Tampereen ammattikorkeakoulu

Web-sovellus Philips Hue -järjestelmän hallintaan

Ville Helminen

OPINNÄYTETYÖ
Toukokuu 2020

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

HELMINEN, VILLE:
Web-sovellus Philips Hue -järjestelmän hallintaan

Opinnäytetyö 28 sivua
Toukokuu 2020

Tässä opinnäytetyössä tehtiin Reactilla verkkosovellus, jolla voidaan hallita Philips Hue -älyvaloja ilman erillisen sovelluksen asentamista. Tehdyllä verkkosovelluksella voidaan myös lisätä ajastimia ja muistutuksia Hue-järjestelmään sekä poistaa niitä. Työn mahdollisti Philipsin valmis ohjelmointirajapinta.

Philips Hue -järjestelmä on yksi suosituimmista älyvalojärjestelmistä. Sen käyttäminen tottuneelle käyttäjälle on helppoa ja lähes ongelmatonta. Hue-järjestelmän käyttämisessä on kuitenkin tietyt hankaluudet. Esimerkiksi talossa vierailijoilla ei ole helppoa ja nopeaa mahdollisuutta hallita Hue-valoja, koska niiden hallinta vaatii erillisen sovelluksen. Vierailijat viipyivät yleensä maksimissaan vain pari päivää, minkä takia he eivät halunneet asentaa puhelimiinsa erillistä sovellusta Hue-valojen hallitsemiseksi. Tästä seurasi ongelma, koska he eivät saaneet talossa valoja päälle. Tämän lisäksi Philipsillä ei ole tietokoneelle hyvää ja monipuolista sovellusta Hue-valojen hallintaan.

Verkkosovellus toimii paikallisesti, joten se vaatii palvelimena toimivan laitteen samasta verkosta kuin Hue-silta. Sovellusta käyttävän laitteen on myös oltava yhdistettynä tähän verkkoon. Palvelimena toimivia laitteita on monenlaisia, esimerkiksi Windows-tietokone tai Linux-tietokone, joten valmis sovellus toimii Docker-kontin sisällä. Käyttämällä Dockeria mahdollistetaan verkkosovelluksen samanlainen toiminta erilaisissa toimintaympäristöissä eli minimoidaan laitekohtaisten asetuksien vaikutus sovelluksen toimintaan.

Sovelluksen skaalautuvuuteen puhelimen ja tietokoneen näytön välillä kiinnitettiin erityistä huomiota. Lisäksi haluttiin sovelluksen käytön olevan mahdollisimman nopeaa, joten sovelluksessa käytetään sovelluskohtaista käyttäjätunnusta. Tämän ansiosta käyttäjän ei siis tarvitse erikseen rekisteröityä tai kirjautua sovellukseen. Työn tuloksena syntynyt sovellus on opinnäytetyön tekijän aktiivisessa käytössä.

Asiasanat: verkkosovellus, philips hue, react, docker

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
ICT Engineering
Software Engineering

HELMINEN, VILLE:
Web Application for Controlling the Philips Hue System

Bachelor's thesis 28 pages
May 2020

In this thesis a web application was implemented with React, which can be used to control the Philips Hue lights. With this application, it is possible to add and delete timers and reminders in the Philips Hue system. Philips provided a programming interface which made this web application possible.

Philips Hue is one of the most popular smart lighting systems, easy to use and almost trouble-free for accustomed user. However, Hue system has its disadvantages. For example, visitors' does not have easy and quick access to control the Hue lights, because it requires a separate application. Usually visitors stay only a couple of days, so they do not want to install a separate app on their phones. This causes inconveniences for the visitors, because they are cannot turn on lights in the house. In addition, Philips does not provide a good and versatile Hue lights controlling application for the computers.

The web application works locally, so it requires a server in the same network as the Hue bridge. The device using this application must also be on the same network. Many types of devices can be used as the server, like Windows computer or a Linux computer, so the finished application runs in a Docker container. By using Docker, it enables the web application to work similarly in different environments, in other words it minimizes the effects of device-specific settings.

Particular attention was paid to the scalability of the application between phones and computers. Additionally, the application was intended to be as quick and user-friendly, so application uses an application-specific username to eliminate the need for a user to log in or register to the application. The result of this thesis is a functioned application that is being actively used by the author of the thesis.

Key words: web application, philips hue, react, docker

SISÄLLYS

LYHENTEET JA TERMIT

Docker	Virtuaalinen sovelluksen toimintaympäristö, joka sisältää kaiken tarvittavan sovelluksen toimintaan.
DOM	Document Object Model. Selaimen muistissa oleva abstraktio HTML-sivun sisällön rakenteesta.
JavaScript	Ohjelmointikieli, jota käytetään web-ohjelmoinnissa.
JSX	Yleisesti React-komponenttien kirjoittamiseen käytettävä syntaksi.
Nginx	Tehokas verkko- ja välityspalvelin.
Ohjelmointirajapinta	Application programming interface (API). Ohjelmointiliitäntä, joka mahdollistaa sovelluksien tai järjestelmien keskinäisen kommunikoinnin.
Pikselitiheys	Pixel density. Mittayksikkö, joka kertoo laitteen näytön pikselien määrän tuumassa. Käytetään, jotta sovellukset skaalautuvat oikein erikokoisille näytöille.
Properties-kenttä	Kenttä, jolla Reactissa välitetään tietoa komponentilta toiselle (props).
React	JavaScript-kirjasto, jolla luodaan käyttöliittymiä.
Redux	Itsenäinen tilanhallintakirjasto JavaScript-sovelluksille.
XML	Extensible Markup Language. Laajennettava merkkauskieli.

1 JOHDANTO

Tässä opinnäytetyössä tehtiin Reactilla verkkosovellus, jolla voidaan hallita Philips Hue -älyvaloja selaimessa. Hue-valojen hallinta haluttiin mahdollisimman helpoksi ja nopeaksi, esimerkiksi talossa vieraileville henkilöille. Tehtyä verkkosovellusta on mahdollista käyttää vain samassa verkossa Hue-valojen kanssa, koska valojen etäkäytölle ei katsottu olevan tarvetta.

Työssä esitellään tärkeimmät tekniikat ja kirjastot, joita käytettiin verkkosovelluksen kehittämiseen. Lisäksi työssä selvitetään verkkosovelluksen toimintaa ja sen käyttöönottamista. Lopuksi työssä esitellään jatkokehitysideoita sovellukselle ja perustellaan muutamien ominaisuuksien pois jättämistä.

Hue-valojen hallinta haluttiin toimivaksi mahdollisimman monilla laitteilla, joten verkkosovellus oli tähän tarkoitukseen luonteva valinta. Philipsin Hue-silloissa on valmis ohjelmointirajapinta, jota käytettiin sovelluksen kehityksessä. Verkkosovelluksessa oli tärkeää, että se skaalautuu puhelimien ja tietokoneiden näytöille. Käyttäjien ei tarvitse rekisteröityä sovellukseen, jotta käyttäminen olisi mahdollisimman nopeaa. Verkkosovelluksesta ei tehty julkista. Se toimii vain laitteiden kanssa, jotka ovat yhteydessä samaan verkkoon kuin Hue-silta. Myös palvelimena toimivan laitteen pitää olla yhteydessä tähän verkkoon.

Verkkosovellusta voidaan käyttää myös muissa Hue-järjestelmissä, koska tarvittavat valojen ja ryhmien nimet ladataan suoraan ohjelmointirajapinnasta. Erilaisia palvelimia sovellukselle voi olla monenlaisia, minkä takia valmis verkkosovellus toimii Docker-kontin sisällä. Dockeria käyttämällä laitekohtaiset asetukset tai ohjelmat eivät vaikuta sovelluksen toimintaan, joten sovelluksen käyttöönottaminen on helppoa kaikilla käyttöjärjestelmillä.

Toteutuksessa on otettu huomioon Hue-valojen erilaiset ominaisuudet, jotka vaikuttavat käyttöliittymän ominaisuuksiin. Verkkosovelluksella pystytään myös luomaan uusia ajastimia ja muistutuksia Hue-järjestelmään sekä poistamaan niitä. Muistutuksissa käytetään päivämäärää ja kellonaikaa pelkän ajastimen ajan sijasta.

2 PHILIPS HUE

Philips Hue on älyvalojärjestelmä, johon kuluu erilaisia lamppuja ja tarvikkeita. Hue julkaistiin vuonna 2012, ja sitä on päivitetty 2015, 2016 ja 2019. Näissä päivityksissä julkaistiin uusia ominaisuuksia ja paranneltuja lamppuja sekä tarvikkeita. (Hue Home Lightning 2019.)

KUVA 1. Vasemmalla takana Hue-silta, oikealla takana himmennin ja edessä kaksi lamppua (Philips Hue 2020)

Toimiakseen Philips Hue tarvitsee Hue-sillan (kuva 1), johon lamput yhdistyvät. Poikkeuksena ovat uusimman julkistuksen yhteydessä julkistetut lamput, joissa on bluetooth-ominaisuus. Uusimpia lamppuja voidaan käyttää ilman Hue-siltaa, mutta ominaisuudet ovat rajoitettuja. Hue-silta pitää yhdistää Ethernet-kaapelilla reitittimeen, siinä ei siis ole langatonta verkkoyhteyttä, mikä pitää ottaa huomioon lamppuja hankkiessaan. Lamppuja voidaan hallita ja järjestellä eri huoneisiin tai ryhmiin, esimerkiksi Philips Huen virallisella sovelluksella. Lamppujen järjestäminen huoneisiin tai ryhmiin vaati Hue-sillan. Philipsin sovellus on saatavilla Androidille ja iOS-laitteille. Huella on tuki myös ääniohjaukselle, esimerkiksi Google Assistantin, Amazon Alexan tai Apple Sirin avulla. (Christ 2019.)

Hue-järjestelmän käyttöönottaminen on helppoa, varsinkin jos ostaa aloituspakkauksen. Pakkaus sisältää erilaisia lamppuja ja Hue-sillan eli kaikki, mitä Hue-

järjestelmän toimintaan tarvitaan. Kuvassa 1 on Philips Hue White and Color Ambiance E27 -aloituspakkaus, jossa Hue-sillan ja himmentimen lisäksi on kaksi värillistä lamppua. Aloituspakkauksessa voi näiden kahden lampun tilalla olla pelkästään valkoisia lamppuja tai G10-kantaa käyttäviä lamppuja. Varsinainen käyttöönottaminen voidaan jakaa kolmeen vaiheeseen:

1. Asennetaan Hue-valo valaisimeen
2. Yhdistetään Hue-silta Ethernet-kaapelilla reitittimeen
3. Avataan Hue-sovellus laitteessa ja lisätään valot haluttuun huoneeseen tai ryhmään

Hue-sovelluksen pitäisi löytää lähellä olevat lamput automaattisesti. Automaattinen lisäys ei välttämättä aina toimi, joten lamput voidaan lisätä Hue-järjestelmään myös manuaalisesti sarjanumeron avulla. Näiden kolmen vaiheen jälkeen Hue-valot ovat valmiina käyttöön. (Signify Holding 2019.)

Hue-laitteet käyttävät kommunikointiin zigbee-protokollaa. Se on langaton viestintäteknologia, jolla Hue-laitteet yhdistyvät toisiinsa ja Hue-siltaan. Hue-järjestelmä ei tarvitse uusia johtoja tai asennuksia Hue-sillan ja lampun vaihtamisen lisäksi. (Christ 2019.)

Tätä opinnäytetyötä tehtäessä Hue-järjestelmään kuului kaksi Hue Play -valaisinta, kolme kolmannen sukupolven White and Color Ambiance -lamppua, yksi White Ambiance -lamppu ja kaksi älypistorasiaa. Älypistorasiat Hue-järjestelmä tulkitsee lampuiksi ja ne voidaan kytkeä päälle ja pois sekä ajastaa Hue-sovelluksen avulla, kuten muutkin lamput. Käyttöliittymässä on otettu huomioon näiden lamppujen erilaiset ominaisuudet.

3 SOVELLUKSEN TEKNOLOGIAT

3.1 React

React on Facebookin vapaan lähdekoodin JavaScript-kirjasto, jolla voidaan kehittää monipuolisia ja suuria verkkosovelluksia mahdollisimman vähällä määrällä koodia. Reactilla pystytään jakamaan verkkosovelluksen rakenne pienempiin osiin eli komponentteihin. Tämän suurin hyöty tulee esille näiden komponenttien uudelleenkäytettävyydessä, esimerkiksi jos samanlaista näkymää käytetään useassa eri paikassa, siitä voidaan tehdä erillinen komponentti. Tätä komponenttia voidaan käyttää uudelleen ilman koko komponentin koodin uudelleenkirjoittamista. (Thinkwik 2017.)

React-komponenttien kirjoittamiseen käytetään yleisesti XML-tyylistä syntaksia, joka on nimeltään JSX. Käyttämällä JSX-syntaksia ei tarvitse luoda elementtejä kutsumalla React.createElement-metodia (kuva 2), vaan React hoitaa tämän käyttäjän puolesta. Selaimet eivät ymmärrä JSX-syntaksia suoraan, joten se käännetään JavaScript-muotoon yleensä Babel-kääntäjällä. (Introducing JSX 2020.)

```
const myelement = <h1>Hello world</h1>;

ReactDOM.render(myelement, document.getElementById('root'));
```

Kuva 2. Esimerkki Hello world -tulostuksesta JSX-syntaksilla

```
const myelement = React.createElement('h1', {}, 'Hello world');

ReactDOM.render(myelement, document.getElementById('root'));
```

KUVA 3. Esimerkki Hello world -tulostuksesta ilman JSX-syntaksia

Kuvissa 2 ja 3 nähdään yksinkertainen Reactilla tehty verkkosovelluksen osa, joka näyttää sivulla Hello World -tekstin. Näistä toinen käyttää JSX-syntaksia (kuva 2) ja toinen ei (kuva 3). Kuvista nähdään, että JSX-syntaksilla sovellus on

nopeampaa tehdä ja luodut komponentit ovat helpommin käytettävissä. Ero kuitenkin korostuu suuremmissa ja monimutkaisemmissa sovelluksissa huomattavasti enemmän.

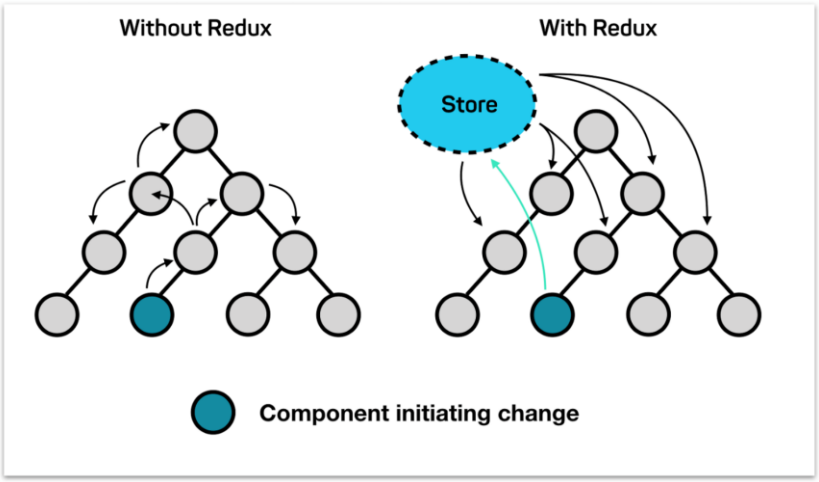
Reactin tehokkuus perustuu sen käyttämään virtuaaliseen DOM-kerrokseen, joka ei kuitenkaan ole Facebookin itse kehittämä. Tavallisesti JavaScript-sovellus aiheuttaa koko DOM-kerroksen uudelleen luomisen, kun käyttöliittymässä tarvitsee tehdä muutoksia. Tämä voi olla hyvinkin hankalaa ja tehotonta, varsinkin suuremmissa selainsovelluksissa, joissa DOM-rakenne on hyvin monimutkainen. Reactin virtuaalista DOM-kerrosta voidaan pitää yksinkertaistettuna DOM-kerroksen kopiona. Tämän ansiosta React pystyy luomaan nettisivulla tapahtuvat muutokset ensin tähän virtuaaliseen DOM-kerrokseen. Kun tarvittavat muutokset ovat valmiina virtuaalisessa DOM-kerroksessa, selainsovelluksesta päivitetään vain tarvittavat komponentit. Tämä tapahtuu vertaamalla virtuaalisen DOM-kerroksen ja varsinaisen DOM-kerroksen eroavaisuuksia. Näillä kahdella DOM-kerroksella ei välttämättä ole suuriakaan eroja, joten React saattaa selaimessa näyttää puhtaalta html-koodilta. (Krajka 2015.)

3.2 Redux

Redux on itsenäinen, JavaScript-sovelluksille kehitetty, tilanhallintakirjasto. Sitä käytetään erityisesti Reactin kanssa. Sen suosio perustuu kirjaston pieneen kokoon ja koodirakenteen selkeyttämiseen. Reactin kanssa käytetään yleensä react-redux-kirjastoa, joka on virallisesti Reduxin tukema ja ylläpitämä kirjasto, koska Redux on alun perin kehitetty Reactille. (Redux 2020.)

Redux-kirjastoa kannattaa käyttää, kun käyttöliittymä on jaettu moniin komponentteihin ja ne käyttävät keskenään samaa dataa tai tietyn komponentin tilaa tarvitaan muissa komponenteissa. Reactin komponenteilla on jokaisella oma tila, jonka perusteella käyttöliittymässä näytetään tietoa. Tätä komponentin tilaa voidaan käyttää toisesta komponentista, kun se välitetään properties-kentässä (props). Tämä voi olla vaikeaa, jos esimerkiksi komponentin, jolle tila välitettiin, täytyy välittää saatua tilaa vielä eteenpäin (kuva 4). Tähän ongelmaan Redux on hyvä ratkaisu. (Ighodaro 2018.)

Redux yhdistää komponenttien tilat yhteen paikkaan, joka on tilasäiliö (store). Tämän ansiosta komponenttien tilaa ei tarvitse välittää jokaiselle komponentille erikseen properties-kentässä, vaan komponentit voivat hakea tarvitsemansa tilan suoraan tilasäiliöstä (kuva 4).



Kuva 4. Tiedon välitys Reactin komponenttien välillä (Korzunov 2018)

Kuvasta 4 nähdään Reduxin hyötyä. Vasemmalla komponentin tilaa joudutaan antamaan alemmille komponenteille properties-kentässä, koska ei käytetä Reduxia. Dataa muokattaessa uusi data joudutaan puolestaan antamaan komponenteille hierarkiassa ylöspäin. Kuvassa 4 oikealla käytetään Reduxia, joten jokainen komponentti voi hakea tarvitsemansa tilan suoraan tilasäiliöstä (store). Tilasäiliön tietoja voidaan muokata mistä tahansa komponentista reducer-funktioilla, jotka ovat puhtaita funktioita (Ighodaro 2018).

3.3 Tuotantotyökalut

Työssä käytettiin kahta työkalua, Docker ja Nginx, jotka eivät vaikuttaneet tai auttaneet varsinaisessa sovelluksen kehityksessä. Näillä työkaluilla varmistetaan valmiin sovelluksen mahdollisimman helppo ja nopea käyttöönottaminen sekä käyttäminen. Näistä kahdesta Docker on myös kehityksessä käytettävä työkalu, mutta tässä opinnäytetyössä sitä käytetään ainoastaan valmiin sovelluksen kanssa.

3.3.1 Docker

Docker on kehitetty helpottamaan ja nopeuttamaan sovelluskehitystä hyödyntämällä kontteja (containers). Kontit mahdollistavat sovelluksen koko rakenteen pakkaamisen yhteen paikkaan, joka sisältää kaiken tarvittavan sovelluksen käyttämiseen. Käytetyt kirjastot ja riippuvuudet sisältyvät myös konttiin. Tätä hyödyntämällä kehittäjä voi luottaa, että kehitetty sovellus toimii myös muiden kehittäjien ja käyttäjien laitteilla. Erilaiset laitteiden asetukset ja säädöt eivät vaikuta sovelluksen toimintaan, kun käytetään Dockeria. (Opensource.com n.d.)

Docker on tavallaan virtuaalikone, mutta se ei tarvitse kokonaista ja erillistä käyttöliittymää toimiakseen. Se on avoimen lähdekoodin tuote, jota nykyisin ylläpitää Docker.Inc-niminen yritys. Docker on saatavissa Linux-, Mac- ja Windows-käyttöjärjestelmille. (Wallenius n.d.)

3.3.2 Nginx

Nginx on hyvin tunnettu avoimen lähdekoodin projekti, jonka alkuperäinen kehittäjä on venäläinen Igor Sysoev. Hän aloitti projektin vuonna 2002 ja teki siitä julkisen vuonna 2004. Tästä lähtien Nginxistä on tullut vakituinen osa tehokkaita ja skaalautuvia nettisivuja ja palvelimia. Kymmenet miljoonat nettisivut käyttävät Nginxiä, kuten Netflix, Airbnb ja Dropbox. Nginx on tehokas ja skaalautuva nettipalvelin, välityspalvelin ja kuormantasaaja. Se optimoi modernien palvelinten käyttöjärjestelmien, kuten Linux, muistin käytön, prosessorin tehon ja internetyhteyden. Tämän ansiosta Nginx pystyy käsittelemään yleensä kymmenkertaisesti enemmän pyyntöjä palvelimelle kuin kilpaileva tuote Apache. (Nginx n.d.)

4 SOVELLUKSEN SUUNNITTELU JA TOTEUTUS

Opinnäytetyössä tehtiin verkkosovellus Philips Hue -älyvalojen hallintaan Reactilla. Philipsin Hue-sillassa on valmis ohjelmointirajapinta, jolla valoja voi hallita itsetehdyllä sovelluksella. Sovelluksen kehityksessä otettiin huomioon helppo ja nopea tapa Hue-valojen hallintaan mahdollisimman monella erilaisella laitteella. Kehitettiin verkkosovellus, jolla talossa vierailevat henkilöt pystyvät käyttämään valoja ilman erillisen sovelluksen asentamista. Lisäksi opinnäytetyön tekijä pystyy käyttämän valoja tietokoneelta. Sovellusta käytetään luultavasti puhelimella sekä tietokoneella, joten verkkosovelluksen pitää skaalautua erikokoisten laitteiden näytöille.

4.1 Rakenne

Sovelluksen käyttöönottaminen haluttiin pitää mahdollisimman yksinkertaisena, joten sillä ei ole tietokantaa. Tarvittavat tiedot ladataan suoraan Philipsin ohjelmointirajapinnasta. Verkkosovellus koostuu neljästä eri sivusta: ryhmät, valot, ajastukset ja asetukset. Sovelluksella ei ole muita osia kuin Reactilla tehty käyttöliittymä, jonka takia sovelluksen kehittämisvaiheessa ei ollut tarpeellista käyttää Dockeria.

Käyttöliittymän osat jaettiin pienempiin komponentteihin yhden ison komponentin sijaan. Pienempien komponenttien ansiosta on helpompi löytää kohta, johon halutaan tehdä muutoksia ja muutenkin selkeyttää koodin rakennetta. Pääkomponenteiksi muodostui sovelluksen sivut: valot, ryhmät, ajastukset ja asetukset. Näillä komponenteilla on useita lapsikomponentteja, joita voidaan hyödyntää koodin eri kohdissa. Näille lapsikomponenteille annetaan tarvittavat tiedot komponentin properties-kentässä tai suoraan Reduxin tilasäiliöstä tilanteen mukaan. Jokainen pääkomponentti toimii omassa url-osoitteessansa. Tähän käytettiin Reactin kanssa yleisesti käytössä olevaa ja suosittua react-router-dom-kirjastoa.

4.2 Näkymät

Näkymiä sovelluksella on neljä: valot, ryhmät, ajastukset ja asetukset eli aiemmin mainitut pääkomponentit. Valot- ja ryhmät-näkymillä on kaksi erilaista näkymää, riippuen siitä, onko vierastila päällä vai pois päältä. Sovelluksessa on kaikki ominaisuudet valojen hallintaan käytettävissä, jos vierastila on pois päältä. Kun vierastila on päällä, sovelluksen ominaisuuksia rajoitetaan. Lisäksi ryhmät-sivulla voidaan valita, mitkä ryhmät näytetään. Hue-järjestelmä on jakanut ryhmät kolmeen osaan: alueet (zones), viihde (entertainment) ja huoneet.

Ajastukset-sivulla hyödynnetään Huen schedules-toimintoa. Sivulla nähdään valmiiksi tehtyjä ajastuksia. Nämä valmiiksi tehdyt ajastukset ovat yleensä ajastimia (timers). Toinen vaihtoehto ajastukselle on päivämäärä ja kellonaika eli muistutus. Sivulla voidaan myös luoda uusia ajastuksia ja poistaa niitä.

Sovelluksessa on sivupaneeli, jota käytetään navigointiin eri näkymien välillä. Sivupaneeli avautuu painikkeesta yläpalkissa. Sivupaneeliin on laitettu mahdollisuus vaihtaa vierastilan tilaa ja linkit jokaiseen näkymään. Lisäksi siinä on näytä kaikki -kytkin. Tämä kytkin vaikuttaa ainoastaan valot-sivun näkymään. Kun kytkin on pois päältä, valot-sivulla näytetään ainoastaan valot, jotka Hue-silta ilmoittaa olevan saavutettavissa. Kaikki valot näytetään, jos kytkin on päällä. Silloin näytetään myös valot, jotka Hue-silta ilmoittaa olevan saavuttamattomissa. Valot voivat olla saavuttamattomissa, jos ne ovat rikki tai niiden yhteys Hue-siltaan on tilapäisesti katkennut. Ainoastaan saavutettavissa olevia valoja voidaan hallita verkkosovelluksella.

4.3 Ominaisuudet

Valojen helppoa hallintaa ajatellen sovelluksessa on vierastila. Vierastilassa voidaan laittaa valoja päälle ja pois sekä säätää niiden kirkkautta. Tässä tilassa valojen värin muuttaminen on pois käytöstä ja piilotettuna käyttöliittymästä. Värin muuttaminen valoille on mahdollista, kun vierastila on pois päältä ja kyseinen valo tukee tätä ominaisuutta. Värin muuttamisessa on kaksi vaihtoehtoa: valkoisen värin eri sävyt ja valinta kaikista väreistä. Hue-valoilla on erilaisia ominaisuuksia,

jotka ovat otettu huomioon käyttöliittymässä. Esimerkiksi värin muuttaminen on poistettu käytöstä, jos kyseinen valo ei tätä ominaisuutta mahdollista.

Hue-järjestelmästä löytyvät ajastukset näytetään ajastukset-sivulla, josta näitä voidaan laittaa päälle ja pois sekä poistaa. Uusia ajastuksia voidaan myös luoda Hue-järjestelmään. Tämä tapahtuu antamalla ajastuksen nimi ja kuvaus sekä ajastimen (timer) aika tai muistutuksen päivämäärä ja kellonaika. Näiden tietojen lisäksi kysytään, mitkä valot suorittavat halutun toimenpiteen. Toimenpiteet ja valo/ryhmä valitaan pudotusvalikoista. Toimenpiteitä on valittavissa kolme: sammuta valitut valot, laita valitut valot päälle tai väläytä valittuja valoja 15 kertaa. Näistä tiedoista on pakollista syöttää kaikki muut, paitsi ajastuksen kuvaus.

Asetuksia sovellukseen tehtiin käytön helpottamiseksi. Asetukset suunniteltiin opinnäytetyön tekijän mieltymysten ja kokemusten perusteella. Sovellukseen haluttiin mahdollisuus asettaa oletuskirkkaus lampuille ja ryhmille, kun ne laitetaan päälle. Lisäksi haluttiin mahdollisuus käynnistää valot aina valkoisiksi. Asetuksista voidaan myös valita sovelluksen aloitussivuksi valot- tai ryhmät-näkymä.

Verkkosovelluksella voidaan luoda käyttäjätunnus Hue-sillalle, joka toimii ohjelmointirajapinnan avaimena. Ilman tätä avainta ei voida kutsua ohjelmointirajapinnan päätepisteitä. Tämä avain tarvitsee luoda vain ensimmäisellä käyttöönottokerralla tai vaihtoehtoisesti käyttää jo olemassa olevaa käyttäjätunnusta. Verkkosovelluksen kaikki käyttäjät käyttävät samaa käyttäjätunnusta, joten avain ei tässä tapauksessa ole henkilökohtainen vaan sovelluskohtainen. Sovellusta käyttävien henkilöiden ei siis tarvitse erikseen luoda tunnuksia hallitakseen Hue-valoja. Tämä oli alkuperäinen tarkoitus verkkosovellusta suunniteltaessa.

Verkkosovellus tunnistaa Hue-sillan IP-osoitteen automaattisesti, mutta myös manuaalinen IP-osoitteen syöttäminen on mahdollista. Kuvasta 5 nähdään, miten automaattinen tunnistus on ohjelmoitu. UseEffect-funktiossa haetaan Hue-sillan IP-osoite fetch-funktiolla, jossa kutsutaan Philipsin palvelua. Vastauksena tältä palvelulta saadaan taulukko, jossa on lueteltuna kaikki löytyneet Hue-sillat verkosta, johon laite on yhdistetty. Tässä verkkosovelluksessa oletetaan, että Hue-siltoja on vain yksi, joten koodissa käytetään kiinteästi vastauksen ensimmäistä

taulukon arvoa. IP-osoite tallennetaan sivuston evästeisiin, kun se löydetään automaattisesti tai syötetään manuaalisesti. Ennen Philipsin palvelun kutsumista tarkistetaan, onko IP-osoite jo tallennettu evästeisiin, jotta ei tehdä turhia kutsuja Philipsin palveluun. Virhetilanteessa käyttäjää pyydetään tarkistamaan, että Hue-silta on yhdistettynä reitittimeen sekä internet-yhteyden tila. Tämä virheilmoitus näkyy uudessa modaali-ikkunassa, jossa on myös mahdollista antaa Hue-sillan IP-osoite manuaalisesti, jos automaattinen tunnistus ei jostain syystä toimi.

```
React.useEffect(() => {
  if (!Cookies.get("url")) {
    fetch("https://discovery.meethue.com/")
      .then(res => res.json())
      .then(res => {
        const ip = res[0] ? res[0].internalipaddress : undefined

        if (!ip) {
          const msg = "Please check your internet connection and make sure that Hue bridge is connected to router!"
          handleOpen(msg)
        }
        else {
          Cookies.set("ip", 'http://' + ip)
          Cookies.set("url", 'http://' + ip + '/api/' + Config.username)
          const msg = "Hue bridge found! IP is: " + ip
          setStates(state => ({ ...state, msg: msg }))
        }
      })
      .catch(e => {
        console.log(e)
        const msg = "Please check your internet connection and make sure that Hue bridge is connected to router!"
        handleOpen(msg)
      })
  }
}, [dispatch])
```

Kuva 5. Hue-sillan IP-osoitteen tunnistaminen automaattisesti

4.4 Käyttöliittymä

Philips Hue -valojen hallintaan tarkoitettu verkkosovellus tehtiin Reactilla sekä Material-UI-kirjastolla. Material-UI on suosittu ulkonäkökirjasto verkkosivuille. Tässä verkkosovelluksessa käytetään Material-UI:n valmiita komponentteja. Verkkosivun skaalautuminen erikokoisille näytöille on tehty Material-UI:n Grid-komponentilla (kuva 6). Muita käytettyjä komponentteja on mm. Textfield, Select ja Dialog. Verkkosovellus on luotu create-react-app-komennolla, joka luo valmiin toimivan pohjan, jota voidaan muokata. Tämä komento hoitaa myös tarpeelliset webpack- ja babel-konfiguraatiot automaattisesti.

Kuvasta 6 nähdään esimerkki, miten verkkosovelluksen toimivuus erikokosilla näytöillä on tehty. Material-UI:n Grid-komponentilla voidaan tehdä erilainen näkymä erikokoisille näytöille. Grid-komponentit laitetaan yhden Grid-komponentin sisään, joka määritetään säiliöksi (container). Tämän sisällä olevat Grid-komponentit määritetään osioiksi (item). Näille osioiksi määritetyille Grid-komponenteille voidaan antaa tieto properties-kentässä, miten niiden sisällä olevat komponentit asetellaan näytölle sen koon mukaan. Maksimiarvo kokokentälle on 12, joka tarkoittaa Grid-komponentin leveyttä. Näitä kokokentän arvoja Grid-komponentille on olemassa viisi, jotka ovat pienimmästä suurimpaan: xs, sm, md, lg ja xl. Kokokenttä xs tarkoittaa näytön kokoa välillä 0 ja 600 pikseliä. Yleisesti puhelimen näyttö kuuluu tälle välille, kun otetaan huomioon näytön pikselitiheys. Md-kenttä on näytön koko välillä 960 ja 1280 pikseliä. Kuvassa 6 xs-kentän arvoiksi on asetettu 12, eli yksi komponentti käyttää koko Grid-komponentin leveyden, joten nämä kaksi komponenttia näkyvät puhelimella allekkain. Md-kentän arvot ovat 8 ja 4, joten nämä kaksi komponenttia käyttävät yhdessä Grid-komponentin leveyden, eli ne näkyvät vierekkäin, kun käytetään yli 960 pikseliä leveää näyttöä. Tässä md-kentän arvoja käytetään aina yli 960 pikseliä leveillä näytöillä, koska Grid-komponenteille ei ole määritetty md-kenttää suurempaa kokokenttää. Tällä tavalla voidaan hyödyntää näytön leveyttä ja saadaan sovellus näyttämään hyvältä erikokoisilla näytöillä.

```
<Grid
    container
    justify="flex-start"
    alignItems="center"
    className={classes.container}
    spacing={2}

>
    <Grid item md={8} xs={12} className={classes.centerText}>
        <Typography variant="caption" className={classes.title2}>
            Default brightness for lights
        </Typography>
    </Grid>

    <Grid item md={4} xs={12} className={classes.centerActions} >
        <CustomSelector
            open={open.lights}
            handleOpen={handleLightsOpen}
            bri={lightsDefaultBri}
            handleChange={handleChangeLights}
            handleClose={handleClose}
        />
    </Grid>
</Grid>
```

Kuva 6. Esimerkki Grid-komponentin käytöstä verkkosovelluksessa

Verkkosovelluksella ei ole erillistä tietokantaa, joten kaikki sovelluksen tarpeelliset tiedot tallennetaan Reduxin tilasäiliöön, jonka kanssa käytetään redux-persist-kirjastoa. Tällä kirjastolla voidaan valita, mitä tietoja Reduxin tilasäiliöstä tallennetaan selaimen paikalliseen muistiin (local storage). Tässä sovelluksessa tallennetaan selaimen muistiin kirjautumisen tila ja käyttäjän muuttamat asetukset. Kirjautumisella tarkoitetaan pelkän salasanan syöttämistä. Salasana tarvitsee antaa ensimmäisellä kerralla, kun sovellusta käytetään, jos salasanakysely on päällä. Nämä tiedot pysyvät muistissa, kunnes laitteelta tyhjennetään selaimen muisti. Tiedot siis pysyvät muistissa, vaikka selain suljetaan. Sovelluksen asetukset eivät tallennu, jos salasanan kysely on pois päältä.

4.5 Kehitys- ja tuotantoympäristöt

Sovelluksen haluttiin toimivan monilla erilaisilla laitteilla ja ilman asennusta, joten verkkosovellus oli looginen valinta toteutukseen. Hue-sillan valmis ohjelmointirajapinta mahdollisti tämän kehityksen. Sovelluksesta olisi ollut mahdollista tehdä julkinen, eli se olisi toiminut julkisen internetosoitteen kanssa, jolloin Hue-valojen etähallinta talon ulkopuolelta olisi ollut mahdollista. Suunnitteluvaiheessa tehtiin

kuitenkin valinta, että sovelluksesta haluttiin ainoastaan paikallinen, koska valojen etäkäyttöä ei haluttu mahdollistaa. Tämä tarkoittaa, että palvelimen sekä sovellusta käyttävän laitteen pitää olla samassa verkossa kuin Hue-silta. Palvelin voi olla, esimerkiksi tietokone, josta verkkoselain lataa tarvittavat tiedot verkkosovelluksen näyttämiseksi.

Kehityksessä ei ole käytetty kiinteitä huoneiden, ryhmien tai valojen nimiä, vaan kaikki nämä tiedot ladataan Hue-sillan ohjelmointirajapinnasta. Tämän ansiosta sovellusta pystytään käyttämään myös muissa Hue-järjestelmissä. Sovelluksen täytyy tietää Hue-sillan IP-osoite ja sen generoima käyttäjätunnus, jotta sitä voidaan käyttää. IP-osoitteen verkkosovellus hakee automaattisesti, jos käyttäjällä on toimiva internet-yhteys ja Hue-silta on päällä sekä yhdistettynä reitittimeen. IP-osoite on mahdollista antaa myös manuaalisesti, jos automaattinen haku ei sitä löydä. Käyttäjätunnus tarvitsee antaa sovellukselle manuaalisesti hueConfig.json-tiedostoon (kuva 7). Verkkosovellus pyytää luomaan käyttäjätunnuksen ensimmäisellä käyttöönottokerralla, koska Hue-valojen hallinta ei ole mahdollista ilman tätä tunnusta. Lisäksi tällä tiedostolla voidaan päättää, halutaanko käyttää salasanasuojausta. Salasana kysytään vain ensimmäisellä kerralla sivulle mentäessä. Käyttäjän muuttamia asetuksia ei tallenneta, jos salasanasuojaus on pois päältä, jolloin asetukset nollautuvat selaimen sulkemisen jälkeen. Salasanan voi vapaasti keksiä itse, eli se ei liity Hue-järjestelmän ominaisuuksiin.

```
{
    "username": "xxxxxxxxxxxxxxxxxxx-yyyyyyyyyyyyyyyyyy",
    "passWord": "canBeAnything",
    "loginEnabled": true
}
```

Kuva 7. Sovelluksen konfigurointitiedosto

Valmista verkkosovellusta voidaan käyttää muissakin Hue-järjestelmissä, joten käyttöympäristöjä on monenlaisia. Tämä voi aiheuttaa ongelmia, jos palvelimina käytetään tietokoneita, jotka käyttävät erilaisia käyttöjärjestelmiä. Docker valittiin ratkaisemaan tämä mahdollinen ongelma. Dockerin kanssa käytetään Nginxiä, joka helpottaa verkkosovelluksen käyttöä. Nginx hoitaa verkkosovelluksen reiti-

tyksen oikealla tavalla ja lyhentää sovelluksen paikallista url-osoitetta. Paikalliselle nettisivulle pääsee palvelimena toimivan laitteen paikallisella IP-osoitteella, jos laitteet ovat samassa verkossa. Palvelimena toimivalta laitteelta tarvitaan lisäksi tieto, missä portissa sovellus on toiminnassa. Ilman Nginxiä käyttäjä menee, esimerkiksi sivulle 192.168.x.y:3000, jossa x ja y ovat korvattu palvelimen IP-osoitteen tiedoilla ja 3000 on portti, jossa sovellus on toiminnassa. Käyttäjän ei tarvitse syöttää porttia lainkaan, kun käytetään Nginxin välityspalvelinta. Nettisivulle pääsemiseen riittää siis pelkkä palvelimen paikallinen IP-osoite. Tämä IP-osoite saadaan, esimerkiksi reitittimen hallintasivulta tai suoraan palvelimelta. Lisäksi Ilman Nginxiä suoran url-osoitteen syöttäminen ei toimi, koska sivujen reititykseen on käytetty react-router-dom-kirjastoa.

5 TULOKSET JA SOVELLUKSEN KÄYTTÖ

Kappaleessa esitellään työssä saadut tulokset ja sovelluksen käyttöliittymää. Lisäksi kerrotaan verkkosovelluksen käyttöönottamisesta ja sen käyttämisestä.

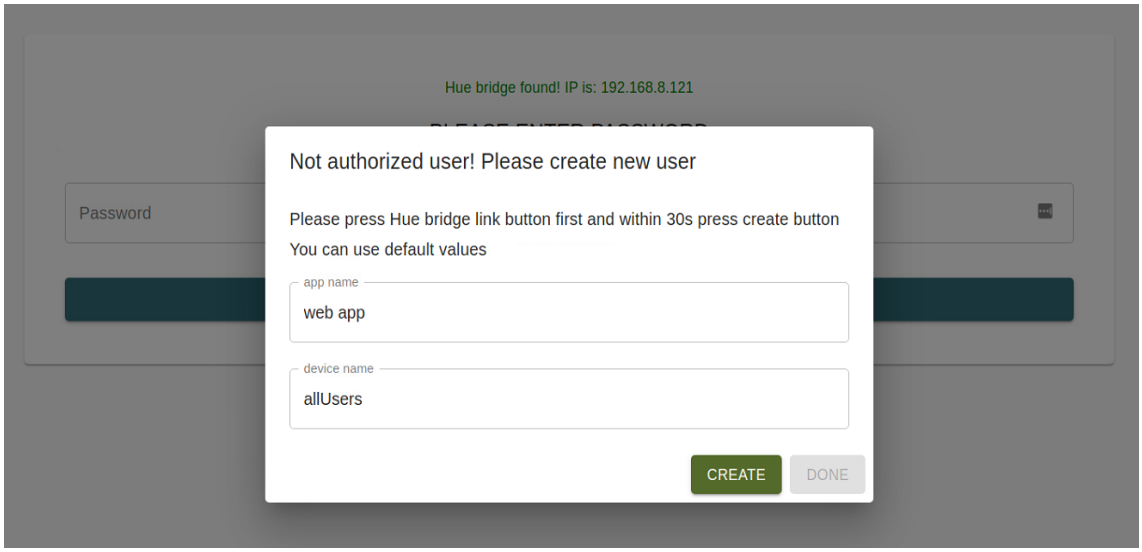
Tuloksena opinnäytetyössä saavutettiin toimiva verkkosovellus Hue-valojen hallintaan. Kaikki suunnitellut ominaisuudet saatiin tehtyä käyttöliittymään halutulla tavalla. Verkkosovelluksen käyttöönottaminen vaatii aiemmin mainitut konfiguraatiot hueConfig.json-tiedostoon. Käyttäjätunnus tarvitsee luoda ja kirjoittaa edellä mainittuun tiedostoon, kun verkkosovellus otetaan käyttöön ensimmäistä kertaa. Käyttäjätunnus voidaan luoda verkkosovellusta käyttämällä, mutta sen luonnin jälkeen Docker-kontti joudutaan rakentamaan (build) uudelleen. Docker-kontin uudelleenrakentamisen jälkeen muutokset tulevat voimaan tuotantotilassa olevaan verkkosovellukseen ja Nginxiin. Sovelluksen käyttöönottaminen tapahtuu siis seuraavasti:

1. Projektikansiossa ajetaan komento docker-compose up -d --build. Tämän jälkeen verkkosovellus on käynnissä laitteen IP-osoitteessa. Selaimessa näkyy kirjautumissivu, johon aukeaa modaali-ikkuna (kuva 8) tarvittaessa. Ikkunassa pyydetään luomaan uusi käyttäjätunnus, jos verkkosovellus otetaan käyttöön ensimmäistä kertaa. Lisäksi aukeaa saman tyylinen modaali-ikkuna, jos Hue-sillan IP-osoitetta ei löydy automaattisesti. Tässä ikkunassa Hue-sillan IP-osoite on mahdollista syöttää manuaalisesti.

2. Hue-sillalle luodaan uusi käyttäjätunnus modaali-ikkunassa (kuva 8). Ikkunassa on mahdollisuus muokata sovelluksen ja laitteen nimeä, mutta se ei ole pakollista. Käyttäjätunnus luodaan painamalla Create-painiketta 30:n sekunnin sisällä Hue-sillan linkkipainikkeen painamisesta. Huen generoima käyttäjätunnus näytetään tekstikenttien alapuolella. Tämä käyttäjätunnus pitää kopioida hueConfig.json-tiedostoon. Tekstikenttien alapuolella näytetään myös mahdollinen virheilmoitus,

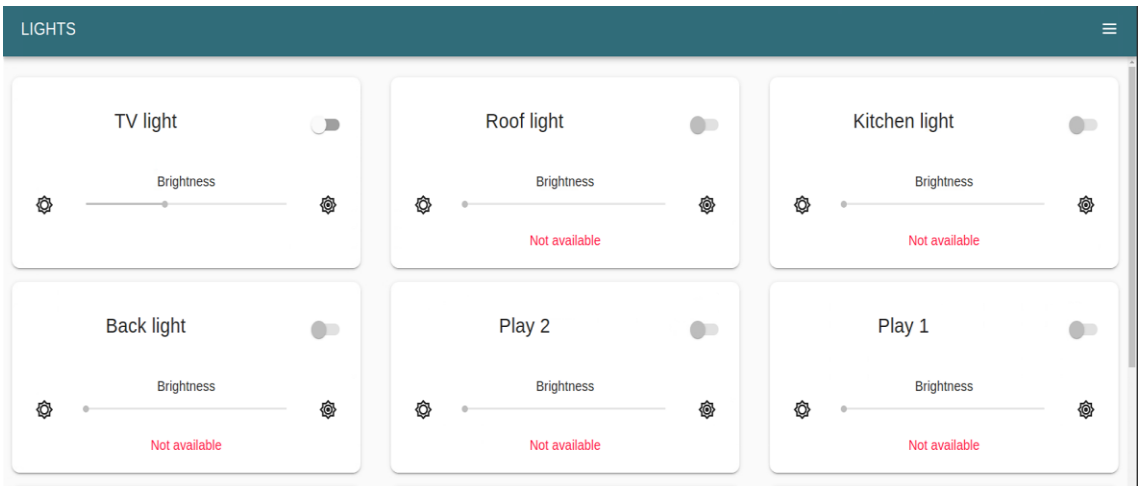
esimerkiksi jos Hue-sillan linkkipainiketta ei ole painettu ennen Create-painikkeen painamista.

3. Viimeiseksi pitää käynnistää ja rakentaa Docker-kontti uudelleen, eli ajaa komennot docker-compose down ja docker-compose up -d --build. Sivulle pääsee nyt syöttämään salasanan, joka on määritetty hueConfig-json-tiedostossa.
4. Kun tarvittavat konfiguraatiot on jo tehty sekä otettu käyttöön, riittää sovelluksen käynnistämiseen komento docker-compose up -d. Docker-konttia ei tarvitse uudelleenrakentaa, jos tehtyjä konfiguraatioita ei muuteta. Sovellus käynnistyy huomattavasti nopeammin ilman rakentamista (build).



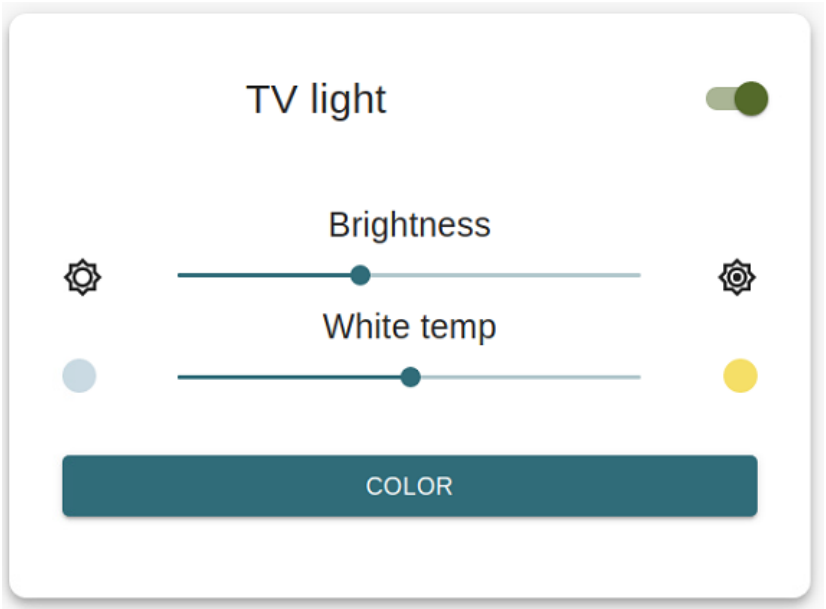
Kuva 8. Näkymä sovelluksen ensimmäisellä käyttöönottokerralla

Oletusasetuksilla käyttäjä siirretään valot-sivulle, kun oikea salasana on syötetty. Sovelluksessa on vierastila päällä, eli valot-sivu on yksinkertaistetussa näkymässä (kuva 9). Sivulla listataan oletuksena kaikki hallittavissa olevat valot yksitellen korttinäkymässä. Sivupaneelista voidaan laittaa päälle näytä kaikki -ominaisuus, jolla listataan myös tavoittamattomissa olevat valot. Näiden valojen kaikki toiminnot ovat poistettu käytöstä. Kuvassa 9 tämä ominaisuus on päällä. Verkkosovelluksen ryhmät-sivu on melkein identtinen valot-sivun kanssa. Ainoina eroina on, että ryhmät-sivulla voidaan valita, mitkä ryhmät näytetään ja tarkistaa, mitkä lamput kuuluvat tiettyyn ryhmään. Ryhmät-sivulla yksi kortti hallitsee kaikkia Hue-valoja, jotka kyseiseen ryhmään kuuluu.



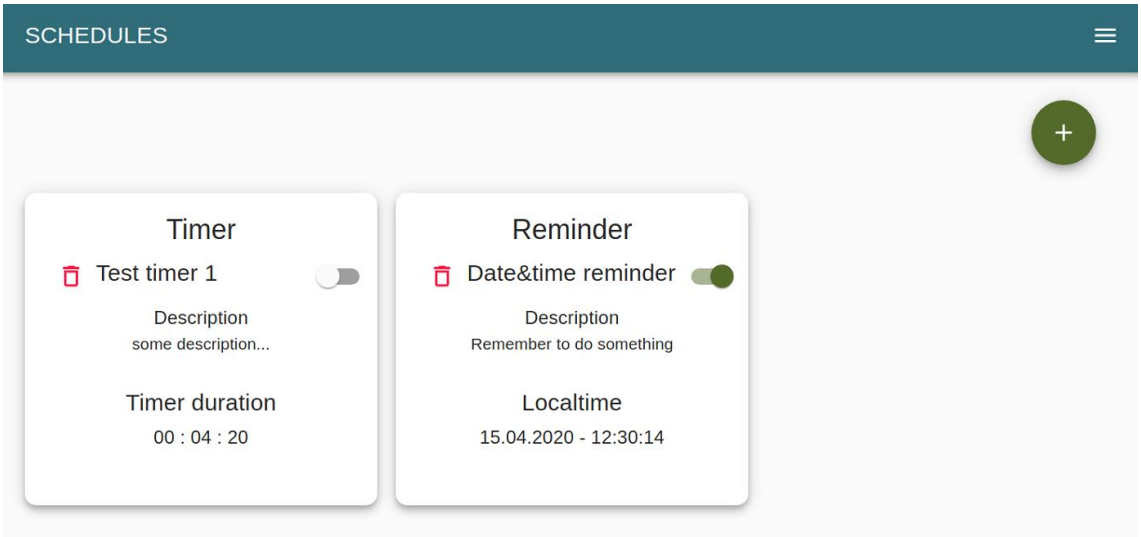
Kuva 9. Valot-sivun yksinkertaistettu näkymä tietokoneella, kun näytä kaikki -kytkin on päällä

Vierastila-kytkimellä saadaan vierastila pois päältä ja päälle suoraan sivupaneelista. Edistyneessä näkymässä, kun vierastila on pois päältä, korteissa on lisää valojen hallintaominaisuuksia (kuva 10). Kuvasta 10 nähdään esimerkki yhdestä kortista, kun vierastila on pois päältä. Valojen säädöt ovat pois käytöstä, jos lamppu ei ole päällä. Painamalla COLOR-painiketta aukeaa värinpoimintatyökalu, jolla voidaan valita haluttu väri valolle tai ryhmälle. Liukuvalitsimien (sliders) muutokset lähetetään Hue-sillalle vasta, kun säätimestä päästetään irti, jotta vältetään turhia kutsuja ohjelmointirajapintaan.



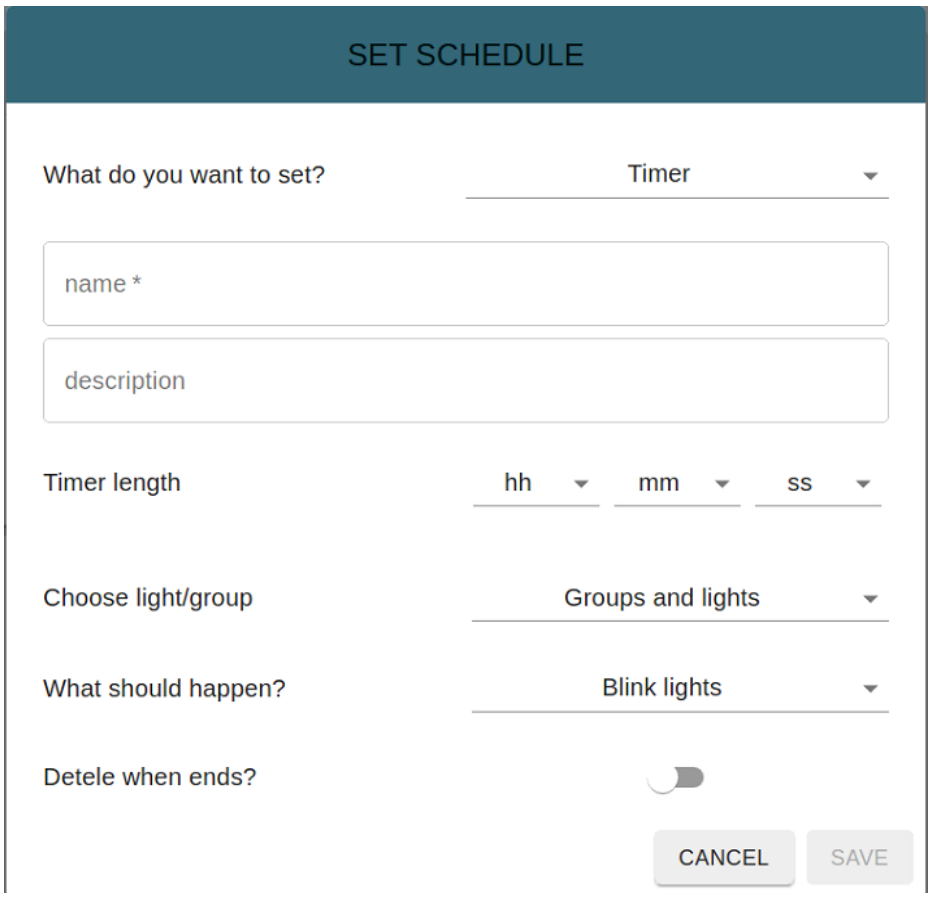
Kuva 10. Valojen hallintakortti, kun vierastila on pois päältä

Ajastukset-sivulla näytetään valmiiksi tehtyjä ajastuksia (kuva 11). Ajastuksia voidaan myös lisätä ja poistaa tällä sivulla. Ajastuksia poistetaan kuvasta 11 nähtävällä punaisella roskakoripainikkeella. Ajastuksia lisätään painamalla vihreää Lisää-painiketta sivun oikeasta yläkulmasta. Kuvassa 11 näkyy kaksi valmiiksi tehtyä ajastusta, joista toinen on ajastin ja toinen muistutus.



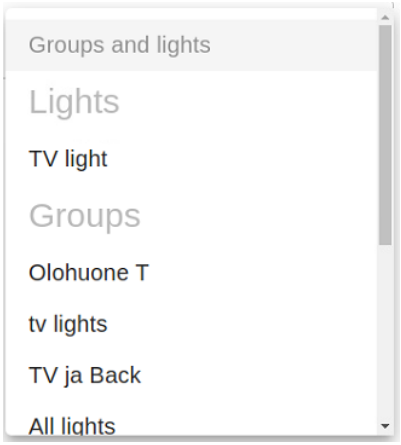
Kuva 11 Ajastukset-sivun näkymä tietokoneella

Kuvasta 12 nähdään modaali-ikkuna, joka aukeaa Lisää-painiketta painamalla. Save-painike on pois käytöstä, kunnes kaikki tarvittavat tiedot on annettu. Tiedoista on pakko syöttää kaikki muut, paitsi ajastuksen kuvaus. Nimeä ja kuvausta lukuun ottamatta kaikki tiedot syötetään valitsemalla tiedot pudotusvalikoista. Kuvassa 12 on esimerkki ajastimen luonnista, mutta päivämäärä ja kellonaika -ajastuksen, eli muistutuksen, lisäys tapahtuu samalla näkymällä. Erona näissä lisäyksissä on, että ajastimen pituuden (timer length) tilalla on päivämäärä- ja kellonaikavalitsin. Lisäksi poista loppuessa -ominaisuutta (delete when ends) ei ole mahdollista asettaa, kun lisätään muistutusta. Tällä ominaisuudella voidaan valita, halutaanko ajastin jättää muistiin myöhempää käyttöä varten. Jos ominaisuus on päällä, ajastin poistetaan Hue-järjestelmän muistista, kun sen aika on loppunut. Muistutukselle tämä ominaisuus ei ole tarpeen, koska sen päivämäärä ja kellonaika on jo mennyt, eikä sitä voida käyttää uudelleen järkevästi. Muistutus poistetaan aina Hue-järjestelmästä, kun se on suoritettu.



Kuva 12. Ajastimen lisäysnäkymä

Esimerkkinä pudotusvalikosta on valon tai ryhmän valinta (kuva 13). Kuvassa 13 valot-otsikon alla on vain yksi valo, koska sovelluksen käyttöhetkellä vain TV light -valo oli yhteydessä Hue-siltaan. Ryhmistä on puolestaan valittavissa kaikki, koska Hue-sillalta ei saa tietoa ryhmien tavoitettavuudesta.



Kuva 13. Pudotusvalikko, jolla voidaan valita, mitä valoa tai ryhmää käytetään ajastuksessa

6 POHDINTA

Philips Hue -älyvalot ovat käytössä hyvin toimivat ja helpot käyttää. Ainoina ongelmina on ilmennyt talossa vierailijoiden vaikeus hallita niitä ja ettei Philipsillä ole tarpeeksi monipuolista sovellusta tietokoneelle. Ratkaisuna näihin ongelmiin kehitettiin Reactilla verkkosovellus, jolla pystytään hallitsemaan Philips Hue -älyvaloja puhelimella ja tietokoneella. Vierailijat viipyivät yleensä muutamasta tunnista muutamaan päivään, joten he eivät halunneet asentaa erillistä sovellusta Hue-valojen hallitsemiseksi. Tehty verkkosovellus toimii selaimessa ilman erillistä kirjautumista.

Työssä käytettiin ensimmäistä kertaa Reduxia, joka osoittautui hyväksi tilanhallintakirjastoksi Reactille. Reduxin käytön aloittaminen vaikutti aluksi vaikealta, koska se vaati monta tiedostoa ja näiden tiedostojen väliset suhteet olivat epäselviä. Lyhyen käytön ja testaamisen jälkeen Reduxin toiminta vaikutti kuitenkin loogiselta ja tiedostojen tarkoitus selkeytyi. Työssä Redux helpotti paljon sovelluksen kehittämistä, koska kaikkia komponenttien tarvitsemia tietoja ei tarvinnut välittää komponenttien properties-kentissä.

Verkkosovellukseen tehtiin muutamia asetuksia, jotka helpottavat Hue-valojen hallintaa. Opinnäytetyön tekijää häiritsi muutama seikka, kun Hue-valoja hallittiin Philipsin sovelluksella. Philipsin sovelluksella valot palautuvat aina edelliseen kirkkauteen ja väriin, kun ne laitetaan päälle. Tehdyssä sovelluksessa on mahdollisuus valita oletuskirkkaus, jolle valot käynnistyvät. Lisäksi valot voidaan käynnistää aina valkoisiksi. Esimerkiksi Philipsin sovelluksella, jos valo sammutetaan punaisella värillä, se myös käynnistyy punaisena. Punainen valo ei valaise huonetta kovinkaan hyvin. Hue-valo on myös suhteellisen hidasta ja kömpelöä erikseen asettaa valkoiseksi heti päälle laittamisen jälkeen. Tämän takia verkkosovelluksen asetuksissa on ominaisuus, joka asettaa valot aina valkoisiksi, kun ne laitetaan päälle.

Jatkokehityksessä olisi mahdollista tehdä verkkosovelluksesta monipuolisempi. Philipsin ohjelmointirajapinnan avulla Hue-järjestelmään voi lisätä uusia laitteita,

muokata olemassa olevia laitteita sekä poistaa niitä. Näitä ominaisuuksia ei ainakaan vielä katsottu tarpeellisiksi, koska ne voidaan hoitaa Philipsin sovelluksella ongelmitta. Lisäksi valojen etäkäyttö talon ulkopuolelta olisi ollut mahdollista. Etäkäytölle ei myöskään katsottu olevan tarvetta, koska virallisen sovelluksen kautta tämäkin on mahdollista. Etäkäytön toteutuksessa olisi myös pitänyt pohtia tietoturvaa kattavasti sekä saada verkkosovellus julkiselle palvelimelle.

Verkkosovellus ei myöskään päivitä valojen tilaa taustalla automaattisesti. Jos Hue-valoja hallitaan toisella sovelluksella, valojen tila ei muutu verkkosovelluksessa, ennen kuin verkkosivu päivitetään tai verkkosovelluksessa siirrytään toiselle sivulle. Esimerkiksi verkkosovellus voi näyttää valon olevan pois päältä, vaikka se on laitettu päälle Philipsin sovelluksella. Taustapäivityksen lisäystä pohdittiin pitkään ja sitä oli jo osittain tehty, mutta se kuitenkin jätettiin toteuttamatta. Syynä oli huomattavasti kasvavien kyselyjen määrä ohjelmointirajapintaan. Lisäksi taustapäivitykselle ei ole erityistä tarvetta, koska valoja on harvemmin hallitsemassa useampi henkilö kerrallaan.

LÄHTEET

Crist, R. 2019. The complete guide to Philips Hue: Bulbs, smart features and lots of colors. Verkkoartikkeli. Luettu 2.3.2020. https://www.cnet.com/how-to/the-complete-guide-to-philips-hue/

Hue Home Lightning. 2019. Differences between 1st, 2nd, 3rd & 4th gen Philips Hue bulbs. Verkkoaineisto. Luettu 2.3.2020. https://huehomelighting.com/philips-hue-release-timeline/

Ighodaro, N. 2018. Why use Redux? Reasons with clear examples. Blogi. Luettu 7.3.2020. https://blog.logrocket.com/why-use-redux-reasons-with-clear-examples-d21bffd5835/

Introducing JSX. 2020. Verkkoaineisto. Luettu 30.3.2020. https://reactjs.org/docs/introducing-jsx.html

Korzunov, A. 2018. Restate – the story of Redux Tree. Verkkoartikkeli. Luettu 7.3.2020. https://hackernoon.com/restate-the-story-of-redux-tree-27d8c5d1040a

Krajka, B. 2015. The difference between Virtual DOM and DOM. Verkkoartikkeli. Luettu 5.3.2020. https://reactkungfu.com/2015/10/the-difference-between-virtual-dom-and-dom/

Nginx. N.d. What is NGINX? How different is it from Apache (for example)?. Luettu 16.3.2020. https://www.nginx.com/faq/what-is-nginx-how-different-is-it-from-e-g-apache/

Opensource.com. n.d. What is Docker?. Verkkoaineisto. Luettu 16.3.2020 https://opensource.com/resources/what-docker

Philips Hue. 2020. Aloituspakkaus E27. Verkkoaineisto. Luettu 2.3.2020 https://www2.meethue.com/fi-fi/p/hue-white-and-color-ambiance-aloituspakkaus-e27/8718699701352

Redux. 2020. Verkkoaineisto. Luettu 7.3.2020 https://github.com/reduxjs/redux

Signify Holding. 2019. Miten Philips Hue toimii?. Verkkoaineisto. Luettu 2.3.2020. https://www2.meethue.com/fi-fi/kuinka-se-toimii#bridge-controlled

Thinkwik. 2017. Why ReactJS is gaining so much popularity these days. Verkkoartikkeli. Luettu 5.3.2020. https://medium.com/@thinkwik/why-reactjs-is-gaining-so-much-popularity-these-days-c3aa686ec0b3

Wallenius, N. n.d. Mikä on Docker ja mitä hyötyä siitä on?. Luettu 16.3.2020. https://niklaswallenius.fi/teknologiat/mika-on-docker/