Mikael Heininen

SaaS-tuotteen kehitys

Tapaus Saunaäpp

Tekniikan ja innovaatiojohtamisen
akateeminen yksikkö
Automaatio- ja tietotekniikka
Diplomityö

Vaasa 2023

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen akateeminen yksikkö
Tekijä: Mikael Heininen
Tutkielman nimi: SaaS-tuotteen kehitys : Tapaus Saunaäpp
Tutkinto: Diplomi-insinööri
Oppiaine: Automaatio- ja tietotekniikka
Työn ohjaaja: Timo Mantere
Valmistumisvuosi: 2023 **Sivumäärä:** 59

TIIVISTELMÄ:

Teknologian nopea kehitys on tuonut uusia liiketoimintamalleja ja uusia tapoja toteuttaa perinteisiä asioita. Ennen ja myös nykyään moni taloyhtiö hoitaa saunavuorojensa hallinnoinnin kirjoittamalla paperille taloyhtiön saunavuorolistan, josta näkyy tulevan kuukauden saunavuorot taloyhtiössä. Tämä saunavuorolista kiinnitetään taloyhtiön ilmoitustaululle tai saunahuoneen oveen. Voi myös olla, että lista on tyhjä ja asukkaat käyvät kirjoittamassa siihen haluamansa vuoron. Tapoja on varmasti lähes yhtä paljon kuin taloyhtiöitä Suomessa. Tällaisessa tavassa saunavuorojen hallinnointi on vaikeaa asukkaalle, taloyhtiölle sekä isännöitsijälle. Tässä opinnäytetyössä keskitytään SaaS–tuotteen liiketoimintamalliin sekä tuotteen tekoon. Taloyhtiöiden saunanvarauksien hallintaan keskittyvä Saunaäpp viedään startup hengessä suunnittelusta lähes tuotantokuntoon. Saunaäpp tulee toimimaan SaaS-liiketoimintamallin mukaisesti jatkuva laskutteisena palveluna taloyhtiöille. Palvelussa pystyy luomaan käyttäjän taloyhtiön adminkäyttäjälle, joka voi luoda palveluun oman taloyhtiönsä asukkaat eli käyttäjät huoneistojen mukaan. Taloyhtiön asukkaat voivat palvelussa varata, muokata, poistaa, nähdä ja listata omat saunavuoronsa. Admin käyttäjä pystyy hallinnoimaan koko taloyhtiön saunavuorokalenteria eri tavoin, niin että palvelu toimii asukkaille mahdollisimman jouhevasti. Saunaäpp sovelluksen käyttöliittymä toteutetaan Reactilla ja taustajärjestelmä Nodejs:llä. Lisäksi käytössä on useampia muita teknologioita eri käyttökohteisiin liittyen. Opinnäytetyössä käsitellään erilaisia kysymyksiä, joita uuden tuotteen tuonti markkinoille aiheuttaa, esimerkiksi kilpailukenttään, laskutukseen ja markkinointiin liittyviä kysymyksiä. Tärkeää on tutkia olemassa olevaa kilpailukenttää, kuinka kireä se on ja onko tuotteella menestymisen mahdollisuuksia. Lisäksi kilpailijoista pystyy ottamaan mallia laskutuksen suunnitteluun. Markkinointi, varsinkin uudelle tuotteelle on tärkeää, jotta saadaan mahdollisimman paljon näkyvyyttä kilpailukentässä.

AVAINSANAT: SaaS, React, Startup, Liiketoimintamalli, B2B, Markkinointistrategia

Sisällys

Kuvat

1 Johdanto

Vahva teknologinen murros 2000-luvun alusta on modernisoinut, miten yritykset, yhteisöt ja yksityishenkilöt toimivat. Internetin käyttö ja sinne luotavat tuotteet ja palvelut ovat ottaneet valtaosan markkinaosuuksista erilaisista tuotteista ja palveluista, jotka ennen internettiä tehtiin muin tavoin. (He, 2010, s. 232). Järjestelmien luonti, tiedon tallennus, varastointi ja haku internetistä tuovat mukanaan hyötyjä, kuten helppous, vaivattomuus, säästöt sekä tehokkuus. (J. D. German ja G. O. Binoya, 2021, s. 231). Esimerkiksi hotellien vastaanotoissa harvoin törmää enää vanhanaikaiseen varauskirjaan, johon kynällä täytetään hotelliin sisäänkirjautuvien asiakkaiden tiedot, vaan hotellinvirkailija kirjaa sisäänkirjautuvien asiakkaiden tiedot hotellin internetissä toimivaan hotellin varausjärjestelmään, jonka avulla pidetään kirjaa hotellivieraista. Tämä helpottaa hotellin toimintaa: tiedot ovat paremmassa tallessa internetissä kuin fyysisessä varauskirjassa sekä varausta on helpompi hallinnoida internetissä. Näiden lisäksi, varausta on helpompi muuttaa, varaustilanteen näkee hotellin varausjärjestelmästä nopeasti sekä varaukseen on helpompi lisätä lisälaskuja muiden palvelujen ostamisesta.

Varausjärjestelmien tavoitteena on tehostaa käyttäjien ja varaustenhallinnoijien työtä. Voisi kuvitella, että varausjärjestelmissä ”yksi hoitaa kaiken”-ajatusmalli, jossa tehdään yksi varausjärjestelmä, jota voitaisiin käyttää monen eri palvelun tarkoitukseen olisi toimiva ratkaisu, mutta markkinaa tutkimalla huomataan, että asia ei ole näin yksinkertainen. Omia varausjärjestelmiä on rakennettu useiden eri palveluiden ympärille. Esimerkiksi taksit, hotellit sekä autonvuokrausyritykset luottavat omiin varausjärjestelmiinsä vuokrattavien resurssiensa hallinnointiin. Jos rakennettaisiin yksi varausjärjestelmä, ja oletetaan että se toimisi kaikilla toimialoilla niin usein tällaisesta ratkaisusta puuttuu juuri sen kyseisen toimialan tärkeimmät toiminnallisuudet. Tärkeimpien toiminnallisuuksien lisäksi, varausjärjestelmää kehitettäessä, palvelun helppokäyttöisyyteen tulisi kiinnittää suurin huomio. Jos palvelu ei ole helppo-käyttöinen, niin heikommat IT-taidot omaava käyttäjä on haluton käyttämään palvelua. Päätoiminnallisuuksien lisäksi, varausjärjestelmien tulisi toteuttaa myös muita

pienempiä, mutta tärkeitä toiminnallisuuksia. Esimerkiksi tietynlaisessa huoneiden-vuokraus-varausjärjestelmässä pelkän huonevarauksen tekeminen olisi riittämätöntä. Olisi tärkeää pystyä myös näyttämään vapaiden huoneiden määrä ja sijainti. Lisäksi olisi tärkeää pystyä muokkaamaan ja poistamaan omia varauksiaan sekä luoda mahdollisuus automaattisten ilmoituksien tekoon uusista vapaista huoneista tiettynä päivänä sähköpostin avulla. Edellä mainittujen asioiden lisäksi palvelun olisi hyvä sisältää mahdollisuus luoda korkeampien oikeuksien käyttäjä, joka pystyy hallinnoi-maan huoneidenvarausta ja koko järjestelmää. Tällä käyttäjällä olisi oma näkymänsä, josta näkee helposti ja nopeasti tämänhetkisen vuokrauksen tilanteen, paljonko huoneita on vuokrattu, paljonko huoneita on vielä vapaana, mitkä ovat vapaiden huoneiden hinnat sekä milloin varattuja huoneita vapautuu seuraavaksi. Korkeamman oikeuden käyttäjä pystyisi lisäksi luomaan uusia käyttäjiä palveluun, päivittää huoneiden tietoja ja hintoja. (Murin ja muut, 2021, s. 267)

Hyvin toimiva varausjärjestelmä voi olla yhtiön tärkein kilpailuetu. Varausjärjestelmässä olisi tärkeää hyvä taustajärjestelmä ja helppokäyttöinen sekä selkeä käyttöliittymä, jossa varauksen tekeminen olisi nopeaa ja vaivatonta. Palvelun käyttäjien tulisi jatkuvasti ymmärtää palvelussa ollessaan omat tämänhetkiset varaukset, ja miten niitä voi hallinnoida. Palvelun helppokäyttöisyys on tuotteen tärkein ominaisuus ja kilpailuetu. (Murin ja muut. 2021, s. 267-268)

Edelleen erilaisten yhteiskäytössä olevien tilojen varaukset suoritetaan usein kynän ja paperin avulla. Paperi pitää huolta varaustenhallinnoinnista ja kynällä kirjoitetaan uusi varaus. Suuri ongelma tässä tavassa toimia on se, että kuka vain pystyy muokkaamaan paperin sisältöä, tai se voidaan hävittää. Varausjärjestelmien luonti pilveen korjaa edellä mainitut ongelmat. Varausjärjestelmään voi luoda erilaisia keinoja resurssien hallinnointiin ja toimintaan. Lisäksi varausjärjestelmään voi luoda tunnistautumisen, jonka avulla käyttäjän täytyy kirjautua palveluun, ennen kuin hän pääsee muokkaamaan varauslistaa.

Diplomityössä tutkitaan, miten saadaan kehitettyä suunnittelusta tuotantoon asti saunavuorojen varauksiin erikoistuva SaaS-tuote. Tutkimuksessa käsitellään kysymyksiä liittyen tuotteen hinnoitteluun ja helppokäyttöisyyteen sekä millaisilla teknologioilla tuote olisi hyvä tehdä. Lisäksi pohditaan, miten uudelle tuotteelle saisi mahdollisimman suuren näkyvyyden ja tätä kautta palvelua käyttäviä asiakkaita.

1.1 Saunaäpp

Saunaäpp tulee olemaan internettiin tehty palvelu, jolla voidaan hallinnoida taloyhtiön tai muun järjestön saunavuoroja. Käyttäjille tarjotaan useampia toiminnallisuuksia saunavuorojensa hallinnointiin, kuten mahdollisuus kirjautua palveluun, varata itselleen vuoroja sekä pitää kirjaa vuoroistaan. Saunaäpp on toteutettu käyttäen moderneja teknologioita.

1.2 Kilpailijat

Vastaavanlaisia palveluita etsiessä internetistä, toistuu hakutuloksissa palvelu nimeltä ”SaunaOnline”. Palvelu toimii osoitteessa https://saunaonline.fi. SaunaOnlinen verkkosivun mukaan palvelun käyttö maksaa *39,0 €/kk (alviton)* sekä *6.0 %* jokaisesta varatusta vuorosta *(1,90 € min)*. Tällöin 50 asunnon taloyhtiössä, kahdeksalla varauksella viikossa *á 12,0 € / 1 tunti* saadaan palvelun kokonaishinnaksi: *39,0 € · 12 kk + 8 varausta/viikko · 52 viikkoa · 1,90 € = 1258,40 € + alv.* (SaunaOnline, 2023) Tämä tarkoittaa esimerkkitapauksessa keskimäärin vähän yli kaksi euroa vastikkeeseen lisää per asukas. Vaikea kysymys on, ovatko taloyhtiöt ja niiden asukkaat valmiita maksamaan tällaisesta palvelusta tuota summaa. Hinta tuntuu lähtökohtaisesti pieneltä ja SaunaOnlinen viimeisten vuosien liikevaihto tukee tätä ajatusta. Tosin tällä hetkellä, kun kaikki kustannukset nousevat korkean inflaation takia hinta voikin olla liian kor-kea. Hintaa, jota asukkaat ja taloyhtiöt ovat valmiita maksamaan olisi hyvä pyrkiä selvittelemään esimerkiksi erilaisin kyselyin. Palvelun avulla onnistutaan pääsemään irti

paperilapuin tehdyistä varauslistoista, niiden kiinnityksistä taloyhtiön ilmoitustau-luille, hallinnoinnista sekä vuorovarausten sekaannuksista, joten siihen nähden sa-noisin hinnan olevan kohtuullinen. Edellä mainittujen syiden takia palvelun käyttö myös säästää muissa kustannuksissa. Lisäksi digitaalisena palveluna toteutettu palvelu helpottaa saunan varausta ja varaustilanteen tarkastusta, koska ne onnistuvat jatkuvasti palvelun kautta. Tärkeää on toteuttaa saavutettava palvelu, jossa tehdään palvelusta mahdollisimman helppokäyttöinen, koska digitaalisten palvelujen käyttö ei ole kaikille helppoa.

Finder (2023) palvelun mukaan SaunaOnline on tehnyt vuosien 2018-2021 aikana melko tasaista liikevaihtoa. Vuonna 2018 198 000€, vuonna 2019 211 000€, vuonna 2020 138 000€ ja vuonna 2021 186 000€. Ottaen huomioon edellä kuvatun esimerkki-laskelman, näihin liikevaihtolukemiin pääsemiseksi tarvittaisiin noin 120 taloyhtiötä. Nämä luvut todistavat, että palvelulle on markkinoita.

Isännöintiliitto (2023) mukaan Suomessa on noin 50 000 taloyhtiötä, jotka ostavat isännöintipalvelua. Jos SaunaOnline pääsee noin 150 000 euron liikevaihtoon vain noin 120 taloyhtiöllä (esimerkkilaskelman mukaan) niin tällöin potentiaali palvelun tuottoisuudelle on erittäin suuri ottaen huomioon taloyhtiöiden määrän Suomessa.

1.3 Käyttöliittymän helppokäyttöisyys

Internetissä tulee usein vastaan palveluita, joissa ei ole mietitty palvelun käytettävyyttä kaikille ihmisille. Usein palveluiden tekijät ajattelevat, että palvelun käyttäjät pystyvät käyttämään normaalia näppäimistöä, kuulemaan ääniherätteet mitä palvelu mahdollisesti tekee, lukea tekstit ja kuvat mitä päätelaitteella näkyy sekä käyttää hiirtä. Näin ei kuitenkaan ole aina, koska on joukko ihmisiä, joilla on yksi tai useampia ongelmia toteuttaa edellä kuvattuja toimintoja. (Kavcic, 2005, s. 1024) Esimerkiksi huono näkö, värisokeus, heikko hahmotuskyky, kuurous tai sokeus ovat syitä, joiden takia edellä kuvatuista toiminnoista ei kaikki onnistu.

Käyttöliittymän helppokäyttöisyys on palvelun tärkeimpiä kriteerejä. Mitä helpompaa palvelua on käyttää, sitä parempaa käyttäjäpalautetta se saa. Palvelua kehitettäessä täytyy ymmärtää se, että digitaalisten palveluiden käyttö ei ole kaikille yhtä selkeää ja helppoa. Esimerkiksi usein seniorikansalaisilla verkossa toimivien sovelluksien käyttäminen on keskimääräistä vaikeampaa kuin nuoremmilla kansalaisilla. Tämä on tiedostettu ongelma, johon pitää kiinnittää suurta huomiota, koska usein tämä ihmisryhmä on ahkerin saunankäyttäjä ja taloyhtiöissä aktiivisimpia päätöksentekijöitä. Etelämäen (2021) mukaan tärkeimpiä yksittäisiä asioita, joita seniori-ikäiset haluavat verkossa toimivan palvelun käyttöliittymältä ovat selkeys, yksinkertaisuus ja helppo omaksuttavuus. Kontrastivärit ja fonttikoot nousivat myös esiin. Etelämäki (2021) jatkaa, että vanhusten mukaan palvelussa olisi tärkeää, että navigointi paikasta toiseen olisi tehty mahdollisimman selkeästi. Esimerkiksi olisi selkeää ymmärtää sijainti palvelussa jatkuvasti sekä miten pääsee etusivulle takaisin. Tärkeää olisi myös, että samanlaista toiminnallisuutta tekevät painikkeet olisivat samannäköisinä samoissa paikoissa jatkuvasti. Etelämäki (2021) pitää tärkeänä myös, että palvelun käyttöön pitäisi löytyä mahdollisimman helposti tukea.

Päätelaitteen, esimerkiksi puhelimen tai tietokoneen käyttöön on tuotettu useita eri avustavia teknologioita, joiden avulla laitteiden käyttö on helpompaa ja mahdollista mahdollisimman monelle. (Kavcic, 2005, s. 1025). Esimerkiksi Windows koneissa on usein Lukija – sovellus, jonka avulla esimerkiksi sokea käyttäjä pystyy lukemaan verkkosivun sisältöä. Lukija – sovellus lukee päätelaitteessa olevaa html sisältöä ääneen, jolloin palvelun käyttö päätelaitteella on mahdollista sokealle käyttäjälle. Tämän takia palvelun kehityksessä on otettava huomioon se, että se noudattaa saavutettavuuteen liittyvää WCAG (Web Content Accessibility Guidelines) standardia mahdollisimman paljon. Web Content Accessibility Guidelines (WCAG) 2.1 standardi määrittää käyttöliittymälle erilaisia minimikriteerejä helppokäyttöisyyden ja saavutettavuuden takaamiseksi. Tähän kuuluvat esimerkiksi fonttien koot, niiden värit, taustavärit, aputekstit eri elementeille, elementtien käyttö ja toiminta näppäimistökäytössä sekä

näytönlukijan toiminta. WCAG standardi on luotu, jotta verkossa toimivia palveluja pystyisi käyttämään mahdollisimman moni. (W3C, 2023) Saunaäpissä, ainoastaan saavutettavuusselosteessa mainitut ongelmat eivät tule noudattamaan WCAG 2.1 standardia, mutta muuten standardia noudatetaan. Ongelmien listaa pyritään pitämään mahdollisimman lyhyenä. WCAG 2.1 noudattamisen lisäksi pyritään pitämään mielessä aiemmin mainitut seniori-ikäisten ihmisten toiveista palvelun käyttöliittymälle. Esimerkiksi, jotta käyttäjä ymmärtää jatkuvasti sijaintinsa palvelun sisällä, olisi hyvä rakentaa palveluun sisäinen murupolku. Näiden lisäksi olisi hyvä tehdä perustoiminnoista käyttöohjeiden lisäksi videot esimerkiksi Youtube-palveluun, johon pääsee suoraan Saunaäpin sivuilta, esimerkiksi miten kirjaudun sisään, miten varaan vuoron, miten poistan oman vuoroni, miten muutan vuorojani ja miten tarkastelen taloyhtiön kalenteria. Videot näyttäisivät vaihe vaiheelta, miten edellä mainittu toimenpide voidaan tehdä. Nämä videot voivat sopia myös tuotteen markkinointiin taloyhtiöille ja järjestöille. Tulevaisuudessa palvelun sisään voidaan rakentaa myös tietynlainen chatti botti, joka antaa neuvoja ja ohjeita eri toimintojen tekoon.

Käytettävyystestauksessa selvitetään eri testaushenkilöiden helppoutta käyttää palvelua. Ennen tuotantoon vientiä olisi hyvä tehdä käytettävyystestausta hakemalla omasta lähipiiristä eri ikäisiä ja eri netinkäyttökokemuksia omaavia henkilöitä testaamaan palvelua. Näin saataisiin arvokasta palautetta palvelun senhetkisestä tilasta käytettävyyden näkökulmasta.

2 Teknologioiden käyttö

SaaS (Software as a Service) on erittäin suosittu tapa tarjota palveluita internetin välityksellä. Ideana on, että internettiin tehdään palvelu, jota voidaan käyttää maksua vastaan. Suuri hyöty on, että palvelun käyttöä voidaan laskutuksen perusteella vuokrata käyttäjille, jolloin palvelua ei tarvitse toteuttaa itse. Palvelu on jatkuvasti käyttövalmis internetin kautta sekä halpa käyttää. Palveluntarjoaja huolehtii, että palvelu toimii ja asiakkaan vastuulla ei ole muuta kuin käyttömaksun maksaminen. (Rahman & Subriadi, 2022, s. 31)

2.1 React

Reactin (2023) mukaan React on työkalu, jonka avulla voi rakentaa käyttöliittymiä. Käyttöliittymä rakennetaan useammista pienemmistä yksittäisistä komponenteista. Redux helpottaa tekemään Reactin avulla tehtyihin käyttöliittymiin tilanhallintaa (Redux, 2023a).

Saunaäpissä käyttäjän käyttöliittymä on toteutettu Reactilla käyttäen tilanhallintaan Redux kirjastoa. Reactin komponenteista on pyritty luomaan mahdollisimman uudelleenkäytettäviä. Komponentit on toteutettu funktionaalisin komponentein luokkakomponenttien sijasta ja useita funktionaalisten komponenttien etuja on käytetty.

2.1.1 Reactin komponentit

Reactilla tehdyt käyttöliittymät rakentuvat tehdyistä komponenteista ja niiden yhdistelmistä. Komponentit voidaan jakaa funktionaalisiin- sekä luokkakomponentteihin. Funktionaalinen komponentti on vain funktio, joka voi ottaa vastaan muuttujia ja se palauttaa Reactin elementtejä. (React, 2023) Kuvassa 1 on esimerkki funktionaalisesta React komponentista. Tämä funktionaalinen komponentti ottaa vastaan muuttujan nimi

ja koko komponentti palauttaa tekstin ”Tämä on funktionaalinen komponentti, jonka nimi on {name}”. Nyt {name} kohtaan tulee muuttuja, joka otettiin vastaan tähän komponenttiin.

```
1  import React from 'react'
2
3  const FunctionalComponent = ({name}) => {
4      return(
5          <>
6              Tämä on funktionaalinen komponentti, jonka nimi on {name}
7          </>
8      )
9  }
10 export default FunctionalComponent
```

Kuva 1. Reactin funktionaalinen komponentti

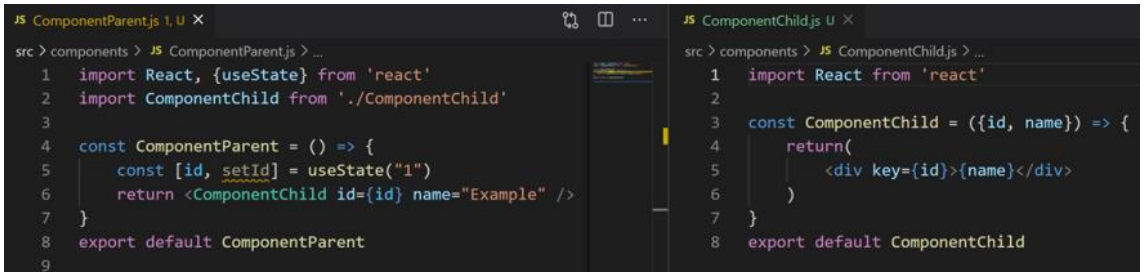
Sama toteutettuna luokkakomponentilla näyttää Kuvan 2 näköiseltä. Luokkakomponenttien käyttö vaatii aina avainsana class:n käytön sekä render() metodin. Näiden asioiden takia, luokkakomponenttien käyttö vaatii enemmän koodikirjoitusta.

```
1  import React from 'react'
2
3  class FunctionalComponent extends React.Component {
4      render() {
5          return <>Tämä on luokkakomponentti, jonka nimi on {this.props.name}</>
6      }
7  }
8  export default FunctionalComponent
```

Kuva 2. Reactin luokkakomponentti

Muuttujia pystyy lähettämään Reactissa komponentista toiseen. Komponentista toiseen lähettäessä, muuttujan tieto välittyy korkeammalta komponentilta alemmalle komponentille. Tämä ketju voi kulkea myös toiseen suuntaan, jolloin lähetetään muuttuja alemmalta komponentilta ylemmälle. (React, 2023)

Kuvassa 3 lähetetään komponentista ComponentParent tilamuuttuja id, jonka arvo on ”1” sekä paikallinen muuttuja name, jonka arvo on ”Example” lapsikomponentille ComponentChild. Lapsikomponentti ottaa muuttujat vastaan ja palauttaa html div-elementin jossa key arvona on lähetetty id sekä div:n arvona on lähetetty name. Reactissa useiden komponenttien luominen ja Kuvan 5 tapainen kerroksisuus esimerkiksi isä-lapsi, isä-lapsi-lapsi tai isä-lapsi-lapsi-lapsi suhteilla on ominaista. Muuttujien arvojen lähettäminen alaspäin isä-lapsi suhteella tekee nopeasti sovelluksen tilanhallinnasta epäselkeää, jos lapsikerroksia on useita.



Kuva 3. Muuttujan lähettäminen komponentista toiseen

2.1.2 Reactin hookit

Reactin hookit ovat tiettyyn tarkoitukseen tehtyjä funktioita, jotka alkavat avainsanalla ”use”. Hookkeja voi luoda myös itse. (React, 2023)

Kuvassa 4 on esimerkki useState hookin käytöstä. Siinä funktionaalinen komponentti ExampleOfuseStateHook palauttaa html button elementin, jossa ei aluksi lue mitään. Kun nappia painaa niin onClick funktiota kutsutaan, jolloin useState hookin setName muuntaa name:n arvon tyhjästä arvosta arvoon new name. Nyt napissa lukee teksti new name. useState hook voidaan tallentaa string tyyppisten muuttujien lisäksi esimerkiksi numeroita, objekteja, taulukoita tai boolean tyyppisiä muuttujia.

```
1  import React, {useState} from 'react'
2
3  const ExampleOfuseStateHook = () => {
4      const [name, setName] = useState('')
5      return <button onClick={() => setName('new name')}>{name}</button>
6  }
7  export default ExampleOfuseStateHook
```

Kuva 4. useState hook esimerkki

Kuvassa 5 on esimerkki useEffect hookin käytöstä. useEffect hook käytetään usein sivuvaikutusten hallintaan. Aina kun komponentti kiinnittyy, eli kun komponenttia ajetaan, useEffect hook ajetaan. Kun komponentin kiinnitys häviää eli komponentin ajaminen lopetetaan, kutsutaan return lausetta useEffectin sisällä kerran.

useEffectiä ajetaan aina vain kerran, kun komponentti kiinnityy. Lisäksi useEffectin lopussa olevan taulukon sisällä olevien muuttujien muuntuessa. Esimerkiksi Kuvan 5 olevassa tilanteessa useEffectiä ajetaan ensimmäisen kerran yhteen kertaan, kun komponentti ExampleOfuseEffectHook kiinnittyy. Sen jälkeen useEffectiä ajetaan vain ja ainoastaan silloin kun useEffectin lopussa olevan taulukon joku muuttuja muuttaa arvoaan. Tässä tilanteessa, jos name muuttujan tila muuntaa arvoaan, niin useEffect ajetaan.

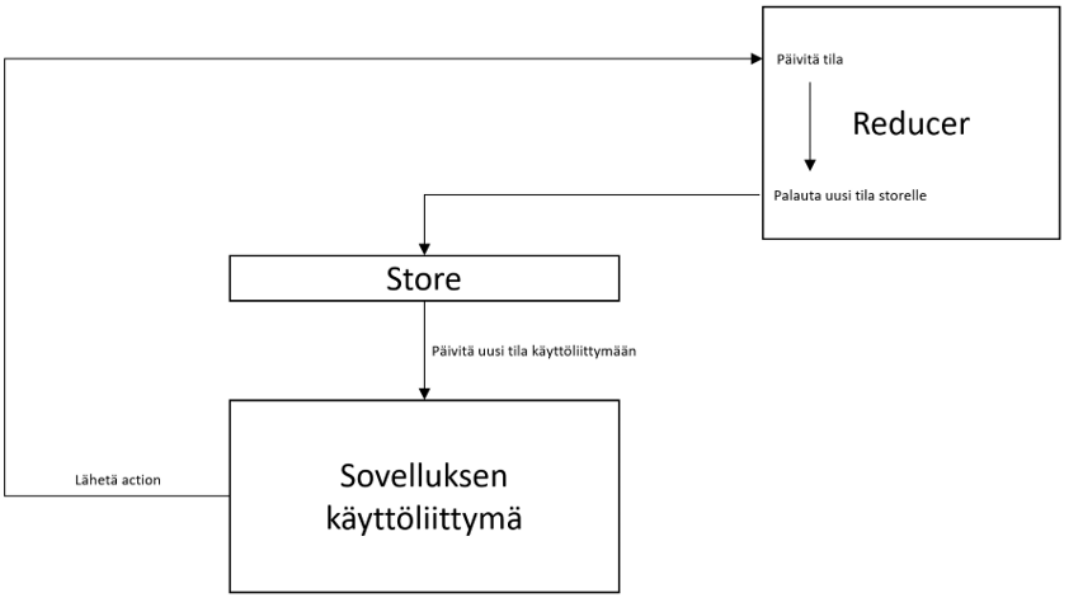
```
1  import React, {useEffect, useState} from 'react'
2
3  const ExampleOfuseEffectHook = () => {
4      const [name, setName] = useState('')
5      useEffect(() => {
6          console.log('Tätä kohtaan kutsutaan aina kun komponentti kiinnittyy')
7          return () => {
8              console.log('Tätä kohtaa kutsutaan aina kun komponentin kiinnitys häviää')
9          }
10     }, [name])
11     return <>Esimerkki useEffectin käytöstä</>
12 }
13 export default ExampleOfuseEffectHook
```

Kuva 5. useEffect hook esimerkki

2.1.3 Sovelluksen tilanhallinta Redux-kirjastolla

Redux kirjasto on avoimeen lähdekoodiin perustuva tilanhallinta kirjasto, jonka avulla sovelluksen muuttujien tilaa voidaan hallinnoida. Redux kirjaston avulla sinne lisättävien muuttujien arvot on saatavissa missä kohtaa sovelluksen hierarkiaa tahansa sekä Reduxiin voi lisätä uusia muuttujia missä kohtaa sovelluksen hierarkiaa tahansa. Redux DevTools on selaimen lisäosa, joka on esitetty Kuvassa 26. Lisäosan avulla sovelluksen tiloja ja käynnistyviä actioneita voi seurata helposti. (Redux, 2023a)

Keskiössä sovelluksen tilanhallintaa on store, actionit sekä reducerit. Reduxiin tallennettuja tietoja säilytetään storessa. Storeen voi tallentaa tietoa minkämuotoisena tahansa, esimerkiksi tauluina, objekteina tai string tyyppisenä. Kun käyttäjä haluaa tallentaa jotain Reduxin storeen, hän lähettää actioneita, jotka sisältävät tyypin ja lähetettävän arvon. Actionin prosessointi tapahtuu reducerissa, jonka tehtävä on päivittää storea. Uuden arvon saatuaan, store päivittää sovelluksen käyttöliittymän ajantasalle. Kuvassa 6 on havainnollistettu Reduxin toimintaa. (Redux, 2023b)



Kuva 6. Reduxin toiminta

Jotta komponentti voisi käyttää Redux kirjaston tilanhallintaa, täytyy Redux yhdistää Reactiin. React-Redux kirjasto tarjoaa Reactin yhdistämiseen Reduxin kanssa connect funktiota. Connect tarjoaa komponentille nykytilat sekä mahdollisuuden päivittää tiloja. Kuvassa 8 on esimerkki miten React komponentti yhdistetään Redux kirjastoon connect funktion avulla. Reduxissa olevat nykymuuttujien tilat tarjotaan Kuvassa 7 mapStateToProps funktiolla ja mahdollisuuden tilojen päivitykseen tarjoaa Kuvassa 7 mapDispatchToProps funktio. (React-Redux, 2023a)

```
export default connect(mapStateToProps, mapDispatchToProps)(ExampleComponent)
```

Kuva 7. React komponentin yhdistys Reduxiin connectin avulla

Funktionaalisiin komponentteihin React-Redux kirjasto on luonut hookit useDispatch ja useSelector, joiden avulla voi suoraan päästä käsiksi muuttujien nykytiloihin sekä päivittää tiloja. Tämä pienentää tarvittavan koodin määrää. (React-Redux, 2023b)

2.2 Nodejs

Nodejs on JavaScriptillä toimiva suorituspalvelu. Nodejs:llä voidaan rakentaa taustajärjestelmiä. (Nodejs, 2023)

Taustajärjestelmä Saunaäpissä on toteutettu Nodejs:llä käyttäen siitä tehtyä Express kehystä. Taustajärjestelmä sisältää erilaisia lohkoja, jotka ottavat käyttöliittymän pyyntöjä sisään, suorittavat tietyn toiminnallisuuden, esimerkiksi hakevat tietoa tietokannasta ja lähettävät vastauksen takaisin käyttöliittymälle. Taustajärjestelmä hoitaa toiminnallisten pyyntöjen lisäksi myös käyttäjän autentikoinnin.

2.3 Middlewaret NodeJS:ssä

Middleware on toiminnallisuus, joka toimii käyttöliittymän ja taustajärjestelmän välissä. Middlewaren tehtävä on mahdollistaa järjestelmän käytön sujuvuus ja helppokäyttöisyys. Middlewarejen käytön suurin hyöty on toteuttaa tietty toiminnallisuus kertaalleen, jolloin kun tätä halutaan käyttää useammin niin tätä kyseistä middlewarea voidaan vain kutsua. Middleware pääsee käsiksi pyyntöön, palautukseen ja mahdolliseen seuraavaan middlewareen. (Express, 2023)

Saunaäpissä käyttöliittymän ja taustajärjestelmän välissä toimivien middlewarejen tehtävät ovat samojen toiminnallisuuksien toteuttaminen vain kerran. Esimerkiksi käyttäjänhallinnassa käyttäjän oikeuksien tarkistus hoidetaan middlewaressa. Sitä kutsutaan monesta paikkaa, mutta se on toteutettu vain kerran yhteen middlewareen. Lisäksi myöhemmin käsiteltävän Rediksen toiminnallisuudet on toteutettu middlewareen.

2.4 Json Web Token

Json Web Token muodostuu kolmesta osasta, jotka on erotettu toisistaan pisteellä. Ensimmäisessä osassa on otsake, toisessa kuorma ja viimeisessä kryptograafinen allekirjoitus. Otsake sisältää tietoa itse pyynnöstä, esimerkiksi pyynnön tyyppi ja koko. Lisäksi otsake sisältää tietoa kryptograafisista operaatioista, joita tarvitaan tokenin luontiin ja sen avaukseen. Kuorma sisältää tokeniin liitettyä dataa. Kryptograafinen allekirjoitus varmistaa, että token on muodostettu siinä palvelussa mihin sitä yritetään käyttää. Tämä viimeinen osa on Json Web Tokenin muodostuksen salausavain. (Okta, 2023) Kuvassa 8 on esimerkki eräästä Json Web Tokenista.

Kuva 8. Json Web Token

2.4.1 Json Web Tokenit Saunaäpissä

Käyttäjien tunnistaminen Saunaäpissä on hoidettu Json Web Tokenien avulla. Json Web Tokenit ovat pyyntöjen ohessa lähetettäviä tiedonpätkiä, jotka sisältävät tietoa pyynnön tekijästä. Saunaäpissä on rooli taloyhtiökohtaiselle käyttäjälle ja tämän taloyhtiön adminille. Palvelussa tunnistautuminen on toteutettu Json Web Tokenien avulla niin, että käyttäjän kirjautuessa sisään luodaan taustapalvelun autentikaatio lohkossa käyttäjälle oma Json Web Token jsonwebtoken kirjaston sign komennolla. Sign komento ottaa parametrikseen ensin tokenin sisälle tallennettavat muuttujat objektina. Kuvassa 9 tokenin sisään tallennetaan käyttäjän taloyhtiön kalenterin id ja hänen oma talon numeronsa. Edellä mainitut tiedot on kirjautumisen yhteydessä haettu tietokannasta, kun käyttäjätunnusta ja salasanaa on vertailtu. Toiseksi parametriksi sign komento ottaa salausavaimen. Salausavaimen tulee olla satunnainen uniikki teksti, jota ei pysty arvaamaan. Koko salaus nojautuu tähän. Kolmanneksi parametriksi sign komento voi ottaa erilaisia valintoja muodostetulle tokenille. Kuvassa 9 on lisätty optionaalinen

parametri expiresIn, jolle annetaan arvo 30d. Tämä tarkoittaa, että token on voimassa 30 päivää tästä hetkestä.

```
const token = jwt.sign(
  {
    calendarid: data.id,
    housenumber: houseToBeFound.housenumber,
  },
  process.env.JWT,
  {
    expiresIn: "30d",
  }
);
```

Kuva 9. Json Web Tokenin muodostus

Tokenin muodostuksen jälkeen, se palautetaan käyttöliittymälle kirjautumisen yhteydessä. Tästä lähin jokaisen pyynnön yhteydessä on lähetettävä mukana tämä token, koska ennen jokaista suoritusta taustajärjestelmässä suoritetaan käyttäjän autentikaation tarkistus tokenin avulla.

2.4.2 Käyttäjien authorisointi pyyntöjen yhteydessä Saunaäpissä

Käyttöliittymän pyyntöjen yhteydessä lähetetään aina käyttäjältä Json Web Token pyynnön mukana. Ennen taustajärjestelmän suoritusta, suoritetaan käyttäjän autentikaatio lähetetyn Json Web Tokenin avulla. Lähetetty token päätyy autentikaatiota varten luotuun middlewareen. Middlewaressa käytetään jsonwebtoken kirjaston verify komentoa. Kuvassa 10 on edellä mainittu tapahtuma. Alussa lähetetty token otetaan pyynnön headerista talteen. Tämän jälkeen suoritetaan jsonwebtoken kirjaston verify funktio, joka palauttaa tokeniin kirjatut salaiset tiedot jos tokenin allekirjoitus on oikea. Verify funktio ottaa parametreikseen itse tokenin ja salausavaimen. Kun salaiset tiedot,

jotka tokeniin on lisätty saadaan, niin sen jälkeen tietoja voidaan käyttää Calendar tietorakenteiden etsintään tietokannasta. Mongoosen findOne funktio palauttaa ensimmäisen tietorakenteen joka täyttää ehdon _id : decodedToken.calendarId eli ensimmäisen Calendar tyyppisen tietorakenteen, jonka _id on sama kuin tokeniin kirjattu calendarid. Kun Calendar tietorakenne on löydetty, tutkitaan Calendar tietorakenteen housingUsersSeedkeysArray muuttujaa, onko tässä taulussa jokin objekti, jossa objektin housenumber on sama kuin tokenista saatu housenumber. JavaScriptin some funktio palauttaa boolean tyyppisen muuttujan true tai false niin, että jos yksikin taulun solu toteuttaa annetun ehdon niin silloin palautetaan true. Jos some funktio palauttaa true niin lisätään pyyntöön uudet muuttujat calendarid ja housenumber ja annetaan niille tokenista saadut salaiset tiedot. Lopuksi kutsutaan next() funktiota, joka vie kutsun takaisin taustajärjestelmään oikeaan paikkaan ja taustajärjestelmä kutsu aloitetaan. Jos some funktio palauttaa false, päädytään else lohkoon, jolloin palautetaan palautus suoraan käyttöliittymälle statuskoodilla 401 ja viestillä, että token ei ollut oikea.

```
const token = request.headers.authorization.split(" ")[1];
const tokenDecoded = jwt.verify(token, process.env.JWT);

Calendar.findOne({
  _id: decodedToken.calendarid,
}).then((data) => {
  if (
    data.housingUsersSeedkeysArray.some(
      (house) => house.housenumber === tokenDecoded.housenumber
    )
  ) {
    request.calendarid = tokenDecoded.calendarid;
    request.housenumber = tokenDecoded.housenumber;
    next();
  } else {
    response.status(401).json({
      message: "Token is not valid",
    });
  }
});
```

Kuva 10. Käyttäjän autentikaatio

2.5 MongoDB-Mongoose

Saunaäpissä käyttäjien tiedot on tallennettu MongoDb nimiseen tietokantaan. MongoDb on tietokanta, joka tarjoaa dokumenttityyppisen lähestymistavan sovelluksen tietojen mallinnointiin. (Mongodb, 2023) Saunaäpissä on mallinnettu esimerkiksi Calendar olio, joka sisältää tietoa yksittäisen taloyhtiön saunavuoro kalenterista (Kuva 11).


```
_id: ObjectId('64036b75c0f1f1397095d774')
housingSeedKey: "123"
▾ housingUsersSeedkeysArray: Array
  ▾ 0: Object
      _id: ObjectId('64036b75c0f1f1397095d775')
      housenumber: "A1"
      houseSeedKey: "$2b$10$M5MSXQIHF71ab3zffYSxrOQvA4V3o32IxlWlHFZ56oz2SJ3ZLSle2"
  ▾ 1: Object
      _id: ObjectId('64036b75c0f1f1397095d776')
      housenumber: "A3"
      houseSeedKey: "$2b$10$T45VPvvAmaZW.w818iCynuoir026M0f7hotPE0lFjroMle6d9oZCG"
  ▾ 2: Object
      _id: ObjectId('64036b75c0f1f1397095d777')
      housenumber: "A2"
      houseSeedKey: "$2b$10$xQz1I6tL5w3pGAomB5aOLu6lPD.7OYbpvgmNjTQJalqOtes81U1Hu"
  ▾ 3: Object
      _id: ObjectId('64036b75c0f1f1397095d778')
      housenumber: "A4"
      houseSeedKey: "$2b$10$0Xklq5lPCZOnEQDom2q8q.67h3GKg1F6FpW5Ak.3fH70nq.fy/o2i"
owner: "testaaja"
▾ housingInformation: Object
    _id: ObjectId('64036b75c0f1f1397095d779')
    houseName: "Asunto Oy RauhankatuTESTI"
    YTunnus: "Y12345-90"
    address: "Rauhankatu 12a 20100, Turku"
  ▾ ruleForReservations: Object
      _id: ObjectId('64036b75c0f1f1397095d77a')
      amountOfTimesSaunaCanBeReservedPerWeek: 1
      amountOfTimesLaundryRoomCanBeReservedPerWeek: 3
▾ reservations: Array
  ▾ 0: Object
      _id: ObjectId('640c80b11e0e7448d4e2bdd6')
      name: "A2"
      date: "2023-03-11 20:00"
```


Kuva 11. Calendar-olio

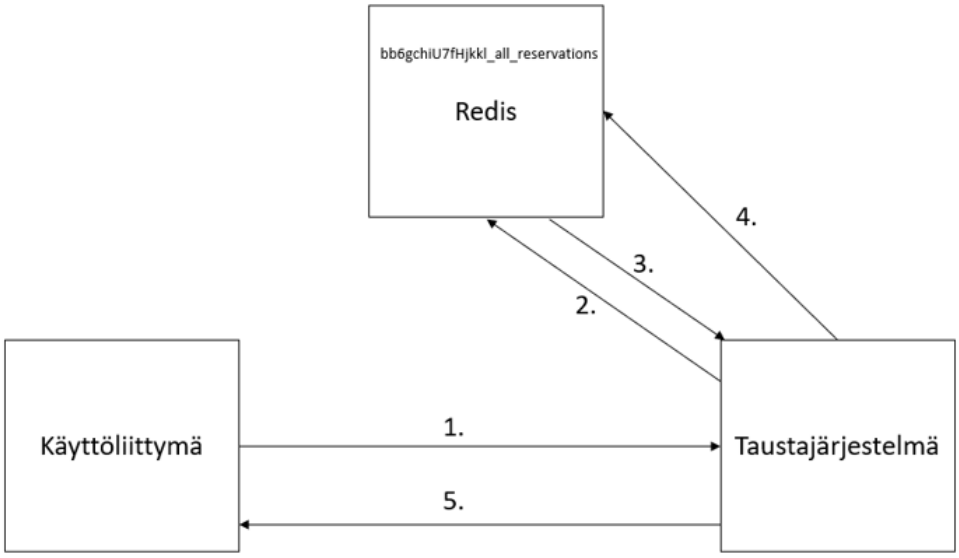
Kuten Kuvasta 11 huomataan, Calendar-olio sisältää uniikin _id muuttujan, joka on parametri, jolla taloyhtiön kalenteria voi hakea. Lisäksi Calendar-oliosta löytyy muuttujat

housingSeedKey, housingUsersSeedkeysArray, owner, housingInformation sekä reservations. housingSeedKeytä asukas käyttää kirjautumiseen oman henkilökohtaisen houseSeedKeyn kanssa. housingUsersSeedkeysArray sisältää taloyhtiössä olevat asukkaat. Jokaisesta asukkaasta on tallennettu houseNumber eli asunnonnumero sekä heidän henkilökohtainen salasanansa, jota voi käyttää kirjautumiseen palveluun housingSeedKeyn kanssa. Henkilökohtainen salasana on tallennettu häshinä bcrypt kirjaston avulla. owner muuttuja kertoo kenen adminin hallussa taloyhtiön on. Tämä on ajateltu kuuluvan isännöitsijälle. Adminilla on palvelussa oikeuksia lisätä käyttäjiä taloyhtiön kalenteriin, poistaa poismuuttaneita, muokata taloyhtiön tietoja, lisätä asukkaalle vuoro, poistaa asukkaalta vuoro ja seurata varattuja saunavuoroja. housingInformation sisältää tietoa taloyhtiöstä, sekä muuttujan ruleForReservations, johon admin voi tallentaa taloyhtiön sääntöjä vuorojen varauksille. Tämän hetken sääntöjä ovat montako kertaa tietty asukas voi varata saunavuoron viikossa sekä montako kertaa asukas voi varata pesutuvan viikossa. Nämä rajoitteet eivät tällä hetkellä ole vielä koodissa käytössä sekä pesutuvan varaus on uusi toimialue, johon saunaäppiä on ajateltu vietävän eli, että saunaäpissä voisi varata myös pesutupaa. reservations taulu sisältää tietoa kaikista varauksista. Tässä on mukana tulevat ja menneet. Jokaisesta tallennetusta vuorosta tallennetaan taloyhtiön asukkaan houseNumber sekä ajankohta. Koodissa määritellään ajankohdan mukaan, onko vuoro jo mennyt vai vasta tulossa.

2.6 Redis

Redis on avoimeen lähdekoodiin perustuva tiedontallennuspaikka, jota voidaan käyttää tiedonväliaikaiseen säilytykseen (Redis, 2023). Redis toimii Saunaäpissä käyttöliittymän ja taustajärjestelmän välissä tiedontallennuspaikkana. Saunaäpin Redikseen tallennetaan tallennettavia arvoja avainten avulla tietyiksi ajoiksi. Esimerkiksi avaimeen bb6gchiU7fHjkkl_all_reservations voidaan tallentaa tietyn taloyhtiön kyseisen hetken kaikki varaukset 30 minuutiksi. bb6gchiU7fHjkkl on tässä tilanteessa nyt kyseisen taloyhtiön (Calendar-olion) uniikki id tietokannassa. Saunaäpissä Redistä käytetään vähentämään tietokannasta haettavia tietoja. Kun jokin tietty resurssi on haettu

tietokannasta, tallennetaan se 30 minuutiksi Redikseen. Jos samaa resurssia haetaan uudestaan 30 minuutin aikaikkunassa, palautetaan tieto Rediksestä suoraan ilman tietokantahakua. Näin pystytään vähentämään tietokannasta haettavia hakuja.



Kuva 12. Rediksen käyttö Saunaäpissä

Kuvassa 12 on havainnollistettu Rediksen käyttöä Saunaäpissä. (1) Käyttöliittymä tekee pyynnön taustajärjestelmään hakien kaikkia tietokannassa olevia vuoroja taloyhtiössään. Taustajärjestelmä poimii taloyhtiön id:n pyynnön mukana tulevasta Json Web Tokenista. (2) Saadun Id:n avulla, taustajärjestelmä etsii Rediksestä löytyykö avaimella bb6gchiU7fHjkkl_all_reservations mitään arvoa. (3) Tässä tilanteessa Rediksestä ei löydy mitään arvoa tällä avaimella, jolloin taustajärjestelmä tekee kyselyn tietokantaan saadakseen tämän taloyhtiön kaikki varaukset. (4) Varausten tietokannasta saamisen jälkeen, taustajärjestelmä tallentaa saadun tiedon avaimella bb6gchiU7fHjkkl_all_reservations Redikseen esimerkiksi 30 minuutiksi. (5) Tämän jälkeen taustajärjestelmä palauttaa kaikki varaukset käyttöliittymälle, joka pääsee esimerkiksi näyttämään ne käyttäjälle. Samalla tavalla Redikseen voidaan tallentaa tietoa useammalla eri avaimella.

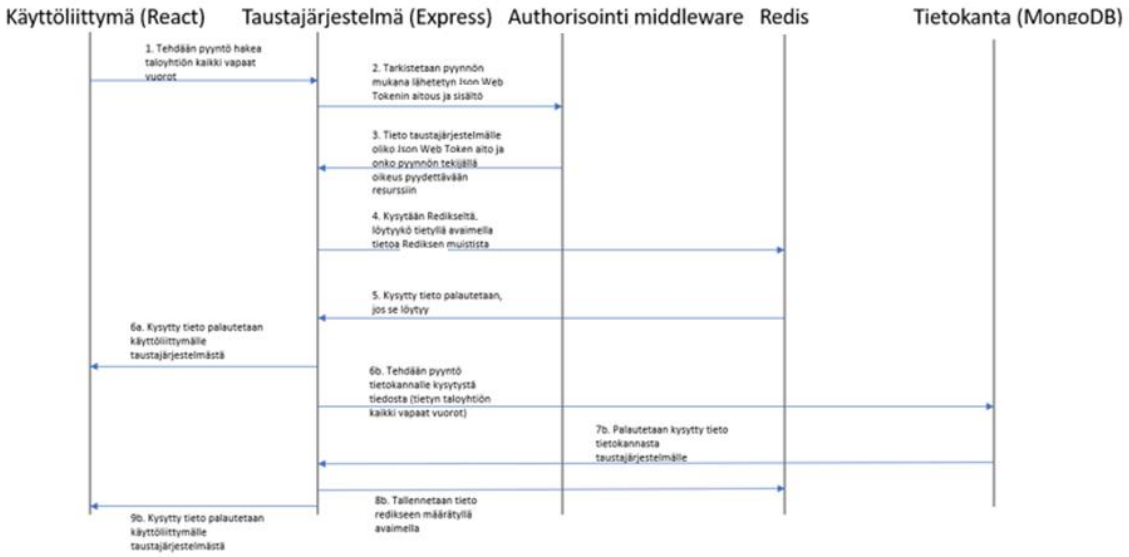
Edellä kuvatussa tilanteessa, usein aikaa vievin prosessi on tietokantahaku. Käyttäjän tehdessä vastaava pyyntö uudestaan, tieto löytyy nyt Rediksestä suoraan, joten tietokantahakua ei tarvitse tehdä ollenkaan. Saunaäpissä ollaan keskimäärin saatu vähennettyä Rediksen avulla hakuja yli 90% kun tieto löytyy suoraan Rediksestä.

Rediksen käytön kanssa pitää olla tarkkana, että tieto Rediksessä vastaa jatkuvasti tietoa tietokannassa. Edellä kuvatussa esimerkissä haettiin tietyn taloyhtiön kaikki vuorot. Jos tehdään operaatio, jossa poistetaan tästä taloyhtiöstä tietty vuoro, niin tietokantaan tekemisen lisäksi tämä muutos tulee tehdä myös Redikseen.

2.7 Saunaäpin vapaiden vuorojen haku esimerkki

Kuvassa 13 on kuvattu esimerkkitapaus tietyn taloyhtiön kaikkien vapaiden vuorojen haun suoritusjärjestyksestä. Muut palvelun operaatiot noudattavat suurin piirtein samaa kaavaa. (1) Ensin tehdään käyttöliittymältä pyyntö taustajärjestelmälle. (2) Tämän jälkeen taustajärjestelmä ohjaa pyynnön authorisointi palveluun, jossa tarkistetaan pyynnön tekijän lähettämän Json Web Token salaus ja sisältö kohdan 2.4.3 selostuksen mukaisesti. (3) Json Web Tokenin tarkistuksen jälkeen palautetaan tieto taustajärjestelmälle, onko salaus oikea. Jos salaus ei ole oikea, palautetaan tässä kohtaa suoraan käyttöliittymälle tieto Json Web Tokenin virheellisyydestä. Jos salaus on oikea, Json Web Token sisältää sisäisenä muuttujanaan jo etsittävän taloyhtiön kalenterin uniikin id:n, jonka avulla taloyhtiön kalenteri etsitään tulevissa vaiheissa, kuten kappaleessa 2.4.1 on selitetty. (4) Kun authorisointi tarkistus on onnistunut, taustajärjestelmä kysyy redikseltä, löytyykö määrätyllä avaimella tietoa redikseen tallennetusta resurssista, kuten kappaleessa 2.6 on selitetty. Kohdassa (5) rediksestä löydetty tieto tai tieto siitä ettei tietoa löytynyt palautetaan taustajärjestelmälle. (6a) Jos tieto löytyi rediksestä, se palautetaan tässä kohtaa taustajärjestelmästä käyttöliittymälle. (6b) Jos tietoa ei löytynyt rediksestä, tehdään taustajärjestelmästä pyyntö tietokantaan, ja palautetaan tieto tietokannasta (7b). (8b) Tallennetaan tieto redikseen tietyksi ajaksi, esimerkiksi 30 minuutiksi, jotta jos tämän ajan aikana tehdään vastaavanlainen pyyntö

niin tieto voidaan palauttaa jo kohdassa 6a käyttöliittymälle ja vältetään kysely tietokannasta. Viimeisessä kohdassa 9b tieto palautetaan käyttöliittymälle.



Kuva 13. Taloyhtiön kaikkien vapaiden vuorojen haku esimerkki

2.8 Cypress

Yksi sovelluskehityksen tärkeimmistä asioista on kehitettävien toiminnallisuuksien testaus. Kattavat testaukset luovat säästöjä monin tavoin. Mitä aikaisemmin virheellinen toiminnallisuus sovelluksessa tunnistetaan, sen vähemmän odotusarvoisesti sen korjaus tulee maksamaan. Lisäksi testien kirjoittaminen vähentää manuaalisen testauksen määrää. (IBM, 2023)

Kattavien testien kirjoittaminen luo myös varmuutta kehittäjälle uuden toiminnallisuuden luonnin jälkeen niin, että kehittäjä voi ajaa automaattiset testit ennen kuin on lisännyt uuden toiminnallisuuden koodikannan julkiseen haaraan. Näin hän voi varmistua ettei oma tuotos riko muita toiminnallisuuksia ennen kuin liittää tekemänsä toiminnallisuuden koodikannan julkiseen haaraan.

Cypress on avoimeen lähdekoodiin perustuva integraatiotestipaketti. Cypresissa voi testata toiminnallisuuksia käyttäjän näkökulmasta. (Cypress, 2023) Erilaisia testattavia toiminnallisuuksia Saunaäpissä ovat esimerkiksi kirjautuminen sisään tulee onnistua, saunavuoron pitää pystyä varaamaan sekä varatun saunavuoron pitää pystyä poistamaan.

```
2  describe('Logging-inhabitant in', () => {
3    it('should log inhabitant in', () => {
4      cy.visit(Cypress.config().baseUrl)
5      const credentials = {
6        housingPassword: 'esimerkkiSalasana',
7        housePassword: 'esimerkkiSalasana'
8      }
9      cy.logInInhabitant(credentials)
10     cy.get('[data-testid="pick-day"]').should('have.text', 'Valitse päivä')
11   })
12 })
```

Kuva 14. Käyttäjän kirjaaminen sisään - testaus Cypressillä

```
2  Cypress.Commands.add('logInInhabitant', (userCredentials) => {
3    cy.get('[data-testid="home-reserve-button"]').click()
4    cy.get('#taloyhtio-salasana').type(userCredentials.housingPassword)
5    cy.get('#asunto-salasana').type(userCredentials.housePassword)
6    cy.get('[data-testid="home-check-inhabitant-credentials"]').click()
7  })
```

Kuva 15. logInhabitant apufunktio

Kuvissa 14 ja 15 testataan Cypressillä, että käyttäjä voi kirjata itsensä sisään. Ensiksi Kuvan 14 rivillä 4 laitetaan testiympäristö kulkeutumaan Cypressin asetustiedostossa osoittamaan verkko-osoitteeseen. Tämän verkko-osoitteen tulee olla se osoite, jossa testit halutaan ajaa, esimerkksi http://localhost:3001. Kun päästään kyseiseen verkko-osoitteeseen luodaan credentials objekti, joka sisältää kaksi attribuuttia: housingPassword ja housePassword. Saunaäpp on rakennettu niin, että sisäänkirjautuminen käyttäjälle vaatii taloyhtiön oman salasanan ja henkilökohtaisen oman salasanan syötön. Tämän jälkeen kutsutaan Cypresillä tehtyä funktiota

logInInhabitant, joka on määritelty Kuvassa 15. logInInhabitant funktio ottaa vastaan userCredentials nimisen muuttujan. Kuvassa 15 rivillä 3 käsketään Cypressiä klikkaamaan sellaiseen HTML attribuuttiin, joka sisältä data-testid:n home-reserve-button. Tämän jälkeen rivillä 4 etsitään HTML attribuutti, joka sisältää id:n taloyhtio-salasana ja tähän kirjoitetaan saadun userCredentialsin housingPassword arvo. Samaan tapaan toimitaan rivillä 5 HTML attribuutti asunto-salasana:lle. Salasanojen kirjoitusten jälkeen rivillä 6 etsitään HTML attribuutti, jonka data-testid on home-check-inhabitant-credentials ja klikataan sitä. Tämän jälkeen suoritus jatkuu Kuvan 14 rivillä 10, jossa oletetaan että HTML attribuutti, jonka data-testid on pick-day sisältää tekstin ”Valitse päivä”. Tämä kyseinen HTML attribuutti tulee näkyviin vain jos käyttäjän kirjautuminen palveluun onnistuu. Tästä voidaan päätellä, että jos kyseinen HTML attribuutti tulee näkyviin niin silloin kirjautuminen on onnistunut, jolloin koko testi on myös onnistunut.

Edellä olevassa testissä käytettiin Cypressin testejä helpottavia toiminnallisuuksia kuten click(), .should.have.text ja .type. Kyseistenkaltaisia helpottavia toiminnallisuuksia on Cypressissä suuri määrä, joiden avulla ja kekseliäällä käytöllä voidaan luoda erittäin kattavat testit sovelluskehityksen tueksi.

2.9 Docker kehityksen apuna

Docker on työkalu jolla voi ajaa sovelluksia eristetyissä konteissa. Ajettavien konttien sisältöä kutsutaan valokuvaksi. Valokuvat sisältävät kaiken tiedon ajettavasta sovelluksesta, kuten lähdekoodin ja muut asetukset. Kuka tahansa voi tehdä sovelluksistaan valokuvia ja niitä voi julkaista myös yhteiseen käyttöön. (Docker, 2023). Esimerkiksi isot yritykset ovat julkaisseet omia tuotteitaan yhteiseen käyttöön. Tämän avulla tuotteita voi helposti ajaa Dockerin kautta eikä itse tuotteita tarvitse ladata omalle koneelle.

Dockerissa pystyy tekemään asetustiedoston, johon voi kirjata useamman ajettavan valokuvan. Tämä esimerkiksi nopeuttaa kehitystyötä, kun kehityksen aikana käytettävät

sovellukset saa nopeasti päälle. Tällaista asetustiedostoa kutsutaan docker compose tiedostoksi.

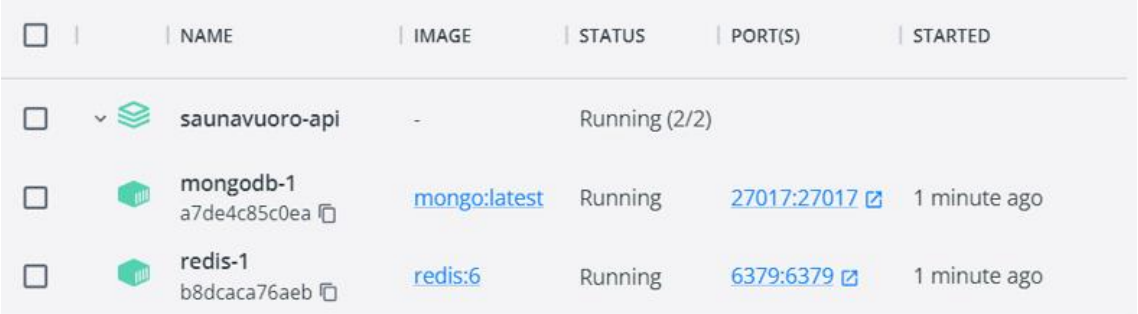
Saunaäpissä on hyödynnetty Dockeria kehitystyön apuna. Sovellukset Redis ja Mongodb ajetaan kehitystyön aikana suoraan Dockerissa. Tämän takia kyseisiä sovelluksia ei ole tarvinnut ladata koneelle. Docker compose asetustiedoston avulla Redis ja Mongodb saadaan nopeasti päälle vain yhdellä komennolla ”docker compose up –build”.

```
version: "3.8"
services:
  mongodb:
    image: mongo:latest
    ports:
      - 27017:27017
    volumes:
      - testidumppi:/data/db
  redis:
    image: redis:6
    ports:
      - 6379:6379
volumes:
  testidumppi:
```

Kuva 16. Docker compose asetustiedoston sisältö

Kuvassa 16 on määritetty käytetty Docker asetustiedosto yaml tiedostotyyppinä. Yaml on helppolukuinen tiedostotyyppi, joka tulee sanoista Yet Another Markup Language (Red hat, 2023). Docker composen versioksi on määritetty 3.8. Palvelut mongodb ja redis on määritetty omalle services otsikon rivilleen. Mongodb käyttää uusinta valokuvaa, joka löytyy dockerista, kun taas Redis käyttää versiota 6. Mongodb on määritetty käyttämään docker verkon sisällä ja verkosta ulospäin porttia numero 27017. Nyt jos otamme yhteyden paikallisen isännän (localhost) porttiin numero 27017 niin siellä toimii määritetty mongodb kontti. Redis toimii verkon sisällä ja ulospäin porteissa numero 6379. Docker compose asetustiedostoon on määritetty myös volyymi nimeltä testidumppi. Volyymi on tallennuspaikka tiedolle, jota mongodb tallentaa. Volyymi tallentaa tietoa docker kontin ulkopuolelle. Ilman volyymin määritystä, jos mongodb kontin poistaa niin

kaikki tallennetut tiedot häviävät. Kuvassa 17 nähdään Redis ja Mongodb ajossa dockerin graafisesta käyttöliittymästä katsottuna.



Kuva 17. Docker kontit ajossa

2.10 Integroitavuus muihin palveluihin

Integroinnit muihin palveluihin ovat nykyaikaa ja ne helpottavat palvelun käyttöä. Tulevaisuudessa olisi hyvä lisätä esimerkiksi sisäänkirjautumiseen mahdollisuus kirjautua muiden tunnettujen palveluiden avulla. Esimerkiksi Facebook ja Twitter tarjoavat mahdollisuuden kirjauttaa käyttäjä sisään heidän sovelluksiin ja tätä tietoa voidaan käyttää ulkoisissa sovelluksissa esimerkiksi Saunaäpissä. Tällaista prosessia kutsutaan nimellä OAuth (Open-Authorization) 2.0. (Internet Engineering Task Force, 2012)

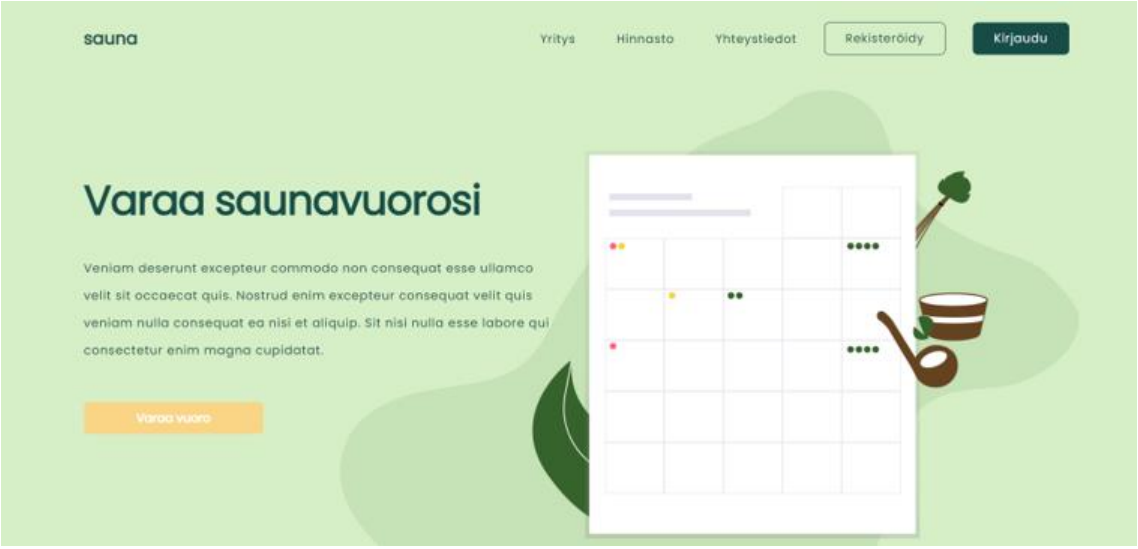
Yleinen idea tällaisessa prosessissa on se, että Saunaäpp sivulla olisi jokin painike, jossa voisi lukea esimerkiksi ”Kirjaudu Facebookilla”, joka ohjaisi käyttöliittymän Facebookin sisäänkirjautumisvaiheeseen. Tähän vaiheeseen kun kirjoittaa käyttäjätunnuksen ja salasanan niin päädytään varmistusdialogiin, jossa Facebook kysyy, että haluatko antaa prosessin määritykseen valitut tiedot sovellukselle Saunaäpp? Nämä tiedot voivat olla esimerkiksi kyseisen kirjautuvan käyttäjän käyttäjätiedot. Tapoja miten tiedot lopulta päätyvät Saunaäpille on useampia. Esimerkiksi Facebook voi palauttaa Json Web Tokenin joka sisältää kyseiset tiedot tai esimerkiksi tietynlaisen salausavaimen, jonka avulla kysellä tietoja tietystä Facebookin rajapinnasta. Lopulta kun tiedot Facebookista päätyvät Saunaäpille niin Saunaäpin tehtävä olisi jotenkin yhdistää tämä Facebookin

käyttäjä ja mahdollisesti jo Saunaäppiin luotu käyttäjä yhteen. Yksi tapa voisi olla esimerkiksi se, että tämän Facebook integraation voi tehdä vain kirjautunut käyttäjä omissa tiedoissaan. Nyt kun kirjautunut käyttäjä tekisi saman prosessin niin tuleva Facebook käyttäjä voidaan helposti yhdistää kirjautuneeseen käyttäjään. Jatkossa kirjautumisen tulisi olla mahdollista Saunaäpin tunnuksilla sekä Facebookilla.

Vuorojen maksamiseen on ajateltu otettavan käyttöön toimintoa, jonka avulla varatut vuorot voitaisiin maksaa suoraan sovelluksessa ulkoisen palveluntarjoajan kautta, esimerkiksi Stripen avulla. Ongelma näissä on kuitenkin hinta, ulkoinen palveluntarjoaja ottaa välistä muutamia prosentteja maksetun tuotteen hinnasta, joka on pitkässä juoksussa suuri hinta taloyhtiölle. Tämän takia Saunaäppiin voisi tehdä esimerkiksi tietynlaisen automaattisen laskutusjärjestelmän, jossa esimerkiksi kuukausittain palvelu kävisi läpi kaikki pidetyt saunavuorot huoneistoittain ja lähettäisi käyttäjien sähköpostiin huoneistokohtaiset laskut automaattisesti. Näin voitaisiin minimoida manuaalista laskujen pyörittelyä.

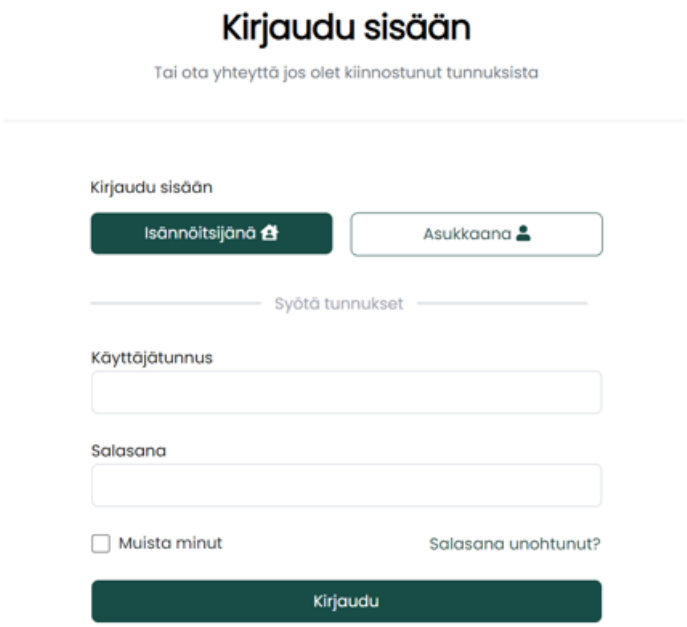
3 Saunaäpin toiminnallisuuksia

Saunaäpp on tällä hetkellä aktiivisessa kehityksessä. Kuvasta 18 näkyy, että Saunaäpin etusivulla voidaan siirtyä kirjautumaan palveluun sekä selaamaan yleisiä asioita palvelusta.



Kuva 18. Saunaäpin etusivu

Kun isännöitsijä haluaa kirjautua sisään päädytään Kuvan 19 näkymään. Jokaisella isännöitsijällä on Saunaäpissä henkilökohtainen käyttäjätunnus ja salasana. Kuvassa 20 on asukkaan kirjautumisnäkymä. Jokaisella taloyhtiöllä on taloyhtiökohtainen salausavain. Lisäksi jokaisella asukkaalla on oma henkilökohtainen salausavain.



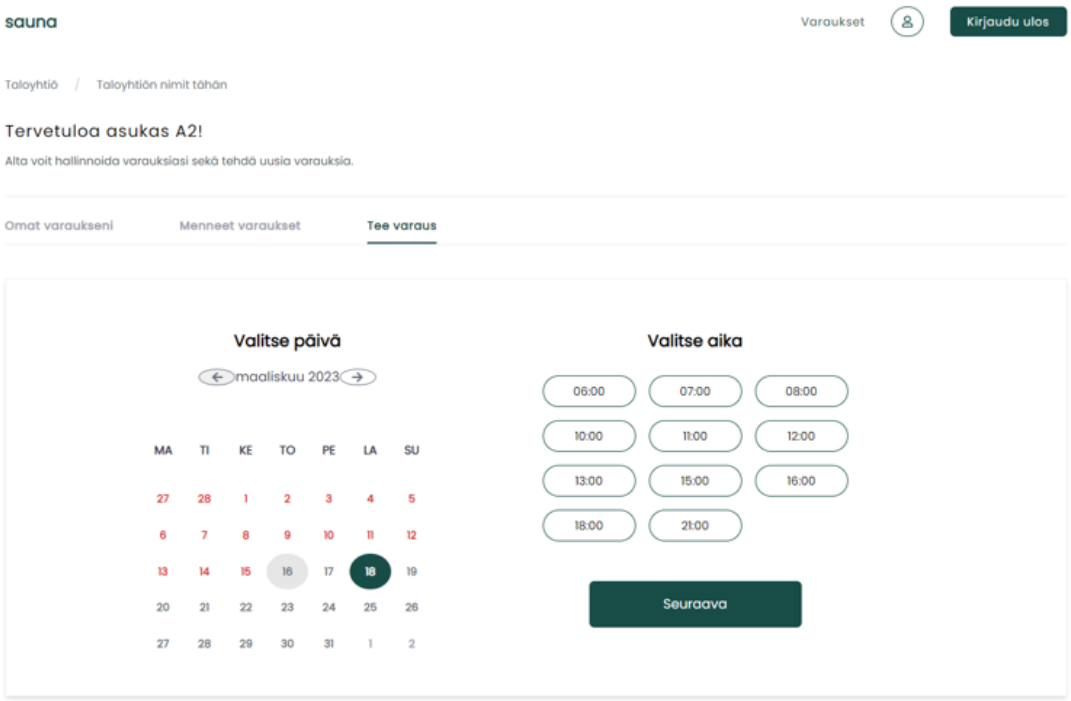
Kuva 19. Sisäänkirjautumisnäkymä isännöitsijälle

Kuva 20. Sisäänkirjautumisnäkymä asukkaalle

Kuvassa 21 on asukas A2 kirjautunut sisään. Kirjautumisen jälkeen etusivullaan käyttäjä näkee varatut vuoronsa ”Omat varaukseni” näkymässä. Menneet varaukset listataan samaan tapaan.

Kuva 21. Asukas A2 kirjautuneena sisään

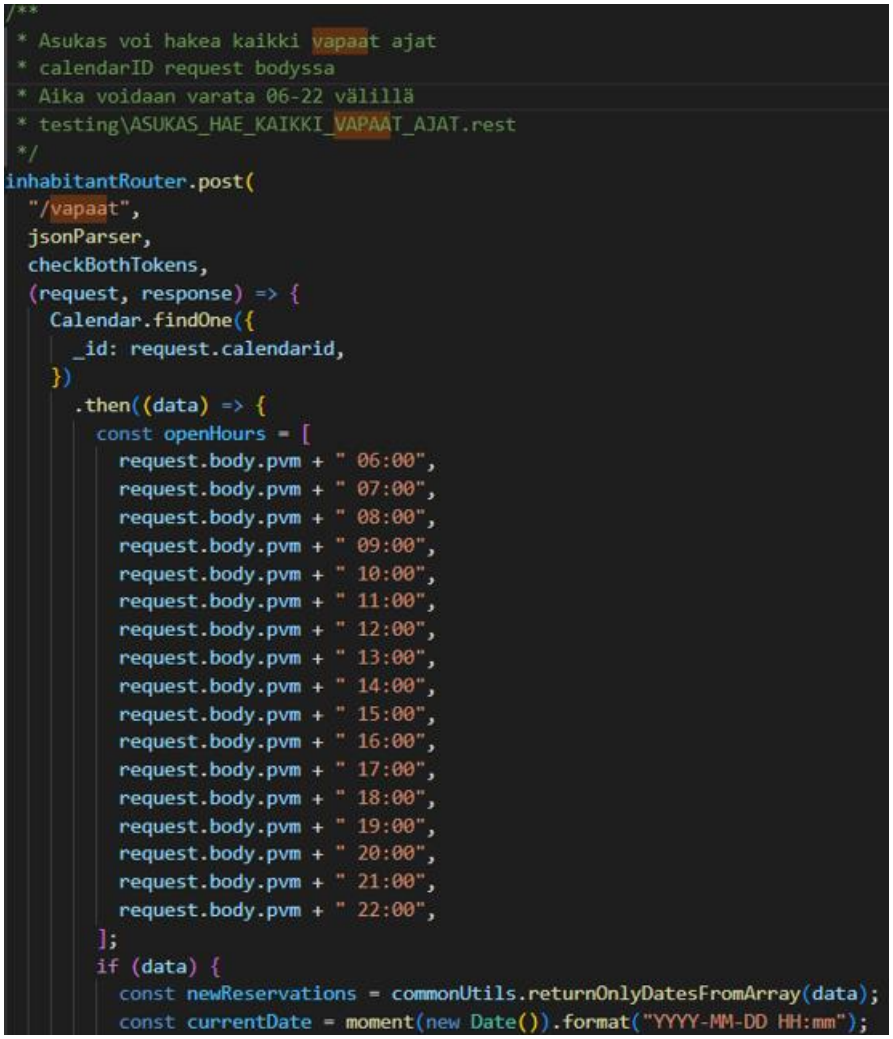
Kuvassa 22 asukas A2 on varaamassa uutta saunavuoroa ylihuomiselle. Tällä hetkellä vuoroja voi varata 06.00-22.00 väliltä, mutta tulevaisuudessa näistä voidaan tehdä taloyhtiökohtaisia niin, että isännöitsijä voi määrittää tämän aikavälin. Ylihuomisen ajoista tietyt ajat puuttuvat, koska ne on jo varattuna jollekkin muulle. Esimerkiksi Kuvassa 21 nähtiin, että asukas A2 oli itse varannut ajat 09.00 ja 14.00, joten niitä ei Kuvan 22 listasta löydy.



Kuva 22. Uuden vuoron varaus

Kuvissa 23 ja 24 on esitetty vapaiden aikojen haun logiikkaa taustajärjestelmässä. Ensin Kuvassa 23 alussa otetaan vastaan käyttöliittymältä tuleva pyyntö osoitteeseen /vapaat. CheckBothTokens middlewaressa suoritetaan pyyntöä tekevän autentikaatio tarkastus lähetetyn Json Web Tokenin avulla, kuten osassa 2.3.3 on kerrottu. Kun autentikaatio on suoritettu etsitään Calendar tyyppistä oliota Mongodb tietokannasta, jonka _id parametri vastaa lähetetystä Json Web Tokenista purettua calendarid parametria. Jos Calendar-oliota ei löydetä, päädytään Kuvan 24 catch osioon ja palautetaan kutsujalle 500 virhekoodi viestillä ”Kalenteria ei löytynyt”. Jos Calendar-olio löydetään, määritetään mahdolliset tunnit, jotka voi varata 06.00 – 22.00. Calendar-olion reservations taulukko sisältää varatut vuorot. Yksittäinen varattu vuoro on Kuvassa 25. Mahdollisten tuntien muodostuksen jälkeen palautetaan reservations taulukon soluista vain date parametri. Näin saadaan Kuvan 23 muuttujaan newReservations olemassaolevat varaukset taulukko muotoon. Tämän jälkeen formatoidaan nykyhetki muotoon ”YYYY-MM-DD HH:mm” moment kirjaston avulla. Tämä vastaa samaa muotoa kuin Kuvan 25 reservations taulukon solun date parametri. Kuvassa 24 selvitetään heti kaikki vapaat vuorot

perustuen aiempiin toimintoihin. OpenHours taulukkoa käydään yksi aika kerrallaan läpi niin, että jos sitä arvoa ei ole newReservations taulukossa, joka saatiin kyseisen Calendar-olion reservations muuttujasta eikä aika ole suurempi kuin nykyhetki niin tällöin palautetaan tämä yksittäinen aika. Näin saadaan allFreeHours muuttujaan kaikki vapaat ajat. Lopuksi allFreeHours taulukosta poistetaan vielä tyhjät arvot, koska jos edellä kuvattua ehtoa ei täytetty niin ei palautettu mitään ja taulukkoon jäi undefined kyseiselle solulle. Viimeisenä palautetaan 200 onnistumiskoodi käyttäjälle, jonka mukana message parametri ”Vapaat ajat löytyivät” ja data parametri, joka sisältää vapaat ajat. Kun nämä vapaat ajat saadaan käyttöliittymälle niin ne tallennetaan redux tilanhallintaan, josta niitä voidaan käyttää (Kuva 26).



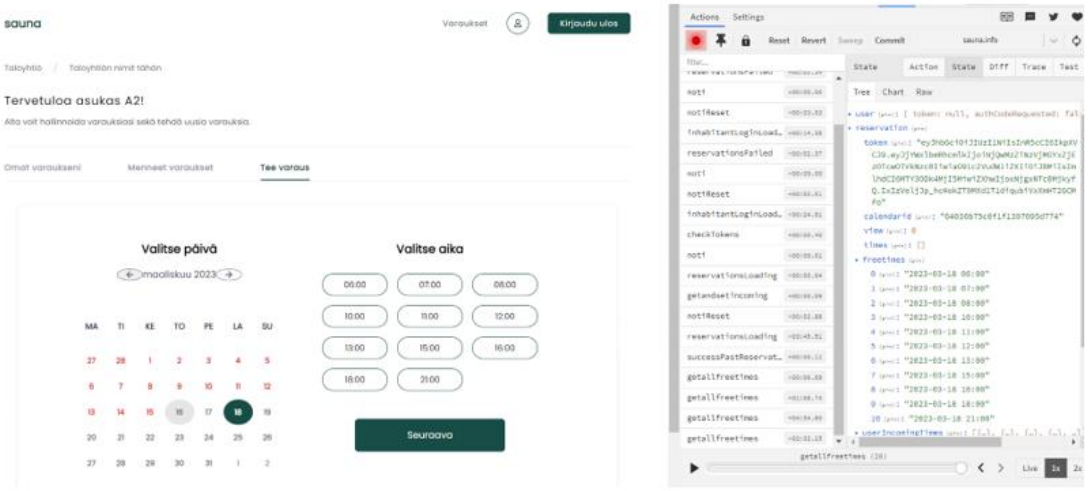
```
/**
 * Asukas voi hakea kaikki vapaat ajat
 * calendarID request bodyssa
 * Aika voidaan varata 06-22 välillä
 * testing\ASUKAS_HAE_KAIKKI_VAPAAT_AJAT.rest
 */
inhabitantRouter.post(
  "/vapaat",
  jsonParser,
  checkBothTokens,
  (request, response) => {
    Calendar.findOne({
      _id: request.calendarid,
    })
      .then((data) => {
        const openHours = [
          request.body.pvm + " 06:00",
          request.body.pvm + " 07:00",
          request.body.pvm + " 08:00",
          request.body.pvm + " 09:00",
          request.body.pvm + " 10:00",
          request.body.pvm + " 11:00",
          request.body.pvm + " 12:00",
          request.body.pvm + " 13:00",
          request.body.pvm + " 14:00",
          request.body.pvm + " 15:00",
          request.body.pvm + " 16:00",
          request.body.pvm + " 17:00",
          request.body.pvm + " 18:00",
          request.body.pvm + " 19:00",
          request.body.pvm + " 20:00",
          request.body.pvm + " 21:00",
          request.body.pvm + " 22:00",
        ];
        if (data) {
          const newReservations = commonUtils.returnOnlyDatesFromArray(data);
          const currentDate = moment(new Date()).format("YYYY-MM-DD HH:mm");
```

Kuva 23. Asukas hakee vapaita vuoroja osa 1

```
        const allFreeHours = openHours.map((hour) => {
          if (!newReservations.includes(hour) && hour > currentDate) {
            return hour;
          }
        });
        const allFreeHoursToReturn =
          commonUtils.removeNullsFromArray(allFreeHours);
        response.status(200).json({
          message: "Vapaat ajat löytyivät",
          data: allFreeHoursToReturn,
        });
      } else {
        response.status(500).json({
          message: "Kalenteria ei löytynyt",
        });
      }
    })
    .catch((e) => {
      response.status(500).json({
        message: "Kalenteria ei löytynyt",
        data: e.message,
      });
    });
  }
);
```

Kuva 24. Asukas hakee vapaita vuoroja osa 2

Kuva 25. Reservations taulukon solu



Kuva 26. Vapaat päivät listattuna freetimes muuttujaan

4 SaaS-tuotteen jatkuvat kulut

SaaS-tuotteen elinkaaressa on ominaista korkeat kehityskulut elinkaaren alussa, koska tuotetta tehdään useamman kehittäjän toimesta. Kehityskulujen suhde liikevaihtoon on usein negatiivinen, koska vasta kehitteillä oleva tuote ei ole käytössä asiakkailla. SaaS-tuotteen jatkuviin kuluihin kuuluvat palvelua kehittävien kehittäjien palkat ja ohjelmistot. Lisäksi kehitykseen ja tuotantoon tarvittavat ympäristöt, tietokannat ja muut sekalaiset kulut kasvattavat kuluja. Kaur ja muut (2014, s. 739) esittelevät neljä SaaS-palvelutarjoama vaihtoehtoa, miten SaaS-tuotteen tarjoaja voisi tarjota palvelua asiakkaille: Basic SaaS, Standard SaaS, Optimized SaaS ja Integrated SaaS. Ne määrittelevät mitä toimintoja SaaS-palvelu voisi tarjota. Basic SaaS-palvelutarjoama tarjoaa asiakkaille vain tyypilliset perustoiminnot, kuten palvelussa on kaikki käyttäjät käyttämässä samaa palvelu instanssia, palvelu ei ole konfiguroitavissa asiakkaan toiveiden mukaan sekä käyttäjäntietoa voidaan tallentaa sille luotuun tietokantaan. Standard SaaS-palvelutarjoamassa erona edelliseen on, että asiakas voi muokata palvelun käyttöliittymää sekä palvelun tallentamia tietoja tietyin ehdoin haluamallaan tavalla. Lisäksi Standard SaaS-palvelutarjoamassa Quality of Service-parametrien avulla kuvataan palveluntarjoaman käytettävyyttä, pyyntöjen vastausaikaa, suorituskykyä ja laajennettavuutta. Optimized SaaS-palveluntarjoamassa edellisen lisäksi, palvelun toimintalogiikkaa voidaan muuntaa. Tämä tarkoittaa sitä, että käyttäjille annetaan palvelun lisäksi mahdollisuus muuntaa oman palveluinstanssinsa lähdekoodia haluamallaan tavalla. Tämän avulla asiakas voi tehdä esimerkiksi integraatioita muihin järjestelmiinsä. Integrated SaaS-palvelutarjoamassa palvelu voidaan ottaa käyttöön omaan sisäverkkoon, konfiguroida palvelua miten vain omaan käyttöön sekä muuntaa mitä tahansa osaa palvelun lähdekoodista. Mitä enemmän SaaS-palvelutarjoaja tarjoaa toimintoja asiakkailleen, sitä enemmän tuotteesta voi pyytää käyttömaksua.

Saunaäpp on tähän asti kehitetty vain muutaman kehittäjän toimesta, jotka ovat palvelun idean keksineet. Tämän takia kehityksessä ei ole ollut toistaiseksi henkilöstökuluja. Sovelluskehitykseen on käytetty paljon ilmaisia avoimen lähdekoodin työkaluja. Esimerkiksi koodinkirjoitukseen Visual Studio Codea, käyttöliittymän

kehitykseen Reactia sekä taustajärjestelmän kehitykseen Nodejs:n Express viitekehystä. Muutkin työkalut, kuten Mongodb ja Redis ovat tietyntyyppisessä käytössä täysin ilmaisia. Edellä mainittujen asioiden takia, voidaan sanoa, että tuotteen kehityksessä ei ole toistaiseksi ollut kuluja.

Kun Saunaäpp siirtyy tuotantoon ja aloitetaan asiakashankinta, kuluja myös syntyy. Saunaäpp tulee julkaista palvelimelle, jonka hallinnointi maksaa tietyn summan kuussa. Erilaisia vaihtoehtoja ovat esimerkiksi julkaiseminen Microsoftin Azureen tai johonkin esimerkiksi suomalaiseen webhotelliin. Oikeastaan ei ole väliä mihin tuote julkaistaan, kuukausittainen summa palvelimen hallinnoinnista pyörii sen verran alhaisissa summissa nykypäivänä. Azuren (2023) mukaan tuotantoa vastaavan ympäristön kuukausittainen maksu olisi 100 dollarin luokkaa.

Tuotteen ollessa tuotannossa ja sillä on oikeita maksavia asiakkaita, tietyntyyppiset ongelmaselvitykset ja jatkokehitykset asiakkaiden toiveesta astuvat kuvaan. Kehittäjäporukan kanssa on tästä selkeä suunnitelma, että alkuun asiakkaiden ongelmaselvitykset sekä jatkokehitykset tullaan hoitamaan samalla porukalla kuin koko tuotteen kehitys. Jos tuote saa enenemissä määrin asiakkaita tulevaisuudessa, voidaan ulkopuolista rekrytointia harkita. Tämän rekrytoitavan henkilön työnkuva koostuisi samoista ongelmaselvityksistä sekä tuotteen jatkokehityksestä. Pitkän aikavälin tavoitteena olisi kasvattaa kuukausittain laskutettavien asiakkaiden määrää niin paljon, että ne kustantavat vähintään koko toiminnan jatkuvat kulut.

5 Liiketoimintamallin luonti

Laudon ja Traver (2021, s. 94) mukaan liiketoimintamallin tulisi sisältää suunnitelmia ja strategisia valintoja, joiden avulla yritys tavoittelee voiton tekemistä jollain markkinalla. Liiketoimintamallin avulla yritys pystyy tavoittelemaan pidemmän ajan menestystä kilpailijoihinsa nähden.

5.1 Lean startup

Kun tuodaan täysin uutta tuotetta markkinoille, on hyvä olla jatkuvasti hereillä sen suhteen, onko tuotteelle oikeasti tarvetta, kysyntää ja mielenkiintoa. Lean startup työskentelytyyli sopii hyvin Saunaäpin kehitykselle. Lean startupissa pyritään kehittämään jatkuvasti arvioiden tuotetta, asiakkaita ja asiakkaiden kokemuksia niin että saadaan mahdollisimman nopeasti luotua oikeasti asiakkaalle arvoa tuovia toimintoja valmiiksi. Tämän avulla huomataan mahdollisimman nopeasti, jos tuotteen kehitys menee asiakkaalle arvoa tuottamattomaan suuntaan ja suuntaa voidaan heti korjata, jonka avulla kehityskulut pysyvät mahdollisimman pieninä. Ajatus onkin luoda mahdollisimman hyvä tuote, mahdollisimman pienillä kuluilla. (Isoherranen & Ratnayake, 2019, s. 736)

5.2 SaaS liiketoimintamalli

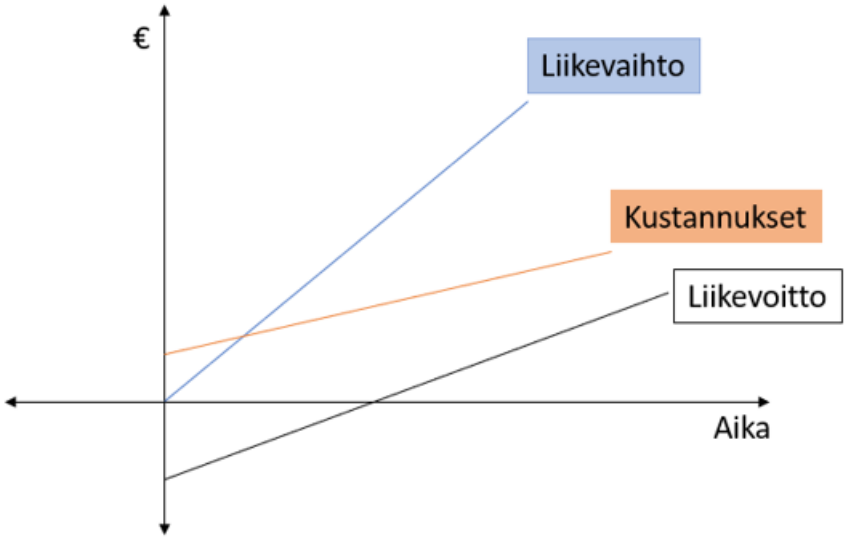
SaaS liiketoimintamalli on uudenlainen liiketoimintamalli, jossa luodaan tietynlainen sovellus tai järjestelmä, jonka käyttöä myydään erilaisin lisenssein. SaaS liiketoimintamallille on usein tyypillistä suuret kustannukset suhteessa tuloihin tuotteen alkukaaressa. Kun tuote on kehitetty markkinoille ja sillä on asiakkaita, kustannusten suhde tuloihin alkaa tasaantumaan. Erittäin voitollista SaaS liiketoimintamallin tuotteissa on se, että uusasiakashankinta ei nosta kustannuksia samassa suhteessa kuin perinteisimmissä liiketoimintamalleissa. Ilman suuria lisäpanostuksia tuotteeseen,

kustannukset pysyvät lähes samana jatkuvasti. (Xiong ja muut, 2010, s. 1) SaaS-tuotteelle on ominaista, että se voidaan julkaista mihin tahansa palvelimelle ja sitä voidaan asiakkaiden toimesta vuokrata riippuen hinnoittelusopimuksesta, esimerkiksi palvelutyyppi, käyttöaika ja käyttömäärä voivat olla hinnoittelusopimuksen osia (He, 2010, s. 232).

SaaS–tuotteen käyttö on helppoa ja vaivatonta, koska usein vain internet yhteys riittää sen käyttöön ja sitä voidaan käyttää missä ja milloin vain. SaaS–tuotteella ja sen myynnillä on useita yhteisiä ominaispiirteitä: SaaS-palvelutarjoamasta riippuen palvelun käyttäjät voivat jakaa sovelluksen käytön muiden käyttäjien kanssa tai käyttäjillä voi olla oma palveluinstanssinsa esimerkiksi oman sisäverkon sisällä. SaaS–tuotteen käytöstä usein laskutetaan käytön mukaan, ajan mukaan, käyttöoikeuksien määrän mukaan tai edellä mainitut yhdessä muodostavat loppuhinnan. SaaS-tuotteessa voi olla lisälaajennuksia, joita voi ostaa lisähintaan tai olla ostamatta. Tämän avulla asiakkaat voivat muokata tuotetta sen mukaan mitä he tarvitsevat ja maksavat vain siitä mitä palvelunosaa käyttävät. (Wang ja muut, 2012, s. 574)

Saunaäpp on hyvä esimerkki tulevasta SaaS-liiketoimintamallin tuotteesta. Asiakkaille tarjotaan käyttöoikeuksia tuotteeseen tietynkokoista hintaa vastaan. Esimerkiksi, jos asiakas on taloyhtiö niin taloyhtiön jokaiselle asukkaalle luodaan henkilökohtaiseen käyttöoikeuteen soveltuvat käyttötunnukset, joiden avulla taloyhtiön asukas voi käyttää palvelua. Näiden lisäksi, taloyhtiön tietylle asukkaalle tai esimerkiksi isännöitsijälle luodaan korkeamman oikeuden käyttötunnukset, joiden avulla voi tehdä enemmän toimintoja taloyhtiön saunavarauskalenterissa. SaaS-tuotetta tarjotaan toistaiseksi Kappaleessa 4 kuvatulla Basic SaaS-palvelutarjoaman mukaisella tavalla niin, ettei käyttäjät pysty muokkaamaan sovellusta yhtään. Lisäksi kaikki käyttäjät toimivat saman palveluinstanssin sisällä julkiverkossa. Tämä tarkoittaa sitä, että esimerkiksi taloyhtiö A Joensuusta käyttää samaa palvelua julkiverkossa kuin taloyhtiö B Turusta. Molempien taloyhtiöiden tiedot ovat tallennettu samaan tietokantaan.

Ilman ainuttakaan asiakasta, kustannusten suhde tuloihin on suuri. Tosin, kun asiakkaita aletaan saamaan uusasiakashankinnan avulla, kustannusten suhde tuloihin alkaa tasoittumaan. Kuten Kuvasta 27 nähdään, tietyssä pisteessä päästään kohtaan, jossa kustannukset ovat yhtä suuret kuin tulot. Tämän jälkeen suurin osa uusasiakkaiden maksuista päätyy suoraan liikevoitoksi, koska palvelun tarjoaminen uusille asiakkaille nostaa kustannuksia hyvin hitaasti. Kuvankaltaisen tilanteen edellytys on uusmyynnin onnistuminen sekä tuotteen tietoisuuden kasvattaminen taloyhtiöiden keskuudessa.



Kuva 27. Kustannusten suhde liikevoittoon SaaS liiketoimintamallissa

5.3 Kohdemarkkinat

Business-to-Business (B2B) on liiketoimintamalli, jossa yritykset pyrkivät myymään tuotteitaan toisille yrityksille (Laudon ja Traver 2021, s. 119). Saunaäpp pyrkii yrityksenä myymään Saunaäpp järjestelmäänsä taloyhtiöille, joten Saunaäpp toimii B2B-markkinassa.

Saunaäpp on tarkoitettu alustavasti saunavuorojen varaamiseen esimerkiksi taloyhtiöille ja muille järjestöille. Oikeastaan kaikki, joilla on yleinen sauna usean eri henkilön

käytössä ovat potentiaalisia asiakkaita palvelulle. Palvelua voidaan markkinoida myös suoraan isännöitsijä yrityksille. Isännöitsijät myyvät omien palveluidensa kautta tuotteitaan taloyhtiöille, joten tämä voisi olla esimerkiksi useamman isännöitsijäyhtiön tuote, jota myydä suoraan taloyhtiöille. Tähän malliin olisi hyvä rakentaa oma hinnoittelumekanismi, jossa isännöitsijöillä olisi vahva halu myydä tuotetta isännöitäville taloyhtiöille.

Uutena toimijana markkinoille tullessa olisi tyhmää lähteä rajaamaan oikeastaan minkäänlaisia potentiaalisia asiakkaita ulkopuolelle. Alkuun kaikki asiakkaat tuovat kuukausittaisten käyttömaksujen lisäksi arvokasta palautetta palvelusta ja sen käytöstä. Näiden palautteiden pohjalta saadaan eri näkökulmasta kehitettyä nopeasti palvelua suuntaan, joka tuo suurimman mahdollisen lisäarvon asiakkaalle.

Tulevaisuudessa palvelun mahdollinen laajennettavuus muille aloille on helppoa. Käyttöliittymän pelkkä tekstienvaihto, mahdollistaa tietyissä raameissa jo järjestelmän käytön esimerkiksi pesutupien ja liikuntatilojen varauksiin. Tämä luo aivan uuden mahdollisuuden tarjota palvelua paljon suuremmalle asiakaskunnalle käyttöön.

Enenemissä määrin esimerkiksi taloyhtiöillä alkaa olemaan taloyhtiön aulassa yleisen ilmoitustaulun sijasta suuri näyttö, josta näkee taloyhtiön asioita. Saunaäpissä voitaisiin luoda tietyn taloyhtiön asukaskäyttäjän ja adminkäyttäjän lisäksi rooli yleiselle käyttäjälle. Tämä yleinen käyttäjä sisältäisi mahdollisuuden tehdä vain hakutyyppisiä operaatioita. Olemassa olevaa tietoa voidaan vain hakea, mutta uutta tietoa ei voi luoda. Esimerkiksi olisi mahdollista hakea tietoa tämänhetkisistä varaustilanteista. Tämän avulla voitaisiin näyttää asukkaille saunatilojen tämänhetkinen varaustilanne yleisessä näytössä. Käyttäjärooleista Saunaäpissä lisää luvuissa 2.4 ja 2.5.

5.4 SaaS-tuotteen hinta-analyysi

Yrityksen ansaintalogiikan suunnittelulla pyritään selvittämään miten yhtiö tuottaa liikevaihtoa. Ansaintalogiikoita on useampia perustuen myytyyn tuotteeseen. Saunaäpin ansaintalogiikka on tilauksiin perustuva ansaintalogiikka (subscription revenue model). Tilauksiin perustuvassa ansaintalogiikassa on ominaista se, että yritys tarjoaa palveluja, joiden käytöstä se perii tietynkokoista maksua. (Laudon & Traver, 2021, s. 95)

Tällä hetkellä yleisimmät tavat määrittää SaaS-tuotteelle hinta ovat kuluihin perustuva hinnoittelu, monopoli hinnoittelu, kilpailuun perustuva hinnoittelu sekä hyötyyn perustuva hinnoittelu. Kuluihin perustuvassa hinnoittelussa pyritään etsimään sopiva hinta, jota asiakas on valmis maksamaan samaan aikaan kun näkee tuotteesta saatavan hyödyn maksua suuremmaksi. Monopoli hinnoittelussa tuotteen tuottaja huomaa olevansa ainut toimija markkinoilla, joten perustuen asiakkaiden saamaan hyötyyn, laskutus voi olla mitä tahansa. Kilpailuun perustuvassa hinnoittelussa, markkina on aggressiivisesti kilpailtu samoilla tuotteilla, joilla millään ei ole suurempaa kilpailuetua toiseen verrattuna. Ainut mikä tekee tuotteen paremmaksi asiakkaan silmissä kilpailijaan verrattuna on hinta. Hyötyyn perustuvassa hinnoittelussa hinta määritetään suoraan asiakkaan tuotteesta saadun välillisten ja välittömien hyötyjen kautta. Markkinakysyntä ohjaa hyötyyn perustuvaa hinnoittelua suuresti. (Xiong ja muut, 2010, s. 1)

Saunaäpp tuote on kehitetty tuotteeksi, josta asiakkaat maksavat kuukausittaista hintaa sen käytöstä. Koska jatkuvia kuluja on hyvin vähän, asiakkaiden maksavat kuukausittaiset maksut valuvat hyvin suurella prosentilla liikevaihdosta suoraan liikevoittoon. Tämän takia kuukausittaisin laskutettavan hinnan ei tarvitse olla kovin korkea, jotta tuote olisi voitollinen. Toisaalta tuotteen omistajat haluavat maksimoida tuotteen tekevän voiton. Näiden asioiden lisäksi pitäisi pohtia markkinoilla olevien kilpailijoiden omia hintojaan ja skaalata omaa hintaa myös tämän kriteerin perusteella. Tuote ei ainakaan toistaiseksi sisällä mitään suurta toiminnallisuuseroa tai hyötyä kilpailijoihin nähden, joten korkeamman hinnan laskuttaminen ei ole mahdollista. Kun edellä mainitut asiat nivotaan yhteen, tuotteelle täytyy löytää mahdollisimman optimaalinen kuukausittain

laskutettava hinta, joka houkuttelee uusia asiakkaita halvan hinnan takia, maksimoi samalla kuitenkin tuotteen voitollisuuden ja mahdollisesti houkuttelee kilpailijoiden tuotteita käyttäviä vaihtamaan halvempaan ratkaisuun.

Edellä mainittujen asioiden lisäksi pitäisi tarkastella erilaisia asiakasryhmiä. Suuri määrä huoneistoja taloyhtiössä nostaa tehtävien pyyntöjen määrää Saunaäpin taustajärjestelmälle, joka taas nostaa tarvittavan muistin määrää tietokannassa. Mitä laskuttaa taloyhtiöltä, jossa on 20 huoneistoa? Mitä laskuttaa taloyhtiöltä, jossa on 200 huoneistoa? Lisäksi erilaiset sopimuksen avaustarjoukset sekä pitkäaikaisten asiakkaiden lisäalennukset tulee huomioida. Uutena toimijana markkinoilla pitäisi ensin miettiä millaisella hinnoittelustrategialla lähdetään tuotetta myymään. Tuntemattomana palveluntarjoajana voisi olla helpointa kilpailla hinnan avulla. Kappaleessa 1.2 Kilpailijat on todettu mitä SaunaOnline laskuttaa asiakkailta. Nämä hinnat voisi toimia hyvänä pohjana hintojen päättämiselle, kun muuta näkyvyyttä hintoihin ei ole. Kappaleessa 4 pohdittiin hinnanmuodostumista SaaS-palvelutarjoaman kautta: mitä enemmän palvelu tarjoaa lisätoimintoja asiakkaille, sitä korkeampi hinnan tulisi olla. Toistaiseksi palvelu on toteutettu niin, että asiakkaille tarjottaisiin Kappaleessa 4 kuvattua Basic SaaS tyyppistä palvelutarjoamaa. Olisi tärkeää kirjata jatkuvalaskutteisiin SaaS sopimuksiin myös johonkin indeksiin sidotut hintojenkorotukset suojaksi korkeaa inflaatiota vastaan, mutta samalla tehdä jatkuvaa vertailua asiakkaan kokonaiskuluista tuotteen käytöstä verrattuna muihin vastaaviin palveluihin.

6 SaaS-tuotteen markkinointisuunnitelma

Markkinointisuunnitelman luonnissa määritetään markkinoinnin tehtävät ja sen tavoitteet yrityksen tulevaisuuden kannalta. Markkinoinnin edistymistä on hyvä seurata erilaisin mittarein, joiden avulla voidaan määrittää, kuinka onnistunutta yrityksen tekemä markkinointi on. (Cohen, 2006, s.1)

6.1 Lähtökohta-analyysi

Lähtökohta-analyysissä pyritään selvittämään yrityksen nykytilaa koko yrityksen toimintakentässä, kuten verrattuna kilpailijoihin, asiakkaisiin ja muihin yrityksiin. Missä tilanteessa yritys on tällä hetkellä? Yrityksen nykytilaa tulisi tutkia siltä pohjalta, että pyritään tekemään yrityksen tulevaisuuden menestymisestä ja tulevaisuuden näkymistä mahdollisimman hyviä. (Rope, 2005, s. 464-469)

SWOT-analyysi on yrityksille hyvin sopiva nykytilanteen analysointi työkalu, jossa kerrotaan ja tiedostetaan kyseisellä hetkellä yrityksen toimintaedellytykset ja tämänhetkinen tila neljästä eri kulmasta: vahvuudet, heikkoudet, mahdollisuudet sekä tulevaisuuden uhat (Lipiäinen, 2000, s. 47).

Vahvuudet	Heikkoudet
- Läpinäkyvä toimintamalli - Ketterä toimija - Kilpailukykyinen hinnoittelu	- Uusi toimija: tuntemattomuus - Osaamisen painottuneisuus
Mahdollisuudet	**Uhat**
- Suuri asiakasmäärä - Toimia kapealla markkinalla, johon ei moni isompi toimija lähde mukaan	- Tuotteen saaminen julkaisukuntoon - Dokumentaation vähäisyys - Bugit

Kuva 28. Saunaäpin SWOT-analyysi

Saunaäpin yksi vahvuuksista on läpinäkyvä toimintamalli. Tulemme kertomaan selkeästi asiakkaille kaikista kuluista, joita tulee Saunaäpin käyttöönotosta sekä miten toimintamme on rakennettu. Palvelu ei tule sisältämään piilokuluja ja hinnoittelu tulee olemaan kilpailukykyinen muihin toimijoihin verrattuna. Pienenä yrityksenä Saunaäpp pystyy olemaan lisäksi ketterä toimija, joka pystyy sopeutumaan alan markkinatilanteeseen eri markkinaympäristöissä.

Saunaäpp on täysin tuntematon ratkaisu taloyhtiöiden ja järjestöjen keskuudessa. Tämän takia tuotteen mainostuksen aloituksessa tulisi painottaa tuotteen hyötyjen lisäksi myös kokonaisvaltaisempaa markkinointia palvelun taustalla olevasta yhtiöstä. Asiakkailla voi herätä kysymyksiä esimerkiksi, että kuka, miten ja milloin palvelu on tehty. Nämä kysymykset juontavat juurensa siitä, onko palvelun tekijä luotettava taho ja onko palvelu tehty esimerkiksi riittävän tietoturvallisesti. Osaaminen on lisäksi painottunut vain muutamille henkilöille mikä aiheuttaa sen, että riski tiedon häviämiseen on korkea. Tämän takia dokumentaatiota Saunaäpin ratkaisuista tulisi tehdä ja ylläpitää jatkuvasti.

Saunaäpp tarjoaa SaaS-palveluna saunavuorojen varausta. Digitalisaatio murros on ollut käynnissä jo useita vuosia, joten markkinaympäristö voi olla otollinen tällaiselle palvelulle. Tällä hetkellä suuri osa taloyhtiöistä hoitaa saunatilojensa varauksen hallinnan edelleen vanhanaikaisesti tulostetulla listalla, johon merkataan saunavuorot kynällä. Tämä on pitkässä juoksussa erittäin aikaa vievää, kun uusia listoja tulee tulostaa ja jakaa tietyin väliajoin esimerkiksi isännöitsijän toimesta. Lisäksi omien vuorojen muuttaminen, peruuttaminen, laskuttaminen ja laskujen kirjanpito on vaivalloista. Saunaäpp tarjoaa ratkaisun tähän ongelmaan tarjoamalla taloyhtiökohtaista saunavuorojen hallinnointi järjestelmää, jossa jokainen asukas ja taloyhtiön varausten hallinnoija saavat kirjautumistunnukset palveluun. Näiden avulla taloyhtiön asukkaat voivat varata, muokata ja poistaa omia saunavuorojaan sekä varausten hallinnoijat voivat hallinnoida kyseisen taloyhtiön varauksia korkeammilla käyttöoikeuksilla. Vastaavia tuotteita löytyy niukasti markkinoilta, mutta esimerkiksi edellä kuvattu SaunaOnline tarjoaa vastaavanlaista palvelua taloyhtiöille ja muille järjestöille. Markkinat ovat suuret taloyhtiöiden suuresta määrästä ja kilpailijoiden vähäisestä määrästä johtuen, joten markkinaa tuotteelle näyttäisi olevan. Tärkeintä olisi pystyä rakentamaan tuote, joka oikeasti tuo lisäarvoa asiakkaille niin että se on helpompi, halvempi ja helppokäyttöisempi kuin vanha tapa hallinnoida saunavuoroja.

Saunaäpin saaminen julkaisukuntoon, dokumentaation vähäisyys sekä bugit ovat Saunaäpin suurimmat uhat. Tuotteen valmistumisen hetki on vaikea määrittää, koska aina voi tulla mieleen uusia ja uusia asioita mitä voisi lisätä ja korjata. Tämän takia olisi hyvä määrittää missä kohtaa tuote on tarpeeksi valmis, jotta se voidaan julkaista tuotantoon. Dokumentaation vähäisyys aiheuttaa sen, että tieto ja osaaminen ei ole tällä hetkellä säilyneenä Saunaäpin kehitysprosesseissa vaan se on palvelun kehittäjien muistissa. Dokumentaatiota on pystyttävä kirjaamaan enemmän. Tulevaisuuden ja nykyiset bugit aiheuttavat ongelmia käyttäjille. Suurien bugien tuleminen Saunaäppiin voi aiheuttaa epäluottamusta asiakkaiden silmissä. Hyvällä automaattitestauksella pystytään minimoimaan bugien määrää. Aina kun tulee uusi bugi, täytyy sen korjauksen yhteydessä tehdä siitä myös testi. Tämä estää sen, että samaa bugia ei tule palveluun.

6.2 Markkinointistrategia

Markkinointi on yleisesti jonkun yrityksen tuotteen, palvelun tai muun myytävän esineen esilletuomista eri kanavissa, joka tähtää markkinoitavan tuotteen kiinnostavuuden nostamiseen kuluttajien keskuudessa. Markkinoinnin päätavoitteena on tietyn yrityksen tuotteen tai palvelun myynnin kasvattaminen, tietoisuuden lisääminen sekä uusien asiakkaiden saanti. (Virtanen, 2010, s. 15-16) Yrityksen päätökset, ulostulot, haastattelukommentit eri medioihin sekä esimerkiksi yrityksen ja työntekijöiden sosiaalisen median päivitykset luovat mielikuvan yrityksestä asiakkaiden silmissä. Tämän takia markkinointi on nykyisin koko yrityksen työntekijöiden tehtävä eikä vain niin sanotun markkinointiosaston. (Bergström & Leppänen 2015, s. 18)

Markkinoinnin päätehtävänä on saada Saunaäpp tunnetuksi taloyhtiöiden keskuudessa. Moni taloyhtiö edelleen hoitaa saunatilavaraustenhoidon vanhanaikaisesti kynällä ja paperilla. Markkinoinnissa tulee korostaa digitalisaation hyödyistä tässä asiassa. Esimerkiksi kulujen säästö sekä helppokäyttöisyys, kun varaustilanteen voi tarkistaa, tehdä uuden varauksen, poistaa varauksen tai muuttaa olemassa olevaa varausta missä vain ja milloin vain ilman mitään välikäsiä matkalla. Lisäksi päästään pois paperienpyörittelystä varaustilanteista, kun kaikki on digitaalisesti. Tämä poistaa väärinkäsityksien syntymisen, että kenen saunavuoro tällä hetkellä onkaan menossa. Saunaäppiin tarjotaan myös varaustilanteen hallinnoijalle oma näkymänsä, joka pääsee helposti käsiksi nykyvaraustilanteeseen taloyhtiössä. Varaustilanteen hallinnoija pystyy optimoimaan helposti omasta näkymästään saunanlämmitystä, ruuhka-aikoja ja muita saunaan liittyviä toimenpiteitä, kun hänellä on jatkuvasti ajankohtainen näkymä saunavuoroista sovelluksessaan. Sovelluksen avulla voidaan pitää esimerkiksi kirjaa suosituimmista sauna-ajoista ja sen avulla taloyhtiö voi päättää, että sauna on päällä vain tiettyinä päivinä ja tiettyinä kellonaikoina, jonka avulla minimoidaan turha lämmitys ja pienennetään energialaskua.

Markkinoinnin tavoitteena on luoda Saunaäpille mahdollisimman hyvät menestymisen mahdollisuudet kilpaillussa markkinassa. Tunnettuvuuden lisääminen, hyötyjen korostaminen ja asiakkaiden mielenkiinnon herättäminen edes yhteydenottoon asti ovat markkinoinnin tavoitteiden keskiössä. Jo pelkkään taloyhtiön yhteydenottoon asti pääseminen on loistava saavutus, koska se tarkoittaa että markkinointi on saavuttanut taloyhtiön, jonka mielenkiinto on herännyt markkinoitujen hyötyjen takia. Vaikka yhteydenotto ei etenisi loppujen lopuksi käyttösopimukseen asti niin tämän asiakkaan avulla on mahdollisesti mahdollista kerätä palautetta, että mikä vika palvelussa on. Jatkuvan palautteen kerääminen on tärkeää, jotta palvelua voidaan kehittää mahdollisimman helposti mahdollisimman asiakasta hyödyttäväksi tuotteeksi Lean startup toimintamallin mukaisesti. Jatkuvan palautteen keräämisen, siitä oppimisen ja kehitykseen asti vietävien palautteesta saatavien kehitysideoiden lopputuloksena saadaan kehitettyä tuote, joka jatkuvasti parantuu asiakkaiden silmissä. Tämä luo luottamusta asiakkaalle, että tuote voi parantua heidän toiveiden mukaan ajansaatossa ja täten varmistaa heille lisäarvoa tuovien toiminnallisuuksien rakentumisen.

6.3 Markkinointiviestintä

Markkinointiviestintä on yrityksestä ulospäin tapahtuvaa viestintää. Viestinnän avulla yritys pyrkii kertomaan asiakkaille tuotteistaan, palveluistaan ja muista tarjoamistaan. Tämän viestinnän tavoitteena on herättää kiinnostusta asiakkaissa yrityksen tarjoamia palveluita kohtaan. Viestintäkeinoja voivat olla esimerkiksi suoramainonta, henkilökohtainen myyntityö, menekinedistäminen sekä suhde- ja tiedotustoiminta. (Rope, 2005, s. 277)

Startupissa ei ole omia työntekijöitä markkinoinnin toteuttamiseen, joten sitä tullaan tekemään kehittäjäporukan kesken kehitystyön ohessa. Tämän takia markkinoinnin tulisi olla mahdollisimman helppoa ja vaivatonta. Tehtävällä markkinoinnilla pyritään saavuttamaan mahdollisimman suuri kohdeyleisö, mahdollisimman vaivattomasti ja halvasti. Tämän takia paras markkinointikanava tämänlaiseen markkinointiin on

mielestäni sosiaalinen media. Hyvä tapa olisi esimerkiksi kerätä aineistoa nykytavan tuomista vaivoista ja kuluista hallinnoida ja varata saunavuoroja taloyhtiöissä ja verrata sitä Saunaäpin ratkaisuun hoitaa sama asia digitaalisesti. Vuonna 2023 lähes kaikki taloyhtiöiden maksut ovat nousseet rajusti edellisistä vuosista, joten monet taloyhtiöt etsivät säästökohteita eri paikoista. Tämän takia tärkein painotettava asia mainonnassa olisi kulusäästöjen osuus. Tulisi esittää selkeästi, miten Saunaäpp tulee säästämään taloyhtiöiden kuluja saunavarausten hallinnoinnissa, kun kaikki saunavarauksiin liittyvät asiat hoidetaan sovelluksen kautta, kaikki manuaaliset paperi - ja hallinnointikulut vähenevät. Tärkeää olisi keskittyä mainostamaan myös kuinka helppoa saunavuorojen hallinta, varaus ja muut toiminnot ovat, kun kaikki toimii internetissä toimivassa sovelluksessa. Saunavuorolistoja ei tarvitse mennä tarkastamaan taloyhtiön saunan ovelle, vuoron voi peruuttaa, muuttaa sekä varata helposti. Enää ei tarvitsisi esimerkiksi lähettää isännöitsijälle viestiä oman vakituisen saunavuoronsa peruuttamisesta, joka hoitaa manuaalisesti peruuttamisen jossain toisessa järjestelmässä, jotta laskutus loppuu. Lisäksi yksittäisten vuorojen peruuttaminen helpottuu, jos esimerkiksi sinulla on vakiovuoro keskiviikkoisin kellonaikana 19:00, niin jos Saunaäpp olisi käytössä, pystyt peruuttamaan ne vuorot joihin et pääse ja näin et maksa käyttämättömästä vuorosta. Jos sauna on tuolloin päällä niin joku muu voi varata tuon vuoron ja näin ei lämmitetä saunaa turhaan.

Saatuja tuloksia tulisi saada tiivistettyä mahdollisimman lyhyeen ja helposti ymmärrettävään muotoon, jotta niiden esittäminen kohdeyleisölle olisi mahdollisimman vaikuttavaa. Tärkeimpiä esitettäviä tuloksia ovat kulusäästöihin ja helppokäyttöisyyteen liittyvät tulokset. Tuloksia voisi esittää esimerkiksi Saunaäpin virallisilla sosiaalisen median sivuilla. Saunaäpp voisi virallisten sosiaalisen median tunnuksillaan osallistua rohkeasti erilaisiin palvelun hyötyjä korostaviin keskusteluihin. Esimerkiksi viestipalvelu X:ssä (ent. Twitter) on nykyisin paljon keskustelua taloyhtiöiden nousevista kustannuksista.

7 Yhteenveto

Teknologinen murros ja internetin kehittyminen on tehnyt sen, että nykyään enenemissä määrin kaikkia järjestelmiä pyritään siirtämään toimimaan internettiin. Esimerkiksi lähes kaikki varausjärjestelmät (hotellit, taksit ja autonvuokraus) on siirretty internettiin. Tämä helpottaa vuokrausyrityksen varaustenhallinnointia, markkinointia ja myyntiä sekä vuokraajan toimintaa varatessa haluamansa hotellin, taksin tai auton varaussovelluksen kautta. Tässä diplomityössä luotiin alusta asti varausjärjestelmä saunavuorojen varaukseen nimeltä Saunaäpp.

Digituotteen suunnittelussa, teossa ja markkinoille viennissä tulee ottaa huomioon useita asioita. Suunnittelussa tulisi ottaa huomioon muun muassa asioita, jotka helpottavat tuotteen valmistamista aina alkumetreiltä tuotantoon viemiseen ja tuotannossa oloon asti. Suunnittelun avulla pyritään miettimään, esimerkiksi miten tuote saadaan kohdeasiakkaiden tietoisuuteen, miten sitä markkinoidaan heille ja mitä siitä voisi laskuttaa. Nykyään korkean inflaation aikaan taloyhtiöt ovat joutuneet nostamaan omia omistajilta pyytämiään vastikkeitaan enemmän kuin pitkään aikaan. Tämä aiheuttaa sen, että taloyhtiöt ovat varmasti haluttomampia kuin ennen ottaa käyttöön palvelua, joka ainakin alkuun tuo lisäkuluja taloyhtiön toimintaan ja voi johtaa vastikkeiden lisänostoon. Tämän takia pitäisi palvelun käytölle keksiä sellainen hinta, jonka taloyhtiöt ovat valmiita maksamaan. Taloyhtiöiden tilanteet ovat tietenkin erilaisia, mutta hinnan tulisi olla sellainen, että keskiverto taloustilanteessa oleva taloyhtiö pystyy tuotteen ottamaan käyttöön.

Edellä mainittujen asioiden lisäksi olisi hyvä pohtia myös, löytyykö markkinoilta jo vastaavia tuotteita ja onko tuotteella kilpailijoita. Tämän lisäksi palvelun teknisessä suunnittelussa olisi hyvä pohtia millaisia teknologioita voitaisiin käyttää, jotta tuote toimisi mahdollisimman hyvin, olisi mahdollisimman helppokäyttöinen ja kustannukset olisivat siedettävät. Saunaäpissä pyritään käyttämään mahdollisimman paljon avoimeen lähdekoodin perustuvia työkaluja, jotta kehitystyö on läpinäkyvää sekä halpaa. Käyttöliittymä on tehty Reactilla, taustajärjestelmä Nodejs:llä sekä tietokantana

käytetään MongoDB:tä. Lisäksi käytössä on muutamia muita teknologioita eri käyttötarkoituksiin. Mitä enemmän asioita pohtii, suunnittelee ja on jatkuvassa vuorovaikuttamisessa asiakkaiden ja palautteiden kanssa, sitä nopeammin pystytään kehittämään asiakkaille oikeasti merkittäviä toiminnallisuuksia. Tämä säästää kehitysresursseja sekä nopeuttaa kilpailukykyisen tuotteen luontia.

Jatkuva tuotteen arviointi kilpailijoihin nähden hinnan, laadun sekä helppokäyttöisyyden kannalta on tärkeää, jotta palvelu säilyy asiakkaiden silmissä kilpailukykyisenä ja laadukkaana. Asiakkaiden palautteiden kuunteleminen on tärkeää, koska heiltä saa jatkuvasti kehitysideoita palveluun kehitettävistä asioista. Helppokäyttöisyyden takaamiseksi Saunaäpissä pyritään noudattamaan WCAG 2.1-standardia mahdollisimman pitkälle. Kappaleen 5.4 mukaisesti palvelun hinnoittelu tulee alkuun pohjautumaan lähelle kilpailijoiden hinnoittelupolitiikkaa, koska uutena toimijana markkinoille tullessa hinnan määrittäminen on haastavaa. Palvelun tunnettuvuuden kohottaminen toteutetaan sosiaalisen median markkinoinnin avulla, kuten Kappaleessa 6.3 on pohdittu. Markkinoinnin on oltava mahdollisimman tehokasta.

Saunaäpp ei ole vielä tuotannossa, mutta tarkoitus olisi jatkaa tästä eteenpäin ja tavoitella markkinoille menoa. Suunnitteilla olisi tehdä palvelun käytettävyystestausta eri ikäryhmille korostaen vanhempia ihmisiä. Vanhemmille ihmisille digipalveluiden käyttäminen on keskimääräistä vaikeampaa, ja he ovat usein taloyhtiöissä ahkerimpia saunojen käyttäjiä sekä aktiivisimpia päätöksentekijöitä. Palvelun täytyy olla heille soveltuva, jotta sillä olisi tulevaisuutta. Vanhemmat ihmiset toivovat, että palvelu olisi mahdollisimman helppokäyttöinen ja selkeä. Palvelussa tulisi olla samaa toiminnallisuutta tekevät painikkeet samoilla paikoillaan samannäköisinä sekä palvelussa tulisi olla helppoa navigoida eri paikkoihin ja paikat tulisi olla loogisia koko palvelun kontekstissa. Kaiken tämän miettimisen jälkeen, silti olisi olla lisää tukea käyttäjille esimerkiksi käyttöohjeiden ja käyttämiseen opastavien videoiden muodossa. On tärkeää, ettei palvelua viedä liian keskeneräisenä tuotantoon asiakkaiden arvioitavaksi, koska palvelun nimeä ei kannata pilata. Ensivaikutelman voi tehdä vain

kerran. Sovelluksessa riittää vielä avoimia kysymyksiä, joita ratkotaan vielä ennen tuotantoon vientiä.

Lähteet

Azure. (2023). Azure pricing. Noudettu 17.11.2023 osoitteesta https://azure.microsoft.com/en-us/pricing

Bergström, S. & Leppänen, A. (2015). Yrityksen asiakasmarkkinointi. Helsinki: Edita Prima Oy.

Cypress. (2023). Test. Automate. Accelerate. Noudettu 17.11.2023 osoitteesta https://www.cypress.io

Cohen, W. (2006). Marketing Plan. New Jersey. Wiley & Sons

Docker. (2023). The Docker platform. Noudettu 12.9.2023 osoitteesta https://docs.docker.com/get-started/overview/

Etelämäki, T. (2021). Mitä seniori toivoo käyttöliittymältä? Sytyke ry. Noudettu 12.9.2023 osoitteesta https://www.sytyke.org/digitalisaatio/mita-seniori-toivoo-kayttoliittymalta/

Express. (2023). Using middleware. Noudettu 17.11.2023 osoitteesta https://expressjs.com/en/guide/using-middleware.html

Finder. (2023). SaunaOnline Taloustiedot. Noudettu 20.6.2023 osoitteesta https://www.finder.fi/Tilaussauna/SaunaOnline/Helsinki/yhteystiedot/2923245

German, J. D., Yap, D. C. G & Binoya, G. O. (2021). "Design and Development of an Integrated Room Reservation System for Higher Education Institutions" 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA), Chengdu, China, 2021, pp. 231-236, doi: 10.1109/ICIEA52957.2021.9436766.

He, H. (2010). "Applications deployment on the SaaS platform," 5th International Conference on Pervasive Computing and Applications, Maribor, Slovenia, 2010, pp. 232-237, doi: 10.1109/ICPCA.2010.5704104.

IBM. (2023). What is software testing? Noudettu 17.11.2023 osoitteesta https://www.ibm.com/topics/software-testing

Internet Engineering Task Force. (2012). The OAuth 2.0 Authorization Framework. Noudettu 17.11.2023 osoitteesta https://datatracker.ietf.org/doc/html/rfc6749

Isoherranen, V. & Ratnayake, R. M. C. (2019). "Use of Pull Product Development for Enhancing Lean Startups," 2019 IEEE International Conference on Industrial

Engineering and Engineering Management (IEEM), Macao, China, 2019, pp. 736-740, doi: 10.1109/IEEM44572.2019.8978890.

Isännöintiliitto. (2023). Mitä on isännöinti? Noudettu 14.5.2023 osoitteesta https://www.isannointiliitto.fi/mita-on-isannointi/

Kaur, T., Kaur, D. & Aggarwal, A. (2014). "Cost model for software as a service," 2014 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), Noida, India, 2014, pp. 736-741, doi: 10.1109/CONFLUENCE.2014.6949281.

Kavcic, "Software Accessibility: Recommendations and Guidelines," EUROCON 2005 - The International Conference on "Computer as a Tool", Belgrade, Serbia, 2005, pp. 1024-1027, doi: 10.1109/EURCON.2005.1630123.

Laudon, K. C. & Traver, C. G. 2021. E-Commerce 2020-2021 : Business, Technology, Society, 16th Edition, Pearson.

Lipiäinen, T. (2000). Liiketoiminnan suunnittelu, markkinointi ja johtaminen uudella vuosikymmenellä. Jyväskylä: Gummerus Kirjapaino Oy.

Mongodb. (2023). What is MongoDB? Noudettu 17.11.2023 osoitteesta https://www.mongodb.com/what-is-mongodb

Murin, M., Jakab, F. & Michalko, M. (2021). "Advanced reservation system," 2021 19th International Conference on Emerging eLearning Technologies and Applications (ICETA), Košice, Slovakia, 2021, pp. 267-275, doi: 10.1109/ICETA54173.2021.9726673.

Nodejs. (2023). About Node.js. Noudettu 17.11.2023 osoitteesta https://nodejs.org/en/about

Okta. (2023). Introduction to JSON Web Tokens. Noudettu 12.9.2023 osoitteesta https://jwt.io/introduction

Rahman, A. & Subriadi, A. P. (2022). "Software as a Service (SaaS) Adoption Factors: Individual and Organizational Perspective," 2022 2nd International Conference on Information Technology and Education (ICIT&E), Malang, Indonesia, 2022, pp. 31-36, doi: 10.1109/ICITE54466.2022.9759891.

React. (2023). React – The library for web and native user interfaces. Noudettu 17.11.2023 osoitteesta https://react.dev/learn

React-Redux. (2023a). connect(). Noudettu 17.11.2023 osoitteesta https://react-redux.js.org/api/connect

React-Redux. (2023b). Using Hooks in a React Redux App. Noudettu 17.11.2023 osoitteesta https://react-redux.js.org/api/hooks

Red Hat. (2023). What is YAML?. Noudettu 15.4.2023 osoitteesta https://www.redhat.com/en/topics/automation/what-is-yaml

Redis. (2023). Introduction to Redis. Noudettu 15.4.2023 osoitteesta https://redis.io/docs/about/

Redux. (2023a). Redux – A Predictable State Container for JS Apps. Noudettu 17.11.2023 osoitteesta https://redux.js.org

Redux. (2023b). Getting Started with Redux. Noudettu 17.11.2023 osoitteesta https://redux.js.org/introduction/getting-started

Rope, T. (2005). Suuri markkinointikirja. Helsinki: Talentum.

SaunaOnline. (2023). Hinnoittelu. Noudettu 15.4.2023 osoitteesta https://saunaonline.fi/taloyhtio/hinnoittelu

Virtanen, P. (2010). Markkinoi ja myy oikein, sallitut ja kielletyt markkinointikeinot.

Wang, J., Luo, W., Wu, X., Li, T., Qian, Y. & Xie, Z. (2012). "An approach to modeling SaaS-oriented software service processes," 2012 International Conference on System Science and Engineering (ICSSE), Dalian, China, 2012, pp. 573-577, doi: 10.1109/ICSSE.2012.6257252.

W3C. (2023). Web Content Accessibility Guidelines (WCAG) 2.1. Noudettu 15.4.2023 osoitteesta https://www.w3.org/TR/WCAG21/

Xiong, H., Wang, P. & Zhu, G. (2010). "The Services Pricing Strategy on Software as a Service," 2010 International Conference on Management and Service Science, Wuhan, China, 2010, pp. 1-5, doi: 10.1109/ICMSS.2010.5576650.