tulee lisäksi kuvata ja perustella sujuvasti käytettyjä menetelmiä sekä toteutustapaa. Lisäksi opiskelijan tulee osoittaa käyttämiensä menetelmien hallintaa siten, että hän kykenee niitä opinnäytetyössään johdonmukaisesti ja konkreettisesti soveltamaan. (Tampereen Ammattikorkeakoulu 2019, 2.) Opinnäytetyön raportissa on pyritty kuvaamaan kattavasti ja selkeästi opinnäytetyöprosessin aikana käytettyjä menetelmiä sekä pyritty tuomaan niitä esille huomioimalla käytännön ratkaisut sekä ongelmanratkaisukeskeinen työote.

5.2 Opinnäytetyöprosessin kuvaus

Toiminnallisessa opinnäytetyössä raportin tulee käsitellä työprosessin etenemistä (Vilkka & Airaksinen 2003, 65). Opinnäytetyön prosessi on kuvattu kuviossa 1. Tampereen ammattikorkeakoulu valikoitui yhteistyökumppaniksi kevään 2020 aikana ja opinnäytetyön aihe vahvistui saman vuoden huhtikuussa. Valittu aihe näyttäytyi opinnäytetyön tekijän näkökulmasta opettavaisena ja mielenkiintoisena tutkimuskohteena. Opinnäytetyön tuotoksena on web-sovellus.



KUVIO 1. Opinnäytetyön prosessia kuvaava aikajana

5.3 Web-sovelluksen kuvaus ja arviointi

Opinnäytetyön tuotos on sääpalveluna toimiva web-sovellus, jonka avulla on tutkittu Node-RED-ohjelmointityökalun soveltuvuutta web-sovelluksen kehittämistyössä. Opinnäytetyön tuotoksen tehtävänä on kuvata prosessia siitä, kuinka Node-RED-ohjelmointityökalu soveltuu web-sovelluksen kehittämistyöhön. Opinnäytetyön tuotoksena kehitettyä sääsovellusta voidaan tarpeen mukaan hyödyntää jatkossa yhtenä esimerkkitapana web-sovelluksen kehittämisessä Node-RED-ohjelmointityökalulla.

5.3.1 Web-sovelluksen suunnittelu

Web-sovelluksen käyttöliittymäsuunnittelu on tärkeää, sillä sovelluksen käyttäjän kannalta käyttöliittymä on web-sovelluksen keskeisin osa (Nevalainen & Silavuori 2020, 16). Nevalaisen ja Silavuoren (2020, 16) mukaan Sinkkonen ym. (2006) toteavat, että web-sovellusta suunniteltaessa tulee miettiä tiedon esittämistapaa ja määrää sekä sovelluksen järjestystä, hierarkiaa, rytmitystä, estetiikkaa sekä asioiden näkyvyyttä. Selkeyttä opinnäytetyön tuotoksessa tavoiteltiin suunnittelemalla helppokäyttöinen ja responsiivinen web-sovellus, jonka käyttö ei vaadi harjoitusta ja jonka tarjoama tieto on yksiselitteistä. Responsiivisuus tarkoittaa sitä, että käyttöliittymä mukautuu eri laitteiden erilaisiin ruudun kokoihin (Almeida & Monteiro 2017). Opinnäytetyön tuotosta suunniteltaessa huomioitiin edellä mainittujen lisäksi sovelluksen toimintavarmuus. Toimintavarmuus tarkoittaa tietokonejärjestelmän kykyä toimia virheellisestä datasta tai suorituksesta huolimatta (Tieteen termipankki 2016).

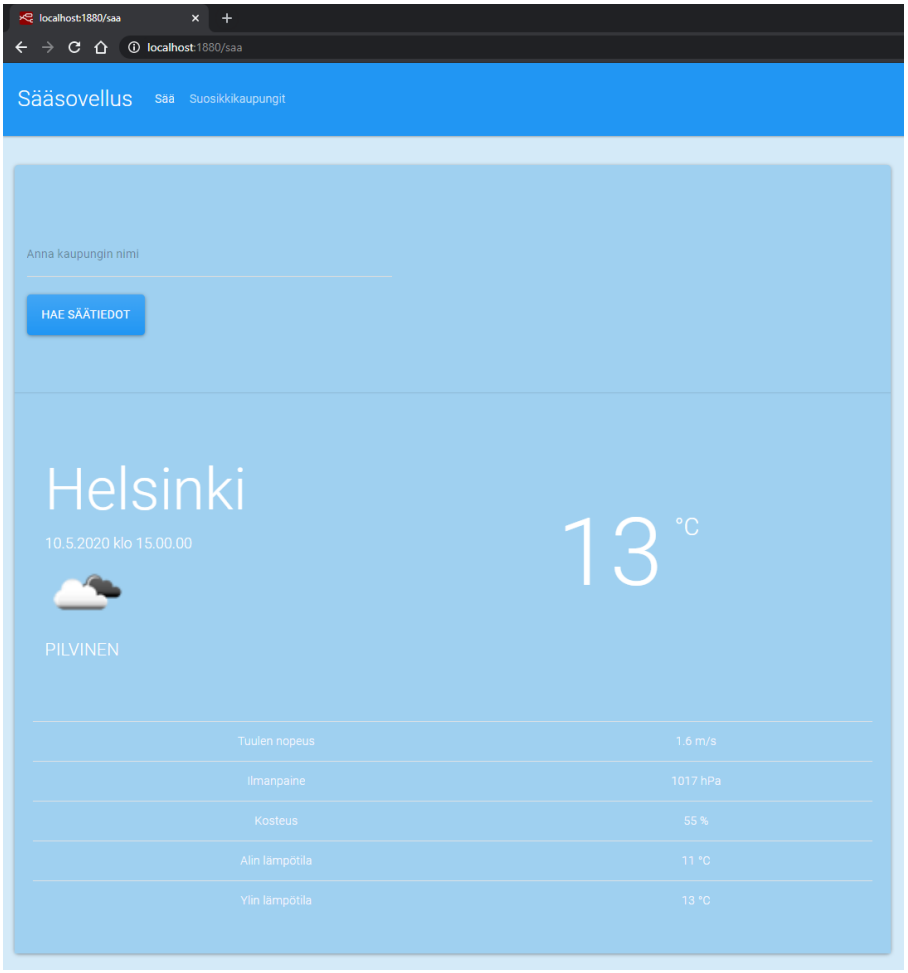
Opinnäytetyön tuotoksena luotiin sääsovellus Node-RED-ohjelmointityökalulla. Sääsovelluksia voidaan pitää ominaisuuksiensa puolesta yksinkertaisina, sillä tiedon pyytäminen ja sen kulkeminen lähteestä selaimeen on suoraviivaista eikä tietoa juurikaan muuteta tai muuten käsitellä tällä matkalla. Tämä tukee ajatusta niin toimintavarmasta kuin selkeästä sovelluksesta. Toimintavarmuus ja selkeys eivät kuitenkaan vielä takaa hyvää käytettävyyttä. Käytettävyys ilmaisee sitä, kuinka helppoa tuotteen tai sovelluksen käyttö on. Hyvän käytettävyyden osa-alueita ovat tyytyväisyys eli kuinka miellyttävää sovellusta on käyttää, tehokkuus eli kuinka nopeasti sovellus suorittaa käyttäjän antamia tehtäviä, muistettavuus eli kuinka hyvin palaavat käyttäjät muistavat sovelluksen sekä opittavuus eli kuinka helposti käyttäjät oppivat käyttämään sovellusta (Nielsen 2003). Sääsovellus suunniteltiin ottamalla huomioon nämä hyvän käytettävyyden osa-alueet.

5.3.2 Web-sovelluksen toteutus

Tässä luvussa käsitellään web-sovelluksen toteutusta. Aluksi esitellään kehitetyn sääsovelluksen toimintaa, jonka jälkeen on kuvattu web-sovelluksen teknin-nen toteutus. Tähän kuuluvat Node-RED-ohjelmointityökalun asennus, tietokannan alustaminen sekä sääsovelluksen toteutus kyseisellä työkalulla.

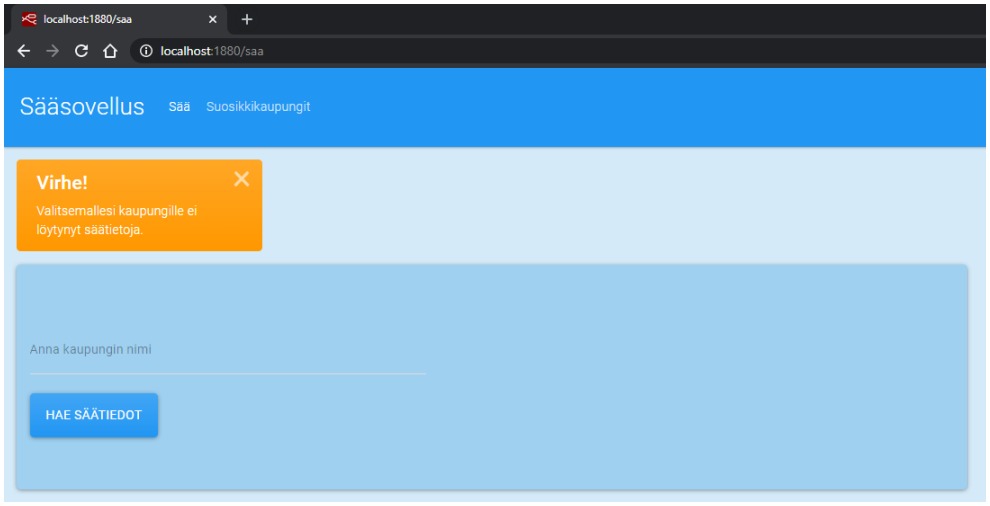
Web-sovelluksen toiminta

Web-sovellus koostuu kahdesta eri näkymästä. Sääpalveluna toimiva päänäkymä luotiin URL-osoitteeseen (Uniform Resource Locator) </saa> (kuva 3). Näkymä koostuu yläosan navigointipalkista, josta löytyvät navigointilinkit molempiin näkymiin. Lisäksi päänäkymässä on korttielementti, joka sisältää tekstinsyöttökentän kaupungin nimeä varten sekä painikkeen säätietojen hakemista varten. Kun säätiedot haetaan onnistuneesti, näytetään selaimessa kaupungin säätiedot.



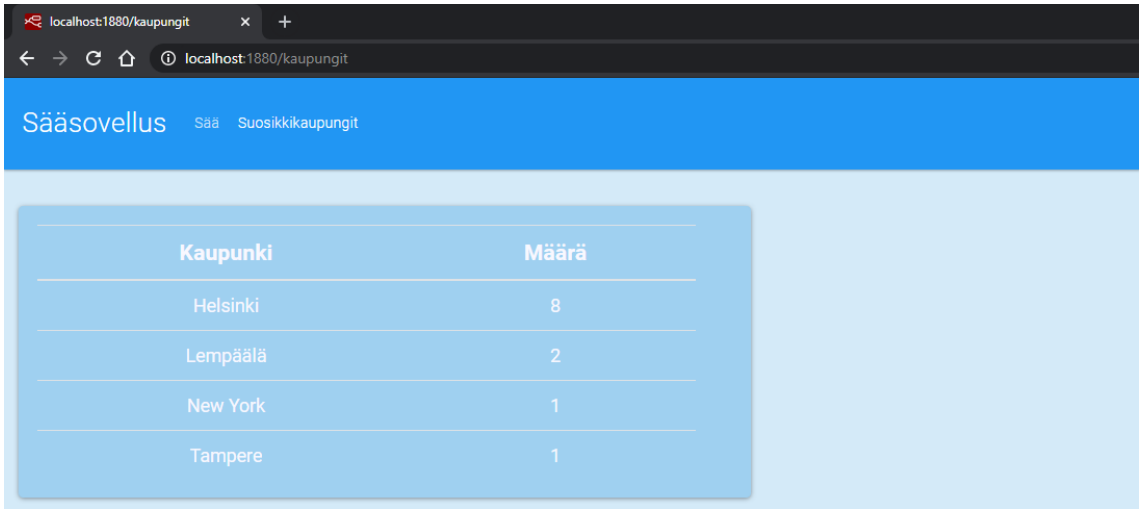
KUVA 3. Sääsovelluksen päänäkymä

Mikäli käyttäjä yrittää hakea säätietoja kaupungin nimellä, jota OpenWeather-Map API ei tunnista, säätietojen sijaan näytetään virheviesti (kuva 4).



KUVA 4. Tuntemattomasta kaupungin nimestä annettava virheviesti

Navigointipalkin Suosikkikaupungit-linkistä voi siirtyä web-sovelluksen toiseen näkymään osoitteeseen </kaupungit>. Näkymä näyttää taulukon, joka kertoo kuinka monta kertaa minkäkin kaupungin säätietoja on haettu (kuva 5). Ylimpänä taulukossa olevaa kaupunkia on haettu useimmiten.



KUVA 5. Suosikkikaupungit-näkymä

Node-RED-ohjelmointityökalun asennus

Node-RED-ohjelmointityökalun asentamiseen on saatavilla erilaisia vaihtoehtoja erilaisille ympäristöille. Paikallisesti tietokoneeseen asentamisen lisäksi työkalun voi ottaa käyttöön esimerkiksi Amazon Web Services- tai IBM Cloud-pilvipalveluissa sekä Raspberry Pi- tai Android-laitteissa (Node-RED n.d.c). Tietokoneelle paikallisesti asennettaessa Node-RED-ohjelmointityökalu vaatii toimiakseen Node-RED:n tukeman version Node.js-palvelinympäristöstä, joka pitää sisällään sille tehdyn npm-paketinhallintatyökalun (Node Package Manager) (Node-RED n.d.e). Tietokoneelle Node-RED-ohjelmointityökalua paikallisesti asennettaessa luonnollisin tapa on käyttää juurikin npm:ää. Kuvasta 6 nähdään Node-RED:n asentamiseen käytettävä komento npm-paketinhallintatyökalulla.

```
sudo npm install -g --unsafe-perm node-red
```

KUVA 6. Node-RED-ohjelmointityökalun asennus npm-paketinhallintatyökalulla (Node-RED n.d.e)

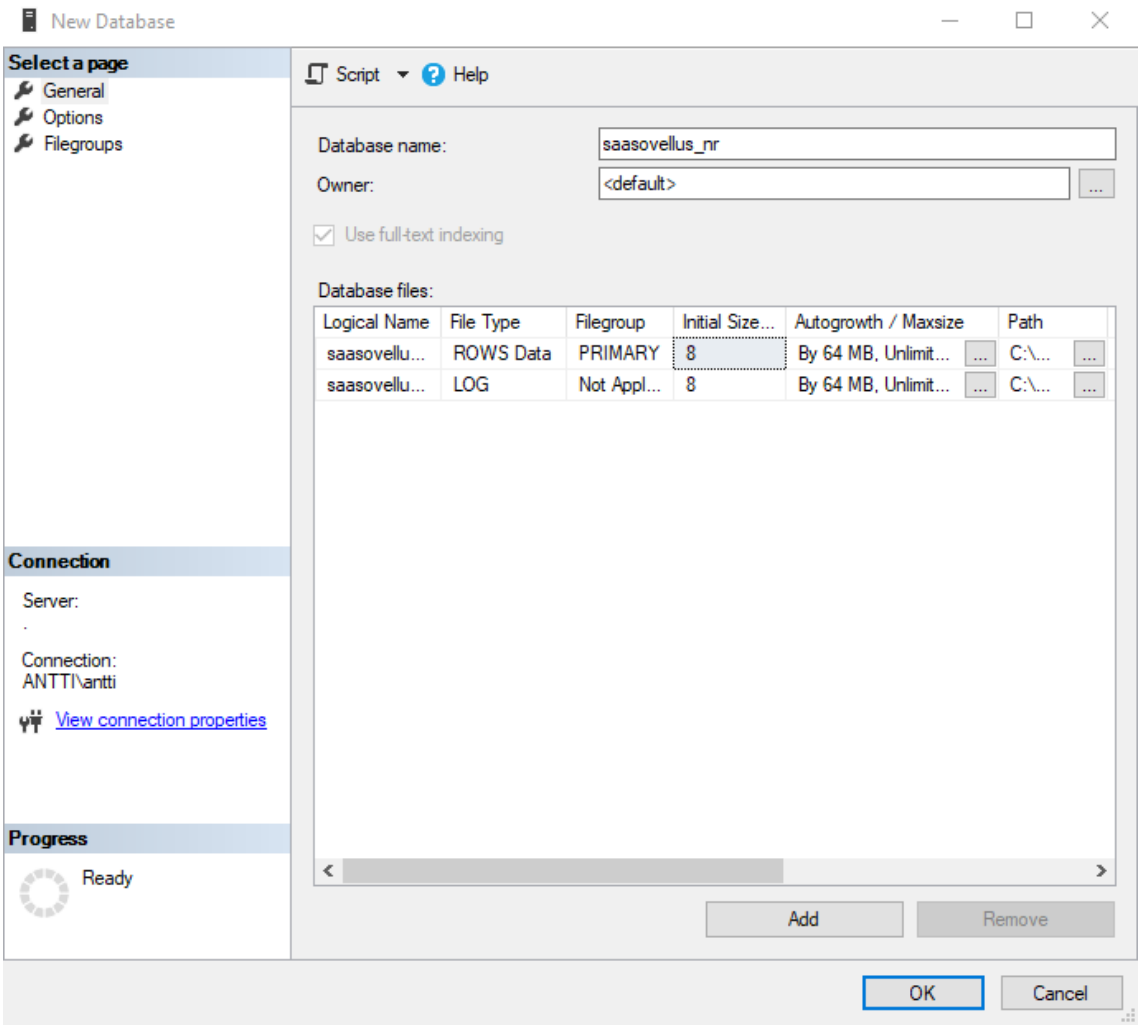
Asennuksen jälkeen työkalu käynnistetään komentokehoteikkunassa komennolla ”node-red” (kuva 7). Tämän jälkeen Node-RED-työkalu on käytettävissä selaimen osoittessa <http://localhost:1880>. Työkalun toiminta vaatii, että komentokehoteikkuna, johon käynnistyskomento on annettu, pidetään auki työkalun käytön aikana (Node-RED n.d.e).

```
C:\>node-red
```

KUVA 7. Node-RED-ohjelmointityökalun käynnistämiskomento

Tietokannan alustaminen

Tietokannan hallintaan ja ylläpitämiseen valittiin Microsoft SQL Server -tietokannan hallintajärjestelmä. Web-sovellusta varten tulee luoda tietokanta sekä tarvittavat taulut. Tietokantaa luodessa voidaan määrittää useita erilaisia asetuksia, mutta tärkeimpänä on asettaa tietokannalle nimi sekä omistaja-roolin saava käyttäjä (kuva 8).



KUVA 8. Uuden tietokannan luonti SQL Server -tietokantajärjestelmässä

Tietokannan luomisen jälkeen tietokantaan luotiin Kaupungit-taulu (kuva 9). Tauluun määriteltiin sarakkeiksi jokaiselle kaupungille oma numerotunnus (id), nimi sekä määrä, kuinka monta kertaa kaupungin säätietoja on haettu.

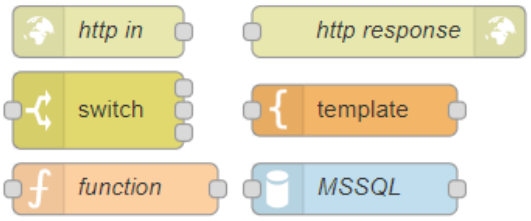
```
CREATE TABLE Kaupungit (
    Id int NOT NULL IDENTITY,
    Nimi varchar(255) NOT NULL,
    Maara int
);
```

KUVA 9. SQL-komento Kaupungit-taulun luomiseen

Sääsovelluksen luominen Node-RED-ohjelmointityökalulla

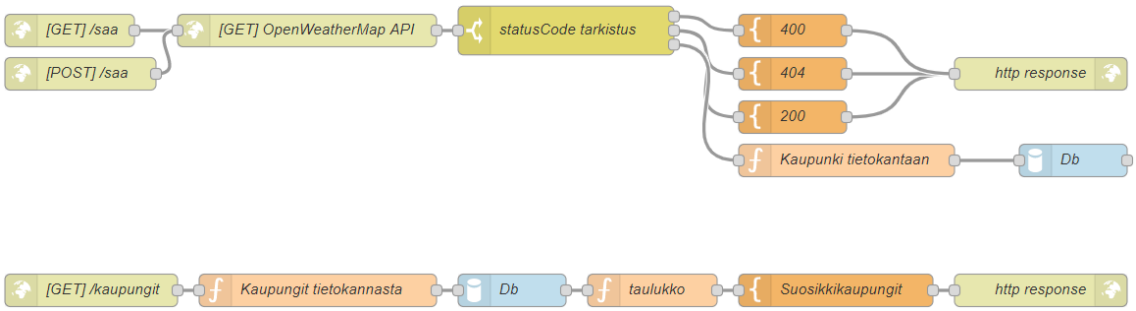
Sääsovelluksen luomiseen tarvittiin kuutta erilaista solmua (kuva 10). Http in -solmulla luodaan HTTP-päätepiste (endpoint), joka määritellään kuuntelemaan

HTTP-pyyntöjä tietyssä URL-osoitteessa. Tälle solmulle täytyy luoda http response -solmu vastapariksi vastaamaan näihin pyyntöihin. Switch-solmulla voidaan tarkastella sille annettavaa dataa ja ehdollisesti määritellä datan liikkuminen riippuen tarkasteltavasta tiedosta. Template-solmu on tyhjä tekstipohja, jossa voidaan näyttää sille annettua dataa halutulla tavalla. Tämän web-sovelluksen tapauksessa haluttu tapa on HTML-koodi käyttöliittymän luomista varten. Function-solmun avulla voidaan käsitellä ja muokata sille annettua dataa JavaScript-kielellä. MSSQL-solmun avulla luodaan tietokantayhteys antamalla sille tarvittavat MS SQL Server -tietokantajärjestelmän yhteystiedot.



KUVA 10. Sääsovelluksen kehittämisessä käytetyt solmut oletusnimineen

Kuva 11 näyttää sääsovelluksen kokonaisrakenteen Node-RED-ohjelmointityökalussa. Ylemmässä puoliskossa viivoilla yhdistetyssä solmurakenteessa on luotu sääsovelluksen päänäkymä. Vastaavasti alemmassa puoliskossa on luotu Suosikkikaupungit-näkymä. Oletuksena data liikkuu solmuissa vasemmalta oikealle. Molempien näkymien kehityksen aluksi luotiin http in -solmut [GET] /saa ja [GET] /kaupungit. Node-RED-ohjelmointityökalun taustalla toimiva Node.js-palvelin kuuntelee pyyntöjä osoitteessa <localhost:1880>. Kun selaimella siirrytään URL-osoitteeseen <localhost:1880/saa>, lähetetään palvelimelle GET-tyypin HTTP-pyyntö tähän osoitteeseen. Jotta palvelin osaisi vastata tähän pyyntöön, luotiin http in -solmulle vastapariksi http response -solmu. Sama tehtiin Suosikkikaupungit-näkymälle osoitteessa <localhost:1880/kaupungit>. Tämän opinnäytetyön liite 1 sisältää sääsovelluksen kokonaisrakenteen JSON-tekstinä tuotuna Node-RED-ohjelmointityökalusta. Ennen sääsovelluksesta tuomista siitä poistettiin säätietojen hakemiseen tarvittava henkilökohtainen API-avain sekä tietokannan yhteystiedot.



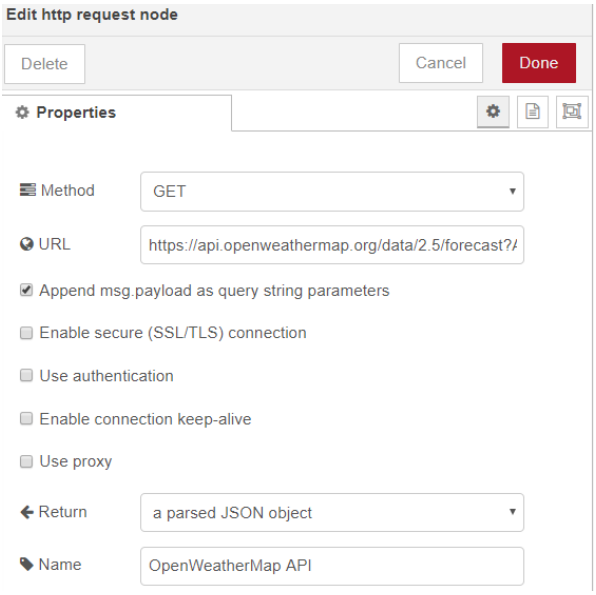
KUVA 11. Sääsovelluksen rakenne kokonaisuudessaan Node-RED-ohjelmointityökalussa

Päänäkymän template-solmuihin luotiin lomake-elementti (kuva 12), joka ”Hae säätiedot” -painiketta painamalla lähettää POST-tyypin HTTP-pyynnön osoitteeseen </saa>. Tämä lomake sisältää muuttujan q, jonka arvo on tekstinsyöttökenttään annettu nimi (esimerkiksi q=Helsinki).

```
<form method="POST" action="/saa">
    <span>
        <label for="city"</label>
    </span>
    <div style="background-color: #9fd0f0; padding-top: 3em; padding-bottom: 2em;">
        <input class="form-control" id="city" name="q" placeholder="Anna kaupungin nimi">
        </br>
        <button type="submit" class="btn btn-primary">Hae säätiedot</button>
    </div>
</form>
```

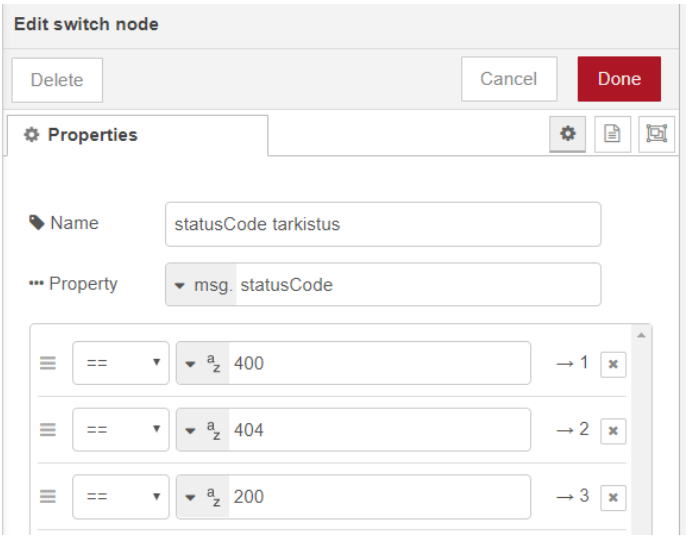
KUVA 12. Lomakekoodi kaupungin nimen lisäämiseksi säätietojen hakupyyntöön

Kaupungin nimi sijoitettiin muuttujaan q, sillä OpenWeatherMap API:n dokumentaation mukaan URL-osoitteen q-parametri luetaan kaupungin nimenä. Näin ollen lomakkeen muuttuja q ja sen arvo lisättiin sellaisenaan [GET] OpenWeatherMap API -solmussa lähetettävään GET-tyypin pyynnön URL-osoitteeseen, joka palauttaa säätiedot JSON-oliona (kuva 13).



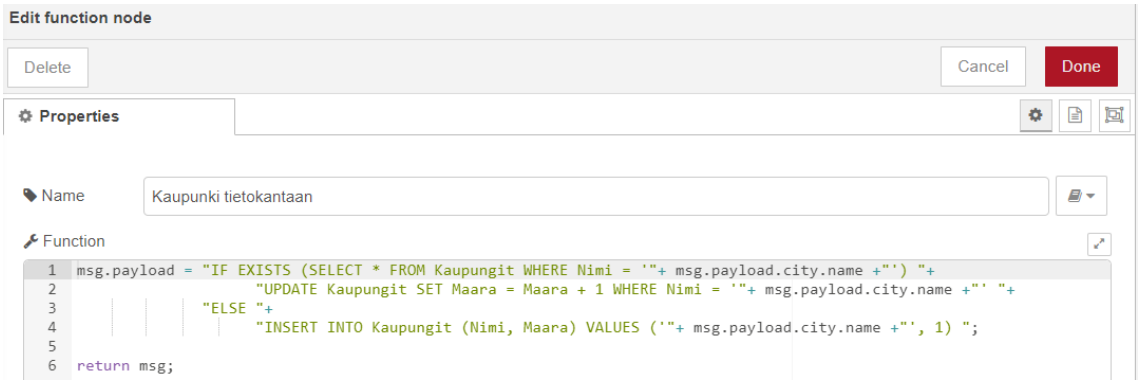
KUVA 13. [GET] OpenWeatherMap API -solmun määritys

Säätiedot sisältävä JSON-olio sisältää statusCode-nimisen nimi-arvo-parin, jonka arvo on jokin HTTP status koodi. Koodin arvo tarkastetaan säätietojen hakua seuraavassa switch-solmussa (kuva 14). Mikäli annetun kaupungin nimellä löytyy säätietoja, on tämän koodin arvo 200 (OK). Tällöin data johdatetaan 200-nimiseen template-solmuun, jonka HTML-koodi sisältää navigointipalkin, korttielementin säätietojen hakua varten sekä itse säätiedot. Mikäli säätietoja haetaan ilman kaupungin nimeä, on koodin arvo 400 (Bad Request). Tällöin käytetään 404-nimistä template-solmua, jonka HTML-koodi sisältää ainoastaan navigointipalkin sekä korttielementin säätietojen hakua varten. Mikäli säätietoja haetaan kaupungin nimellä, jota OpenWeatherMap API ei tunnista, on koodin arvo 404 (Not Found). Tällöin käytetään 400-nimistä template-solmua, jonka HTML-koodi sisältää edellisen lisäksi virheviestin virheellisen kaupungin nimen merkiksi.



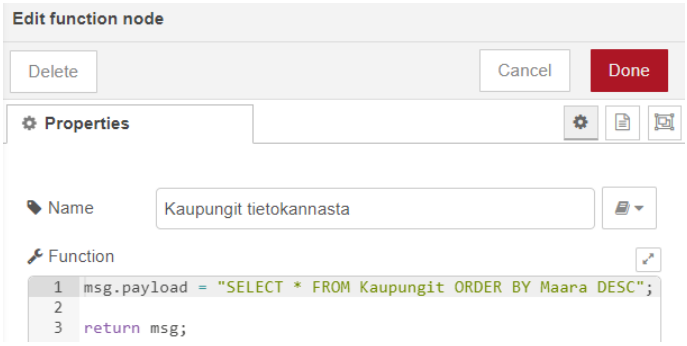
KUVA 14. Switch-solmu säätietohaun status koodin tarkastusta varten

Mikäli status koodi on 200 eli säätiedot haettiin onnistuneesti, Kaupunki tietokantaan -nimisessä function-solmussa luotiin tietokantakomento kaupungin nimen lisäämiseksi tietokannan Kaupungit-tauluun (kuva 15). Tämä komento tarkastaa onko taulussa jo rivi kyseisellä kaupungin nimellä ja jos on, nostaa kyseisen kaupungin Maara-sarakkeen kokonaislukua yhdellä. Mikäli kaupungin nimellä ei löydy riviä Kaupungit-taulusta, lisätään kaupungin nimi uudelle riville ja kaupungin Maara-sarakkeeseen asetetaan kokonaisluku yksi.



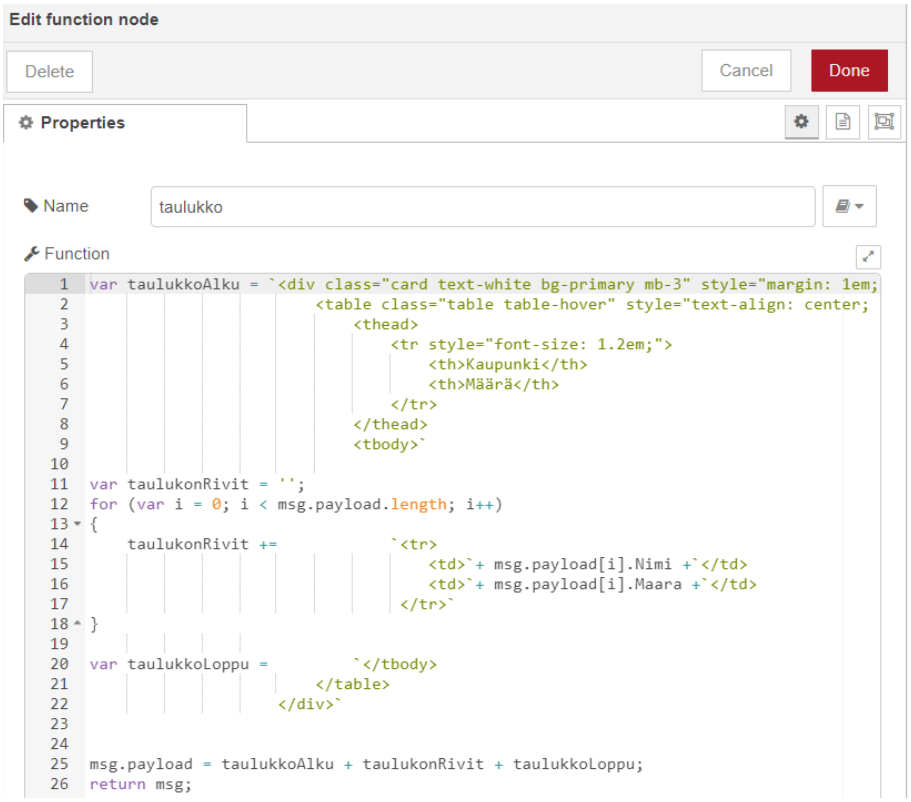
KUVA 15. Kaupungin nimen tietokantaan lisäävän tietokantakomennon luominen

Suosikkikaupungit-näkymään siirryttäessä käytettiin niin ikään function-solmua kaupungit tietokannasta hakevan tietokantakomennon luomiseksi (kuva 16). Tämä komento hakee kaikkien kaupunkien kaikki tiedot tietokannan Kaupungit-taulusta ja asettaa ne laskevaan järjestykseen Maara-sarakkeen kokonaislukujen mukaan. Tietokantajärjestelmä palauttaa kaupunkien tiedot JSON-oliona.



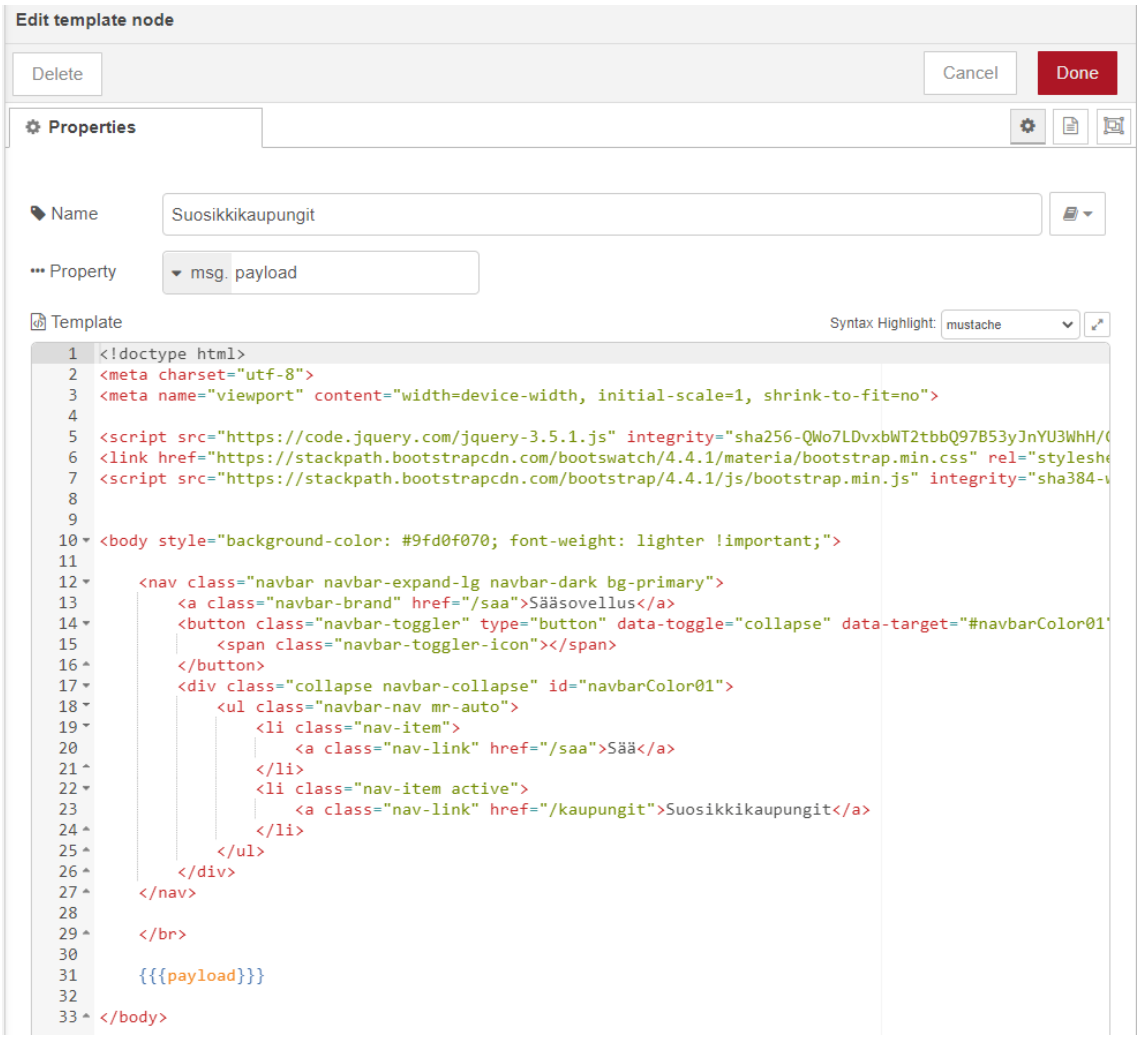
KUVA 16. Kaupunkien tiedot hakeva tietokantakomento function-solmussa

Tietokannasta saatujen kaupunkien tiedot lisättiin Suosikkikaupungit-näkymän taulukkoon. Kaupunkien tietoja sisältämää JSON-oliota ei kuitenkaan voitu tarjoilla sellaisenaan Suosikkikaupungit-näkymän HTML-koodiin, koska template-solmujen käyttämä mustache-pohja (template) on logiikaton. Käytännössä logiikaton pohja esti JSON-olion datan läpikäymisen niin, että kaupunkien nimet ja hakumäärät olisivat pystytty valitsemaan kerralla taulukkomuodossa esittämistä varten. Näin ollen Suosikkikaupungit-nimisen template-solmun ja tietokantasolmun väliin lisättiin taulukko-niminen function-solmu, jossa JavaScript-kielellä luotiin HTML-koodin mukainen merkkijono (string) taulukkomuodon saavuttamiseksi (kuva 17).



KUVA 17. Suosikkikaupungit-näkymän taulukon HTML-koodin luominen

Merkkijono sijoitettiin Suosikkikaupungit-näkymän käyttöliittymäkoodiin mustache-pohjan mukaisesti (kuva 18), jolloin selain näyttää merkkijonossa luodun taulukon.



KUVA 18. Suosikkikaupungit-näkymän käyttöliittymäkoodi

5.3.3 Web-sovelluksen arviointi

Opinnäytetyön tuotoksena kehitetty web-sovellus suunniteltiin ottamalla huomioon tiedon esitystapa ja määrä, järjestys, hierarkia, rytmitys, estetiikka sekä asioiden näkyvyys. Lisäksi tavoitteena oli kehittää helppokäyttöinen, selkeä, responsiivinen sekä toimintavarma sovellus, jossa on huomioitu myös sovelluksen hyvä käytettävyys.

Käyttöliittymäsuunnittelun lähtökohdaksi otettiin Bootstrap-kirjaston hyödyntäminen web-sovelluksessa. Bootstrap on alun perin Twitter:in kehittämä maailman suosituin web-kehityksessä käytetty käyttöliittymäkirjasto, jonka tarkoituksena on nopeuttaa ja helpottaa kaikilla selaimilla toimivien responsiivisten käyttöliittymien kehitystä (Andrea 2014, 32; Moreto 2016, 1). Lähes kaikki sääsovelluksen käyttöliittymän elementit ovat Bootstrap-elementtejä. Tätä kirjastoa käyttämällä saavutettiin käyttöliittymän hyvä responsiivisuus, selkeys ja estetiikka sekä selkeä tiedon esittämistapa.

Bootstrap-kirjasto otettiin käyttöön CDN-verkkoa (Content Delivery Network) käyttäen. CDN-termiä käytetään toisiinsa yhdistetyistä tietokoneista muodostuvasta verkostosta, jonka tarkoituksena on tarjota sisältöä suorituskykyisesti hyvällä saatavuudella (Moreto 2016, 7). CDN-verkkoa käyttäen kirjasto saatiin nopeasti ja helposti käyttöön. Tätä käyttöönottotapaa hyödyntäen ei kuitenkaan voitu muokata Boostrap-kirjaston sisältämiä tiedostoja. Tätä varten Bootstrap-kirjasto olisi jouduttu lataamaan osaksi sääsovelluksen kansiorakennetta, jolloin olisi voitu muokata yksittäisiä tyylejä ja elementtejä suoraan lähdekoodin sisältävistä tiedostoista, joissa ne on määritelty. Sääsovelluksessa käytettyjä Bootstrap-elementtejä muokattiin kuitenkin tarpeen mukaan HTML-koodin sisällä style-attribuuttia käyttäen. HTML-koodin selkeyden sekä luettavuuden puolesta olisi ollut perusteltua ladata Bootstrap-kirjasto osaksi projektin kansiorakennetta, mutta sääsovelluksen näkymien elementtien määrän ollessa vähäinen valittiin Bootstrap-kirjaston nopeampi ja helpompi käyttöönottotapa CDN-verkkoa hyödyntäen.

Bootstrap-kirjaston lisäksi responsiivisuuden saavuttamiseksi käyttöliittymän kehittämisessä hyödynnettiin myös CSS:n mediasääntöjä (@media rules). Mediasääntöjen avulla web-sovelluksen rakenne saadaan mukautumaan eri kokoisten ja tyyppisten laitteiden ruuduille sopivaksi (Eygi 2020). Mediäsääntöjä käytettiin sääsovelluksen päänäkymässä rakenteen selkeyttämiseksi käyttäjän laitteen ruudun koon ollessa pieni. Responsiivisuuden huomioiminen sääsovelluksessa paransi tiedon esittämistä, järjestystä, estetiikkaa sekä näkyvyyttä.

Sääsovelluksen yksinkertaisen luonne palveluna varmistaa sovelluksen hyvän käytettävyyden ominaisuuksista tehokkuuden sekä opittavuuden. Käyttäjän

kanssa interaktiivisia elementtejä on verrattain vähän, mikä osaltaan tukee sääsovelluksen opittavuutta sekä muistettavuutta. Sääsovelluksen käyttö ei aiheuta raskaita toimenpiteitä, joten tehokkuuden pitäminen hyvän käytettävyyden tasolla on vaivatonta. Toisaalta sääsovelluksen yksinkertainen luonne ja toiminta perustelee tiedon määrän vähäisyyttä ja antaa omalta osaltaan edellytykset sovelluksen jatkokehittämiseen. Esimerkiksi OpenWeatherMap API:n tarjoamat sääennusteet tuleville päiville lisäisivät sääsovelluksen tarjoaman tiedon määrää sekä käyttäjän tarpeista riippuen käyttäjän tyytyväisyyttä.

6 POHDINTA

6.1 Node-RED-ohjelmointityökalun hyödynnettävyys web-sovelluksen kehittämisessä

Yksi Node-RED-ohjelmointityökalun merkittävimmistä seikoista web-sovelluksen kehittämisessä on web-sovelluksen backend-osan teknologian näkymättömyys kehittäjälle. Node-RED on kehitetty Node.js-palvelinympäristön päälle, joten web-sovelluksia kehitettäessä backend-teknologialle ei ole muita vaihtoehtoja. Taustalla toimiva Node.js-palvelinympäristö tulee Node-RED:n mukana työkalulle oletusasetuksin valmiiksi määritettynä, mutta Node-RED:n oman kansiorakenteen sisäisiä tiedostoja muokaten sen asetuksia voi kuitenkin uudelleenkonfiguroida. Osittain tai projektin tarpeista riippuen kokonaan valmista backend-teknologia voidaan pitää projektista riippuen niin hyvänä kuin huonona ominaisuutena.

Toinen selkeä ero perinteiseen web-ohjelmointiin verrattuna on perinteisen web-ohjelmistoprojektin kansiorakenteen puute Node-RED:ssä. Perinteisessä web-ohjelmistoprojektissa voi esimerkiksi palvelinohjelma olla sijoitettu server-kansion alle ja käyttöliittymä client-kansion alle. Nämä kansiot sisältävät eri tyyppisiä tiedostoja, esimerkiksi client-kansion sisällä voi olla HTML- ja CSS-tiedostoja. Node-RED:iä uudelleenkonfiguroimalla on kyllä mahdollista määritellä kansiorakenne, jonka sisältö otetaan luodun sovelluksen ajon aikana huomioon, mutta itse Node-RED:n editori-ikkunassa ei tiedostoihin ole pääsyä, sillä erilaisia tiedostoja ei ole. Node-RED:n itsensä taustalla on oma kansiorakenne, mutta flow-näkymissä kehitetyt projektit kuten web-sovellukset koostuvat ainoastaan JSON-tiedostosta tai -tiedostoista, jotka löytyvät Node-RED:n omasta kansiorakenteesta. Perinteisen web-ohjelmistoprojektin kansiorakennetta voikin verrata Node-RED:n flow-näkymiin ja kansioiden sisältämiä tiedostoja Node-RED:n solmuihin. Tämän opinnäytetyön tuotoksena kehitetyn sääsovelluksen kohdalla kansiorakenteen puute aiheutti esimerkiksi sen, että käyttöliittymän CSS-koodia ei voitu selkeästi eriyttää vaan se sisällytettiin HTML-koodiin sekä style-attribuutein että <style>-tagia käyttäen. Tämä heikensi käyttöliittymän HTML-koodin selkeyttä, luettavuutta ja ylläpidettävyyttä.

Kansiorakenteen puutteen johdosta Node-RED ei välttämättä ole optimaalinen ohjelmointityökalu kehitettäessä laajoja web-ohjelmistoprojekteja. Tätä seikkaa tukee myös Node-RED:n editori-ikkunan koon rajallisuus. Editori-ikkunaa saa loitonnettua, lähennettyä ja siirreltyä, mutta monimutkaisia kokonaisuuksia rakennettaessa flow-näkymistä voi tulla epäselviä. Node-RED:ssä on kuitenkin mahdollisuus tiivistää yhdessä flow-näkymässä olevia useamman solmun kokonaisuuksia yhdeksi solmuksi, joita kutsutaan nimellä subflow (Node-RED n.d.f). Subflow-solmu ilmestyy omana solmunaan solmupalettiin ja sitä voi käyttää kuten normaaleja solmuja. Vaikka useamman subflow-solmun käyttäminen poistaisi flow-näkymien epäselvyyttä, voi niiden piilottaman logiikan näkymättömyys olla haitaksi kehitykselle. Toisaalta logiikan hahmottaminen monimutkaisissa perinteisin menetelmin kehitetyissä web-ohjelmistoprojekteissa voi olla niin ikään haasteellista laajan kansiorakenteen ja suuren tiedostomäärän takia.

Node-RED kehitettiin alun perin teollisen IoT:n tarpeisiin ohjelmointityökaluksi, jolla voitaisiin kehittää konseptitodistuksia pikaisesti. Vaikka Node-RED on monipuolistunut tästä alkulähtökohdasta, näkyy se vieläkin työkalun nopeassa käyttöönotossa ja matalassa oppimiskäyrässä. Nämä seikat tukevat erityisesti maltillisen kokoisten web-sovellusten tehokasta kehittämistä ilman merkittävän pohjatiedon määrän omaksumista.

Tämän opinnäytetyön tuotoksena luodun web-sovelluksen lähtökohdaksi otettiin web-sovelluksen kehittäminen vain frontend-kehityksen pääkieliä sekä Bootstrap-kirjastoa käyttäen. Frontend-kehitykseen tarkoitettuja viitekehyksiä ei hyödynnetty, sillä haluttiin luoda konseptitodistus Node-RED-ohjelmointityökalun soveltuvuudesta web-sovelluksen kehittämisessä käyttäen perinteisiä frontend-kehityksen pääkieliä. Myös opinnäytetyön aikataulu sekä aiheen sopiva rajaus asettivat rajoituksia käytettyjen teknologioiden implementoinnissa. Frontend-kehitykseen tarkoitettujen viitekehyksien hyödynnettävyys kehitettäessä web-sovelluksia Node-RED-ohjelmointityökalulla vaatisikin lisää tieteellistä tutkimusta.

Opinnäytetyön aikataulu sekä aiheen sopiva rajaus asettivat niin ikään rajoituksia otettaessa huomioon web-sovelluksen kehittäminen Node-RED-ohjelmointityökalulla tuotannollisesta näkökulmasta. Node-RED tarjoaa ominaisuuksia,

jotka puoltavat sen käytettävyyttä tuotannollisessa web-kehityksessä. Yhtenä tällaisena tärkeänä ominaisuutena voidaan pitää mahdollisuutta ottaa projektit käyttöön Node-RED:n asetuksista. Ottamalla projektit käyttöön mahdollistetaan Node-RED:n ja Git-versionhallintatyökalun integraatio Node-RED-ohjelmointityökalussa. Käytännössä tämä paljastaa uusia asetuksia ja välilehtiä Node-RED-ohjelmointityökalussa, joihin kuuluvat muun muassa mahdollisuus nähdä omat paikallisen version muutokset versionhallinnassa olevaan projektin versioon verrattuna, näiden lisääminen versionhallintaan sekä versionhallintaan tehtyjen muutosten historian tarkastelu suoraan Node-RED-ohjelmointityökalussa (Node-RED n.d.f). Node-RED:ssä on myös tietoturvaan, lokitukseen ja reitityksiin liittyviä asetuksia. Lisäksi yksikkötestausta varten Node-RED:ssä voi ottaa käyttöön sitä varten luodun oman npm-moduulin (Node-RED n.d.c). Tämän opinnäytetyön pohjalta voidaan todeta, että Node-RED monipuolisena ohjelmointityökaluna soveltuu myös web-sovelluksien kehittämiseen. Node-RED-ohjelmointityökalun hyödynnettävyys tuotannollisessa web-kehityksessä vaatii kuitenkin lisätutkimusta.

6.2 Opinnäytetyön arviointi

Opinnäytetyön tulee olla innovatiivinen, työelämää kehittävä, ajankohtainen, työelämän tarpeista lähtevä sekä opiskelijan omaa ammatillista osaamista kehittävä. Opinnäytetyö tulee rajata tarkoituksenmukaisesti ja sen tavoite ja tarkoitus on asetettava ja perusteltava johdonmukaisesti. (Tampereen ammattikorkeakoulu 2019, 1.)

Opinnäytetyön teoreettisten lähtökohtien sekä lähteiden käytön osalta opinnäytetyössä käytettyihin lähteisiin tulee olla perehtynyt laaja-alaisesti, lähteiden on oltava luotettavia ja ajantasaisia sekä sisällettävä myös kansainvälisiä lähteitä. Lisäksi opinnäytetyössä käytetyt käsitteet tulee määritellä kattavasti. (Tampereen ammattikorkeakoulu 2019, 1.) Opinnäytetyössä on käytetty sekä kansainvälisiä että kansallisia tutkimuksia, artikkeleita, verkkolähteitä sekä kirjallisuutta. Opinnäytetyön aihealueen tiimoilta osoittautui haastavaksi löytää luotettavana pidettäviä ajankohtaisia lähteitä johtuen todennäköisesti alan jatkuvasti kehittyvästä luonteesta. Opinnäytetyöhön on kuitenkin onnistuttu löytämään kattavasti

relevantteja lähteitä. Opinnäytetyössä on pyritty määrittelemään käytetyt käsitteet riittävällä tarkkuudella lukijan ymmärryksen mahdollistamiseksi.

Opinnäytetyön toteutustavan, tekemisen, opinnäytetyöprosessin sekä eettisyyden ja luotettavuuden osalta opinnäytetyöraportissa tulee kuvata opinnäytetyössä käytetyt menetelmät, työn toteutustapa sekä perustella ja soveltaa tekemiään valintoja sujuvasti. Opinnäytetyöprosessin tulee edetä suunnitellussa aikataulussa ja ohjausta tulee käyttää tarkoituksenmukaisesti. Lisäksi opinnäytetyön tekemisessä tulee osoittaa innovatiivisuutta, itsenäisyyttä ja pohtia opinnäytetyön luotettavuuteen sekä tutkimusetiikkaan liittyviä periaatteita. (Tampereen ammattikorkeakoulu 2019, 2.) Opinnäytetyössä käytetyt menetelmät, työn toteutustapa sekä prosessin aikana tehdyt valinnat on esitetty raportissa kattavasti ja loogisessa järjestyksessä. Opinnäytetyöprosessin aloituksen viivästymisen johdosta opinnäytetyöprosessin aikana ohjauksen mahdollisuutta ei hyödynnetty suunnitellusti. Opinnäytetyön aihevalinnassa sekä toteutuksessa on pyritty osoittamaan itsenäisyyttä, innovatiivisuutta sekä kiinnitetty huomiota tutkimuseettisiin periaatteisiin.

Opinnäytetyön tulosten, johtopäätösten, tuotoksen sekä pohdinnan osalta opinnäytetyössä tulee kuvata sitä, kuinka opinnäytetyön tuotos, tulokset ja johtopäätökset vastaavat asetettuihin tavoitteisiin ja tehtäviin. Opinnäytetyön tulee tuottaa uutta käytännöllistä tietoa sekä osoittaa kriittisyyttä, luovuutta, innovatiivisuutta sekä kykyä soveltaa teoriatietoa. Opinnäytetyön pohdintaosiossa tulee esittää mahdollisia jatkotutkimusaiheita. (Tampereen ammattikorkeakoulu 2019, 3.) Opinnäytetyön tavoitteena on lisätä tietoa Node-RED-ohjelmointityökalun käyttömahdollisuuksista ja soveltuvuudesta web-sovelluksen kehittämistyössä. Opinnäytetyö lisää ajankohtaista ja tutkittua tietoa Node-Red-ohjelmointityökalun käytettävyydestä web-sovelluksen kehittämisessä tuomalla esille monipuolisesti eri näkökulmia työkalun hyödyistä sekä haasteista web-sovelluksen kehittämistyössä. Tarkoituksena on luoda web-sovellus Node-RED-ohjelmointityökalulla. Opinnäytetyössä on tuotettu käyttäjän valitseman kaupungin säätietoja tarjoava web-sovellus Node-Red-ohjelmointityökalun avulla. Opinnäytetyön tuotoksen tehtävänä on kuvata prosessia siitä, kuinka Node-RED-ohjelmointityökalu soveltuu web-sovelluksen kehittämistyöhön. Opinnäytetyössä on kuvattu moni-

puolisesti ja eri näkökulmista tarkastellen sitä prosessia, kuinka Node-RED-ohjelmointityökalua on mahdollista hyödyntää web-sovelluksen kehittämistyössä. Lisäksi opinnäytetyössä on koottu yhteen opinnäytetyössä rajattuun aiheeseen liittyviä kansallisia ja kansainvälisiä tutkimuksia, jolloin opinnäytetyö muodostaa aihealueen tiimoilta tiivistetyn kuvan Node-Red-ohjelmointityökalusta. Opinnäytetyön pohdintaosiossa on nostettu esille kehittämiskohteita, joita käyttämällä voitaisiin tutkia muun muassa kyseisen ohjelmointityökalun hyödynnettävyyttä web-sovelluksen kehityksessä hyödyntämällä moderneja käyttöliittymäkehittämiseen tarkoitettuja viitekehyksiä.

Opinnäytetyön kirjallisen raportin tulee olla tyyliltään ja rakenteeltaan viimeistelty sekä ehjä kokonaisuus, jossa otsikot kuvaavat sisältöä tarkasti. Opinnäytetyöraportin tulee olla kirjoitettu TAMKin kirjallisen raportoinnin ohjeistuksen mukaisesti sekä olla tyyliltään hyvää asiatekstiä. Opinnäytetyöraportti tulee tarvittaessa esitellä monipuolisesti tuomalla esille ammatillista kypsyyttä. (Tampereen ammattikorkeakoulu 2019, 4.) Opinnäytetyön kirjallinen raportti on tyyliltään asiatekstiä, jossa otsikointi on harkittu vastaamaan tarkasti tekstisisältöä. Raportti on laadittu TAMKin kirjallisen raportoinnin ohjeistuksen mukaisesti.

6.3 Opinnäytetyön hyödynnettävyys sovelluskehittäjän näkökulmasta

Ammattikorkeakoulussa työelämälähtöiseltä opinnäytetyöltä odotetaan käytännöllisyyttä ja hyödynnettävyyttä työelämässä sekä opiskelijan oman asiantuntijuuden kehittymistä ja ammatillista kasvua (Rissanen 2003, 241–242). Tällä hetkellä Tampereen ammattikorkeakoulun ohjelmistotekniikan alan opetussuunnitelmaan ei juurikaan sisälly opintoja koskien visuaalista ohjelmointia tai tietovuo-ohjelmointia, joten ne jäävät vaihtoehtoisina tyyleinä opiskelijan oman kiinnostuksen sekä työelämästä ja harjoitteluista saatujen kokemusten varaan. Tämä opinnäytetyö lisää siten omalta osaltaan opiskelijan oman asiantuntijuuden kehittymistä sekä laajentaa näkemystä ohjelmistotekniikan monipuolisuudesta. Lisäksi opinnäytetyön tuotos ja raportti on toteutettu käytännönläheisesti ja huomioimalla hyödynnettävyys työelämässä.

Työelämälähtöinen opinnäytetyö harjoittaa opiskelijan kyvykkyyttä toimia monipuolisissa työyhteisön konteksteissa, mikä vaatii osaltaan perinteisen asiantuntijatiedon rajan ylittämistä (Rissanen 2003, 256). Ohjelmistotekniikan teknologioiden sekä työelämän jatkuvasti muuttuessa tulee myös sovelluskehittäjän ylläpitää ja kehittää omaa ammattitaitoaan sekä olla valmis sopeuttamaan omaa työnkuvaansa. Tämä opinnäytetyö lisää lukijan tietoa Node-RED-ohjelmointityökalun hyödynnettävyydestä web-sovelluksen kehittämisessä ja tuo siten esille uusia vaihtoehtoisia näkökulmia perinteisen web-ohjelmoinnin rinnalle.

6.4 Opinnäytetyön jatkokehittämisehdotukset

Opinnäytetyöprosessin pohjalta nousi esille muutamia jatkokehittämisehdotuksia. Opinnäytetyön lähdeaineistoa etsittäessä haasteeksi nousi niin kansainvälisten kuin kotimaistenkin tieteellisten tutkimusten vähäinen saatavuus. Aihealueen tiimoilta on löydettävissä verkkoartikkeleita sekä ohjelmointiaiheisia sivustoja, mutta niiden luotettavuutta voidaan pitää kyseenalaisena vertaisarviointien puuttuessa. Näin ollen opinnäytetyöstä nousi esille tarve saada luotettavaa tieteellistä tutkimustietoa tietovuo-ohjelmoinnista sekä esimerkiksi Node-RED-ohjelmointityökalun käytettävyydestä erityisesti web-kehityksessä.

Opinnäytetyön tuotos on toteutettu käyttäen perinteisiä frontend-kieliä. Käyttöliittymäkehitykseen tarkoitettujen viitekehyksien kuten Angular:n käyttö helpottaisi kuitenkin hyvin rakennetun web-sovelluksen tuottamista sekä mahdollistaisi toimintalogiikan suorittamisen osittaisen siirtämisen web-sovelluksen käyttöliittymän puolelle (client side). Tämä toimintatapa on yleistynyt viime vuosina, sillä se parantaa loppukäyttäjän käyttäjäkokemusta poistamalla tarpeen ladata koko web-sivu kerralla. Lisäksi viitekehysten käyttö mahdollistaa sellaisten web-sovellusten luonnin, jotka käyttäytyvät mobiililaitteiden natiivien sovellusten tapaan. (Wikhög 2018, 7.) Tutkittaessa Node-RED-ohjelmointityökalun hyödynnettävyyttä web-sovelluksen kehityksessä olisikin jatkossa optimaalista pyrkiä käyttämään erilaisia moderneja käyttöliittymäkehittämiseen tarkoitettuja viitekehyksiä.

Node-RED-ohjelmointityökalulla on ominaisuuksia ja asetuksia, jotka tukevat ohjelmointityökalun käyttöä tuotannollisessa web-kehityksessä. Node-RED-ohjelmointityökalulla tehtävän web-kehityksen tarkastelu tuotannollisesta näkökulmasta vaatisi kuitenkin lisätutkimusta.

LÄHTEET

Almeida, F. & Monteiro, J. 2017. The Role of Responsive Design in Web Development. Webology 2/2017, 48. https://search-proquest-com.libproxy.tuni.fi/docview/2084838907/fulltextPDF/641331C6E0D14F6FPQ/1?accountid=14242

Andrea, S. 2014. Mastering Magento Theme Design. Create responsive themes using Bootstrap, the most widely used frontend framework. Birmingham: Packt Publishing.

Brajesh, D. 2017. API Management: An Architect's Guide to Developing and Managing APIs for Your Organization. Luettu 16.5.2020. https://master-workshop.skillport.com/skillportfe/assetSummaryPage.action?assetid=RW$5776:_ss_book:128140#summary/BOOKS/RW$5776:_ss_book:128140

Eygi, C. 2020. CSS Media Queries: Breakpoints, Media Types, Standard Resolutions, and More. Luettu 20.5.2020. https://www.freecodecamp.org/news/css-media-queries-breakpoints-media-types-standard-resolutions-and-more/

Fielding, R. 2000. Architectural Styles and the Design of Network-based Software Architectures. Information and Computer Science. University of California, Irvine. Dissertation. https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

Kaipiainen, A. 2016. Funktionaalinen ohjelmointi web-ohjelmistokehityksessä. Tietotekniikan koulutusohjelma. Tampereen teknillinen yliopisto. Diplomityö. https://trepo.tuni.fi/bitstream/handle/123456789/24541/Kaipiainen.pdf?sequence=3&isAllowed=y

Kolowich, L. 2018. Web Design 101: How HTML, CSS, and JavaScript Work. Päivitetty 24.3.2020. Luettu 15.5.2020. https://blog.hubspot.com/marketing/web-design-html-css-javascript

Moreto, S. 2016. Bootstrap 4 By Example. Master Bootstrap 4's frontend framework and build your websites faster than ever before. Birmingham: Packt Publishing.

Morrison, J. P. n.d. Flow-based Programming. Luettu 15.5.2020. https://jpaulm.github.io/fbp/

Nevalainen, N & Silavuori, P. 2020. Käyttäjäystävällisen web-sovelluksen suunnittelu ja toteutus. Tieto- ja viestintätekniikka. Metropolia Ammattikorkeakoulu. Insinöörityö. https://www.theseus.fi/bitstream/handle/10024/339157/nevalainen_nina_silavuori_petra_2020_kayttajaystavallisen_web-sovelluksen_suunnittelu_ja_toteutus.pdf?sequence=2&isAllowed=y

Nielsen, J. 2012. Usability 101: Introduction to Usability. Luettu 16.5.2020. https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Node-RED. n.d.a. About. Luettu 15.5.2020. https://nodered.org/about/

Node-RED. n.d.b. Creating your first node. Unit testing. Luettu 21.5.2020. https://nodered.org/docs/creating-nodes/first-node#unit-testing

Node-RED. n.d.c. Getting Started. Luettu 15.5.2020. https://node-red.org/docs/getting-started/

Node-RED. n.d.d. Projects. Luettu 21.5.2020. https://nodered.org/docs/user-guide/projects/

Node-RED. n.d.e. Running Node-RED locally. Luettu 15.5.2020. https://node-red.org/docs/getting-started/local

Node-RED. n.d.f. Subflows. Luettu 21.5.2020. https://nodered.org/docs/user-guide/editor/workspace/subflows

O'Leary, N. 2016. Moving to the JS Foundation. Luettu 16.5.2020. https://node-red.org/blog/2016/10/17/js-foundation

OpenJS Foundation. n.d. About the OpenJS Foundation. Luettu 15.5.2020. https://openjsf.org/about/

Rissanen, R. 2003. Työelämälähtöinen opinnäytetyö oppimisen kontekstina. Fenomenografisia näkökulmia tradenomin opinnäytetyöhön. Kasvatustieteiden tiedekunta. Tampereen yliopisto. Väitöskirja. https://tampub.uta.fi/bitstream/handle/10024/67321/951-44-5806-0.pdf?sequence=1

Rodger, L. 2016. Node-RED: Lecture 1 – A brief introduction to Node-RED. Luettu 16.5.2020. http://noderedguide.com/nr-lecture-1/

Sinisalo, A. 2017. Web frontend component quality model. Master's Degree Programme in Information Technology. Tampereen teknillinen yliopisto. Master of Science thesis. https://trepo.tuni.fi/bitstream/handle/123456789/25280/sinisalo.pdf?sequence=4&isAllowed=y

Tampereen ammattikorkeakoulu. 2019. AMK-opinnäytetyön arviointikriteerit. Luettu 18.5.2020. https://content-webapi.tuni.fi/proxy/public/2019-08/intra_opinnaytetyoamk_arviointikriteerit_muokattu19062019_19062019.docx

Tieteen termipankki. 2016. Toimintavarmuus. Luettu 17.5.2020. http://tieteentermipankki.fi/wiki/Nimitys:toimintavarmuus

Vilkka, H. & Airaksinen, T. 2003. Toiminnallinen opinnäytetyö. Helsinki: Kustannusosakeyhtiö Tammi.

Wales, M. 2014. 3 Web Dev Careers Decoded: Front-End vs Back-End vs Full Stack. Päivitetty 10.1.2020. Luettu 15.5.2020. https://blog.udacity.com/2014/12/front-end-vs-back-end-vs-full-stack-web-developers.html

Wikhög, M. 2018. Comparing frontend frameworks and cloud services by using House of Quality. Computer Engineering. Department of Computer and System science. Mid Sweden University. Degree of Bachelor. http://miun.diva-portal.org/smash/get/diva2:1190384/FULLTEXT01.pdf

LIITTEET

Liite 1. Sääsovellus JSON-tiedostona

[{"id":"453d419d.7131","type":"tab","label":"Sääsovellus","disabled":false,"info":""},{"id":"ea84a4a8.242c28","type":"template","z":"453d419d.7131","name":"200","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<!doctype html>\n<meta charset=\"utf-8\">\n<meta name=\"viewport\" content=\"width=device-width, initial-scale=1, shrink-to-fit=no\">\n\n<script src=\"https://code.jquery.com/jquery-3.5.1.js\" integrity=\"sha256-QWo7LDvxbWT2tbbQ97B53yJnYU3WhH/C8ycbRAkjPDc=\" crossorigin=\"anonymous\"></script>\n<link href=\"https://stackpath.bootstrapcdn.com/bootswatch/4.4.1/materia/bootstrap.min.css\" rel=\"stylesheet\" integrity=\"sha384-1tymk6x9Y5K+OF0tlmG2fDRcn67QGzBkiM3IgtJ3VrtGrIi5ryhHjKjeeS60f1FA\" crossorigin=\"anonymous\">\n<script src=\"https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js\" integrity=\"sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6\" crossorigin=\"anonymous\"></script>\n\n\n<body style=\"background-color: #9fd0f070; font-weight: lighter !important;\">\n \n <nav class=\"navbar navbar-expand-lg navbar-dark bg-primary\">\n   Sääsovellus\n   <button class=\"navbar-toggler\" type=\"button\" data-toggle=\"collapse\" data-target=\"#navbarColor01\" aria-controls=\"navbarColor01\" aria-expanded=\"false\" aria-label=\"Toggle navigation\">\n   <span class=\"navbar-toggler-icon\">\n   </button>\n   <div class=\"collapse navbar-collapse\" id=\"navbarColor01\">\n   <ul class=\"navbar-nav mr-auto\">\n   <li class=\"nav-item active\">\n   Sää\n   \n   <li class=\"nav-item\">\n   Suosikkikaupungit\n   \n   \n   \n </nav>\n \n</br>\n \n <div class=\"card text-white bg-primary mb-3\" style=\"margin: 1em; max-width: 60rem; background-color: #9fd0f0 !important;\">\n   \n   <div style=\"max-width: 25rem; margin: 1em\">\n   <form method=\"POST\" action=\"/saa\">\n   \n   <label

```
for=\"city\"</label>\n            </span>\n            <div style=\"background-co-
lor: #9fd0f0; padding-top: 3em; padding-bottom: 2em;\">\n                <input
class=\"form-control\" id=\"city\" name=\"q\" placeholder=\"Anna kaupungin
nimi\">\n            </br>\n            <button type=\"submit\" class=\"btn
btn-primary\">Hae säätiedot</button>\n        </div>\n    </form>\n
</div>\n    \n    <hr class=\"my-4\">\n    \n    \n    <div
class=\"card-body\" style=\"background-color: #9fd0f0;\">\n        <div
id=\"city\" style=\"width: 50%; float: left; text-align: left; padding-left: 1em; pad-
ding-bottom: 5em; padding-top: 1em;\">\n            <div style=\"font-size:
4rem;\">{{payload.city.name}}</div>\n            <span id=\"datetime\"
style=\"font-size: 1rem;\">{{payload.list.0.dt_txt}}</span>\n            </br>\n
<div>\n            <img style=\"width: 20%;\" src=\"http://openweather-
map.org/img/w/{{payload.list.0.weather.0.icon}}.png\"/>\n            </div>\n
<span style=\"text-transform: uppercase; font-size: 1.2rem;\">\n
{{payload.list.0.weather.0.description}}\n            </span> \n
</div>\n        <div id=\"temp\" style=\"width: 50%; float: right; padding-top:
4em;\">\n            <span \n            id=\"mainTemp\"\n
style=\"font-size: 8em;\n            width: 50%;\n            float: left;\n
text-align: right;\">\n            {{payload.list.0.main.temp}}\n
</span>\n            <span\n            id=\"degree\"\n
style=\"width: 50%;\n            float: right;\n            text-align: left;\n
padding-top: 1.2em;\n            padding-left: 0.5em;\n            font-size:
2em;\">\n            &deg;C\n            </span>\n        </div>\n
\n        </br>\n        \n        <table class=\"table table-hover\"
style=\"text-align: center; color: ghostwhite\">\n            <tbody>\n
<tr>\n            <td>Tuulen nopeus</td>\n
<td>{{payload.list.0.wind.speed}} m/s</td>\n            </tr>\n
<tr>\n            <td>Ilmanpaine</td>\n
<td>{{payload.list.0.main.pressure}} hPa</td>\n            </tr>\n
<tr>\n            <td>Kosteus</td>\n
<td>{{payload.list.0.main.humidity}} %</td>\n            </tr>\n
<tr>\n            <td>Alin lämpötila</td>\n            <td id=\"min-
Temp\">{{payload.list.0.main.temp_min}} &deg;C</td>\n            </tr>\n
<tr>\n            <td>Ylin lämpötila</td>\n            <td id=\"max-
Temp\">{{payload.list.0.main.temp_max}} &deg;C</td>\n            </tr>\n
```

</tbody>\n </table>\n \n \n \n</body>\n\n<script>\n\n var rawDateTime = document.getElement-ById(\"datetime\").innerText;\n document.getElementById(\"datetime\").in-nerText = formatDateTime(rawDateTime);\n\n var rawMainTemp = docu-ment.getElementById(\"mainTemp\").innerText;\n var rawMinTemp = docu-ment.getElementById(\"minTemp\").innerText;\n var rawMaxTemp = docu-ment.getElementById(\"maxTemp\").innerText;\n rawMinTemp = rawMin-Temp.substring(0, rawMinTemp.indexOf(\" °C\"));\n rawMaxTemp = rawMax-Temp.substring(0, rawMaxTemp.indexOf(\" °C\"));\n document.getElement-ById(\"mainTemp\").innerText = formatTemperature(rawMainTemp);\n docu-ment.getElementById(\"minTemp\").innerText = formatTemperature(rawMin-Temp) + ' °C';\n document.getElementById(\"maxTemp\").innerText = format-Temperature(rawMaxTemp) + ' °C';\n \n\n function formatDateTime(raw-DateTime) {\n return new Date(rawDateTime).toLocaleString();\n }\n\n function formatTemperature(rawTemp) {\n return Number(rawTemp).toFi-xed(0).toString();\n }\n \n</script>\n\n<style>\n @media screen and (max-width: 560px) {\n #city {\n width: 100% !important;\n padding-bottom: 0 !important;\n }\n #temp {\n width: 100% !important;\n padding-top: 0 !important;\n margin: 1em;\n }\n #mainTemp {\n width: 65% !important;\n }\n #degree {\n width: 35% !impor-tant;\n }\n}\n</style>\n","output":"str","x":750,"y":180,"wi-res":[["2ace31a2.280a6e"]]},{"id":"2ace31a2.280a6e","type":"http res-ponse","z":"453d419d.7131","name":"http response","status-Code":"","headers":{},"x":980,"y":140,"wi-res":[]},{"id":"c19206c9.7a75b8","type":"http re-quest","z":"453d419d.7131","name":"[GET] OpenWeatherMap API","met-hod":"GET","ret":"obj","paytoqs":true,"url":"https://api.openweather-map.org/data/2.5/forecast?APPID=&units=metric&lang=fi","tls":"","per-sist":false,"proxy":"","authType":"","x":300,"y":120,"wi-res":[["3074efa2.b16ae"]]},{"id":"3074efa2.b16ae","type":"switch","z":"453d419d.7131","name":"statusCode tarkistus","property":"statusCode","property-Type":"msg","ru-les":[{"t":"eq","v":"400","vt":"str"},{"t":"eq","v":"404","vt":"str"},{"t":"eq","v":"200","vt":"str"}],"checkall":"true","repair":false,"outputs":3,"x":540,"y":120,"wi-res":[["dbd2b44b.0d2f38"],["2b99048b.38c10c"],["ea84a4a8.242c28","b98c8508.

2f94c8"]]},{"id":"2b99048b.38c10c","type":"template","z":"453d419d.7131","name":"404","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<!doctype html>\n<meta charset=\"utf-8\">\n<meta name=\"viewport\" content=\"width=device-width, initial-scale=1, shrink-to-fit=no\">\n\n<script src=\"https://code.jquery.com/jquery-3.5.1.js\" integrity=\"sha256-QWo7LDvxbWT2tbbQ97B53yJnYU3WhH/C8ycbRAkjPDc=\" crossorigin=\"anonymous\"></script>\n<link href=\"https://stackpath.bootstrapcdn.com/bootswatch/4.4.1/materia/bootstrap.min.css\" rel=\"stylesheet\" integrity=\"sha384-1tymk6x9Y5K+OF0tlmG2fDRcn67QGzBkiM3IgtJ3VrtGrIi5ryhHjKjeeS60f1FA\" crossorigin=\"anonymous\">\n<script src=\"https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js\" integrity=\"sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6\" crossorigin=\"anonymous\"></script>\n\n\n<body style=\"background-color: #9fd0f070; font-weight: lighter !important;\">\n \n <nav class=\"navbar navbar-expand-lg navbar-dark bg-primary\">\n   Sääsovellus\n   <button class=\"navbar-toggler\" type=\"button\" data-toggle=\"collapse\" data-target=\"#navbarColor01\" aria-controls=\"navbarColor01\" aria-expanded=\"false\" aria-label=\"Toggle navigation\">\n   <span class=\"navbar-toggler-icon\">\n   </button>\n   <div class=\"collapse navbar-collapse\" id=\"navbarColor01\">\n   <ul class=\"navbar-nav mr-auto\">\n   <li class=\"nav-item active\">\n   Sää\n   \n   <li class=\"nav-item\">\n   Suosikkikaupungit\n   \n   \n   \n </nav>\n \n<div class=\"alert alert-dismissible alert-warning\" style=\"max-width: 60rem; margin: 1em;\">\n <button type=\"button\" class=\"close\" data-dismiss=\"alert\" aria-label=\"Close\">\n   <span aria-hidden=\"true\">×\n </button>\n <h4 class=\"alert-heading\">Virhe!</h4>\n <p class=\"mb-0\">Valitsemallesi kaupungille ei löytynyt säätietoja.</p>\n \n \n <div class=\"card text-white bg-primary mb-3\" style=\"margin: 1em; max-width: 60rem; background-color: #9fd0f0 !important;\">\n   \n   <div style=\"max-width: 25rem; margin: 1em\">\n   <form method=\"POST\" action=\"/saa\">\n   \n   <label

for=\"city\"</label>\n \n <div style=\"background-color: #9fd0f0; padding-top: 3em; padding-bottom: 2em;\">\n <input class=\"form-control\" id=\"city\" name=\"q\" placeholder=\"Anna kaupungin nimi\">\n </br>\n <button type=\"submit\" class=\"btn btn-primary\">Hae säätiedot</button>\n \n </form>\n \n \n \n</body>","output":"str","x":750,"y":140,"wires":[["2ace31a2.280a6e"]]},{"id":"6aae8180.6baad","type":"http in","z":"453d419d.7131","name":"[GET] /kaupungit","url":"/kaupungit","method":"get","upload":false,"swaggerDoc":"","x":100,"y":340,"wires":[["21112d7e.3abf52"]]},{"id":"2ac251cf.3f57fe","type":"template","z":"453d419d.7131","name":"Suosikkikaupungit","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<!doctype html>\n<meta charset=\"utf-8\">\n<meta name=\"viewport\" content=\"width=device-width, initial-scale=1, shrink-to-fit=no\">\n\n<script src=\"https://code.jquery.com/jquery-3.5.1.js\" integrity=\"sha256-QWo7LDvxbWT2tbbQ97B53yJnYU3WhH/C8ycbRAkjPDc=\" crossorigin=\"anonymous\"></script>\n<link href=\"https://stackpath.bootstrapcdn.com/bootswatch/4.4.1/materia/bootstrap.min.css\" rel=\"stylesheet\" integrity=\"sha384-1tymk6x9Y5K+OF0tlmG2fDRcn67QGzBkiM3lgtJ3VrtGrIi5ryhHjKjeeS60f1FA\" crossorigin=\"anonymous\">\n<script src=\"https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js\" integrity=\"sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6\" crossorigin=\"anonymous\"></script>\n\n\n<body style=\"background-color: #9fd0f070; font-weight: lighter !important;\">\n \n <nav class=\"navbar navbar-expand-lg navbar-dark bg-primary\">\n Sääsovellus\n <button class=\"navbar-toggler\" type=\"button\" data-toggle=\"collapse\" data-target=\"#navbarColor01\" aria-controls=\"navbarColor01\" aria-expanded=\"false\" aria-label=\"Toggle navigation\">\n <span class=\"navbar-toggler-icon\">\n </button>\n <div class=\"collapse navbar-collapse\" id=\"navbarColor01\">\n <ul class=\"navbar-nav mr-auto\">\n <li class=\"nav-item\">\n Sää\n \n <li class=\"nav-item ac-

tive\">\n Suosikkikaupungit\n \n \n \n </nav>\n \n </br>\n \n {{{payload}}}\n \n</body>","output":"str","x":790,"y":340,"wires":[["4ca8717a.7437b"]]},{"id":"4ca8717a.7437b","type":"http response","z":"453d419d.7131","name":"http response","statusCode":"","headers":{},"x":980,"y":340,"wires":[]},{"id":"b98c8508.2f94c8","type":"function","z":"453d419d.7131","name":"Kaupunki tietokantaan","func":"msg.payload = \"IF EXISTS (SELECT * FROM Kaupungit WHERE Nimi = '\"+ msg.payload.city.name +\"') \"+\n\t \"UPDATE Kaupungit SET Maara = Maara + 1 WHERE Nimi = '\"+ msg.payload.city.name +\"' \"+\n \"ELSE \"+\n \"INSERT INTO Kaupungit (Nimi, Maara) VALUES ('\"+ msg.payload.city.name +\"', 1) \";\n\nreturn msg;","outputs":1,"noerr":0,"x":800,"y":220,"wires":[["4a3d39c9.e39c98"]]},{"id":"4a3d39c9.e39c98","type":"MSSQL","z":"453d419d.7131","mssqlCN":"ddae2ff9.f06bf","name":"Db","query":"","outField":"payload","x":1010,"y":220,"wires":[[]]},{"id":"21112d7e.3abf52","type":"function","z":"453d419d.7131","name":"Kaupungit tietokannasta","func":"msg.payload = \"SELECT * FROM Kaupungit ORDER BY Maara DESC\";\n\nreturn msg;","outputs":1,"noerr":0,"x":310,"y":340,"wires":[["8d7edc4d.668a5"]]},{"id":"8d7edc4d.668a5","type":"MSSQL","z":"453d419d.7131","mssqlCN":"ddae2ff9.f06bf","name":"Db","query":"","outField":"payload","x":490,"y":340,"wires":[["d2630ceb.faa12"]]},{"id":"d2630ceb.faa12","type":"function","z":"453d419d.7131","name":"taulukko","func":"var taulukkoAlku = `<div class=\"card text-white bg-primary mb-3\" style=\"margin: 1em; max-width: 40rem; background-color: #9fd0f0 !important;\">\n <table class=\"table table-hover\" style=\"text-align: center; color: ghostwhite; margin: 1em; width: 90%;\">\n <thead>\n <tr style=\"font-size: 1.2em;\">\n <th>Kaupunki</th>\n <th>Määrä</th>\n </tr>\n </thead>\n <tbody>`\n \nvar taulukonRivit = '';\nfor (var i = 0; i < msg.payload.length; i++)\n{\n taulukonRivit += `<tr>\n <td>`+ msg.payload[i].Nimi +`</td>\n <td>`+

msg.payload[i].Maara +`</td>\n </tr>`\n}\n \nvar taulukkoLoppu = `</tbody>\n </table>\n </div>`\n\n\nmsg.payload = taulukkoAlku + taulukonRivit + taulukkoLoppu;\nreturn msg;","outputs":1,"noerr":0,"x":620,"y":340,"wires":[["2ac251cf.3f57fe"]]},{"id":"dbd2b44b.0d2f38","type":"template","z":"453d419d.7131","name":"400","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<!doctype html>\n<meta charset=\"utf-8\">\n<meta name=\"viewport\" content=\"width=device-width, initial-scale=1, shrink-to-fit=no\">\n\n\n<script src=\"https://code.jquery.com/jquery-3.5.1.js\" integrity=\"sha256-QWo7LDvxbWT2tbbQ97B53yJnYU3WhH/C8ycbRAkjPDc=\" crossorigin=\"anonymous\"></script>\n<link href=\"https://stackpath.bootstrapcdn.com/bootswatch/4.4.1/materia/bootstrap.min.css\" rel=\"stylesheet\" integrity=\"sha384-1tymk6x9Y5K+OF0tlmG2fDRcn67QGzBkiM3IgtJ3VrtGrIi5ryhHjKjeeS60f1FA\" crossorigin=\"anonymous\">\n<script src=\"https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js\" integrity=\"sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6\" crossorigin=\"anonymous\"></script>\n\n\n<body style=\"background-color: #9fd0f070; font-weight: lighter !important;\">\n \n <nav class=\"navbar navbar-expand-lg navbar-dark bg-primary\">\n Sääsovellus\n <button class=\"navbar-toggler\" type=\"button\" data-toggle=\"collapse\" data-target=\"#navbarColor01\" aria-controls=\"navbarColor01\" aria-expanded=\"false\" aria-label=\"Toggle navigation\">\n <span class=\"navbar-toggler-icon\">\n </button>\n <div class=\"collapse navbar-collapse\" id=\"navbarColor01\">\n <ul class=\"navbar-nav mr-auto\">\n <li class=\"nav-item active\">\n Sää\n \n <li class=\"nav-item\">\n Suosikkikaupungit\n \n \n \n </nav>\n \n </br>\n \n <div class=\"card text-white bg-primary mb-3\" style=\"margin: 1em; max-width: 60rem; background-color: #9fd0f0 !important;\">\n \n <div style=\"max-width: 25rem; margin: 1em\">\n <form method=\"POST\" action=\"/saa\">\n \n <label

for=\"city\"</label>\n \n <div style=\"background-color: #9fd0f0; padding-top: 3em; padding-bottom: 2em;\">\n <input class=\"form-control\" id=\"city\" name=\"q\" placeholder=\"Anna kaupungin nimi\">\n </br>\n <button type=\"submit\" class=\"btn btn-primary\">Hae säätiedot</button>\n \n </form>\n \n \n \n</body>","output":"str","x":750,"y":100,"wires":[["2ace31a2.280a6e"]]},{"id":"16d84b0.e2e35b5","type":"http in","z":"453d419d.7131","name":"[GET] /saa","url":"/saa","method":"get","upload":false,"swaggerDoc":"","x":80,"y":100,"wires":[["c19206c9.7a75b8"]]},{"id":"5a861baf.325c94","type":"http in","z":"453d419d.7131","name":"[POST] /saa","url":"/saa","method":"post","upload":false,"swaggerDoc":"","x":90,"y":140,"wires":[["c19206c9.7a75b8"]]},{"id":"ddae2ff9.f06bf","type":"MSSQL-CN","z":"","name":"","server":"","encyption":true,"database":"saasovellus_nr"}]