Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Eero Nieminen

# Vuorosuunnittelu-lisäpalvelun kehitys

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikka

Insinöörityö

5.4.2021

| | |
|---|---|
| Tekijä<br>Otsikko<br><br>Sivumäärä<br>Aika | Eero Nieminen<br>Vuorosuunnittelu-lisäpalvelun kehitys<br><br>30 sivua<br>5.4.2021 |
| Tutkinto | Insinööri (AMK) |
| Tutkinto-ohjelma | Tieto- ja viestintätekniikka |
| Ammatillinen pääaine | Ohjelmistotuotanto |
| Ohjaajat | Lehtori Ilpo Kuivanen<br>Ohjelmistokehittäjä Krister Landén |


Insinöörityön aiheena oli kuvata Vuorosuunnittelu-lisäpalvelun taustaa, kehitystä sekä toimintaa. Tavoitteena oli kehittää hyvin toimiva, vuoronvarausarkea helpottava työkalu seuran tilojen varausten ja vuorojen hallintaan. Palvelu kehitettiin lisäpalveluksi myClubiin, joka on seurojen sähköinen jäsenpalvelu.

Vuorosuunnittelu-lisäpalvelun kehitys toteutettiin Scrum-projektinhallinnan viitekehystä noudattaen kahden viikon sprinteissä. Insinöörityössä perehdytään palvelun taustaan, suunnitelmaan, aloitukseen sekä kehityksen eri vaiheisiin. Työssä tarkastellaan myös markkinoilla jo valmiiksi olevia varausjärjestelmiä sekä niiden toimintaa.

Vuoronvaraukseen tehtiin käyttäjätutkimus, jonka perusteella palvelun käyttöliittymän ja toiminnan suunnitelma tehtiin. Tavoitteena oli tuoda helpotusta seurojen tilojen varausten hallintaan. Työssä perehdyttiin myClub-integraatioon sekä vuoro- ja varausnäkymien kehitykseen ja toimintaan.

Tuloksena saatiin ominaisuuksiltaan monipuolinen ja toimiva vuorosuunnittelujärjestelmä, joka toimii yhteydessä myClubin kanssa. Vuorosuunnittelu on osana myClubin lisäpalvelutarjontaa ja on tilattavissa myClubia käyttäville seuroille.



| | |
|---|---|
| Avainsanat | Vuorovaraus React Ruby on Rails |

| | |
|---|---|
| Author<br>Title<br><br>Number of Pages<br>Date | Eero Nieminen<br>Development of Vuorosuunnittelu Add-On<br><br>30 pages<br>5 April 2021 |
| Degree | Bachelor of Engineering |
| Degree Programme | Information and Communications Technology |
| Professional Major | Software Engineering |
| Instructors | Senior Lecturer Ilpo Kuivanen<br>Software Developer Krister Landén |

The subject of the study was to describe the background, development and functionalities of the Vuorosuunnittelu add-on service. The goal was to develop a well-functioning tool that provides relief to club's venues shift and reservation management needs. The service was developed as an add-on service to myClub which is a web-based service developed to help clubs manage their members, events and invoicing.

Scrum project management framework was used with two-week sprints in the development of Vuorosuunnittelu add-on service. The study introduces the background, planning, start of the service and the various stages of development. The study also examines the shift planning services already on the market and their operation.

The user interface and functionality of the service were planned based on user study that was made prior. The goal was to bring relief to the management of club’s venues reservations. The study focused on the myClub-integration, development and functionality of shift and reservation views.

The result was a feature-rich and functional shift planning system that works in conjunction with myClub. Vuorosuunnittelu is part of myClub's add-on service offering and can be ordered for clubs using myClub.

| | |
|---|---|
| Keywords | Shift planning React Ruby on Rails |

## Sisällys

## Lyhenteet

| | |
|---|---|
| API | Application Programming Interface. Ohjelmointirajapinta, jonka avulla eri sovellukset voivat välittää tietoa toistensa välillä. |
| CSS | Cascading Style Sheets. Yleisesti HTML-elementtien tyylien kuvaamiseen käytettävä tyyliohjekieli. |
| HTML | Hypertext Markup Language. Kuvauskieli, jota käytetään web-sivujen rakenteen esittämiseen. |
| JSON | JavaScript Object Notation. Avain-arvopareista koostuva tiedostomuoto, jota käytetään usein palvelimen ja asiakkaan väliseen tiedon välitykseen. |
| JSX | JavaScript XML. JavaScriptin syntaksilaajennus, joka helpottaa HTML-elementtien kuvaamista React-ohjelmistokehyksessä. |
| MVC | Ohjelmistokehityksen suunnittelumalli, jossa ohjelmiston rakenne on jaettu kolmeen osaan: malliin (Model), näkymään (View) ja ohjaimeen (Controller). |
| SaaS | Software as a Service. Tilauskohtainen palvelu, joka tarjoaa käytön internetin kautta ilman tarvetta asentaa palvelua paikallisesti. |
| XML | eXtensible Markup Language. HTML:n kaltainen kuvauskieli, jota käytetään tiedon välittämiseen sekä tiedostomuotona dokumenttien tallennukseen. |

# 1 Johdanto

Maailman muuttuessa ja teknologian kehittyessä yhä enemmän asioita hoidetaan verkossa, ja erilaisille verkossa toimiville palveluille on entistä suurempi kysyntä. Digitalisaation myötä yhä suurempi osa palveluista myös toimii verkossa. Erilaiset verkossa toimivat varausjärjestelmät ovat Suomessa nykypäivää. Esimerkiksi hammaslääkäriin tai auton huoltoon on voinut varata ajan netissä jo pitkään. Useimmat kaupungit ja toimialat ovatkin jo siirtyneet sähköisiin ajanvarausjärjestelmiin.

Monien muiden toimialojen lailla myös urheiluseurojen on pysyttävä kehityksen aallon harjalla. Hallin seinään tussilla merkittyjen tai tulostettujen taulukoiden aika on ollut jo pidemmän aikaa historiaa. Toisaalta Excel tai muut vastaavat taulukko-ohjelmat eivät ole olleet kovin toimiva tai ainakaan joustavin ratkaisu: hallivuorojen suunnitteleminen ja jakaminen taulukko-ohjelman avulla voi olla hidasta ja hankalaa. Myös seurattavuus sekä laskutuksen hallinta voi olla tuskallista, jos siihen ei ole oikeita työkaluja. Ratkaisuna näihin ongelmiin on kehitetty Vuorosuunnittelu-lisäpalvelu. Vuorosuunnittelu-lisäpalvelu tuo helpotusta ja ajansäästöä seurojen urheilutilojen hallintaan ja seurantaan. Vuorosuunnittelun avulla seura voi myös hoitaa tiloihin liittyvien varausten laskutusta.

Insinöörityön tarkoituksena oli kehittää suomalaisen Taikala Oy:n kehittämälle myClub-palvelulle Vuorosuunnittelu-lisäpalvelu tuomaan helpotusta seuran tilojenhallinnan arkeen. Vuorojen suunnittelu on olennainen osa joidenkin seurojen toimintaa, joten palvelulle oli selvä tarve. Asiakkaiden tarpeen lisäksi palvelun kehitys oli oleellista myös kilpailukyvyn kannalta.

Opinnäytetyön tavoitteena on kuvata Vuorosuunnittelu-lisäpalvelun kehitysprossia ja palvelun toimintaperiaatteita. Pääpaino on Vuorosuunnittelu-lisäpalvelun suunnittelun, toteutuksen ja perustoiminnallisuuksien kuvaamisessa. Insinöörityössä käsitellään sitä, mitä itse vuorovarausjärjestelmällä tarkoitetaan, millaisia toimijoita markkinoilla on ja mitä hyötyä vuoronvarausjärjestelmät tuovat yrityksille ja yksityishenkilöille. Teoriaosuudessa esitellään projektin teknologioita ja niiden toimintaa sekä soveltuvuutta kokonaisuudessa. Lisäksi teoriaosuudessa tutustutaan myClubiin ja siihen, miten Vuorosuunnittelu liittyy siihen.

Insinöörityö tehtiin kehitystyönä Taikala Oy:lle, jonka palvelu myClub on. Taikala Oy on vuonna 2009 perustettu ohjelmistotalo, joka tarjoaa tuote- ja projektipalveluita moderneilla web-teknologioilla.

## 2 Vuoronvarausjärjestelmät

Vuoronvarausjärjestelmällä tarkoitetaan yleensä sähköistä varausjärjestelmää, jossa käyttäjän on mahdollista tehdä tai nähdä erilaisia vuoroja ja varauksia. Asiat, jotka on ennen tehty soittamalla tai sähköposteja lähettämällä, onnistuvat vuoronvarausjärjestelmää käyttämällä helposti verkossa. Monissa varausjärjestelmissä on myös laskutusominaisuudet. Laskutusominaisuuksien avulla varausjärjestelmässä voi tehdä suoraan laskuja esimerkiksi asiakkaille tai seuran jäsenille. Varausjärjestelmän avulla asiakkaalle jää enemmän aikaa muihin asioihin, koska monet asiat voidaan automatisoida palvelun avulla. Varausjärjestelmät tuovat myös seurattavuuden ja raportoinnin mahdollisuuden käyttäjilleen. Varausjärjestelmät ovatkin saavuttaneet suuren suosion niiden helppouden ja tehokkuuden vuoksi. (Rowena Gungon 2021.)

Markkinoilla on useita vuoronvarausjärjestelmiä, joista yksi tunnettu palvelu on Timmi. Timmi on käytössä monella kaupungilla tilojen varauksien hallintajärjestelmänä, ja varaukset voivat olla nähtävissä julkisesti. Timmin avulla kaupungit pystyvät kirjaamaan eri tilojen varauksia ja hallitsemaan myös niiden laskutusta, sillä Timmi sisältää myös laskutuspalvelun. Timmissä voi valita tilan ja katsoa varauskalenterista varauksia, joita tiloille on tehty. (TIMMI Tilavarausohjelmisto 2021.) Näin esimerkiksi koululainen voi käydä katsomassa, missä tilassa ja mihin aikaan koulun liikuntatunti on.

Timmin lisäksi on olemassa myös muita vuoronvarauspalveluita, kuten Helsingin kaupungin pilotissa oleva Varaamo sekä ASIO:n eri varausjärjestelmät (Asio Varausjärjestelmä 2021; Tietoa varaamo.hel.fi –palvelusta 2021). Useista olemassa olevista vuoronvarausjärjestelmistä huolimatta Vuorosuunnittelu-lisäpalvelulla on paikkansa markkinoilla ennen kaikkea sen takia, että se on osa myClubia ja tarjoaa näin ollen käyttäjilleen heidän kaipaamaansa vuorosuunnittelupalvelua.

# 3 myClub

myClub on Suomen suosituin ja nopeimmin kasvava jäsenpalvelu, jolla on asiakkaanaan jo yli 650 seuraa. Jäsenpalvelulla tarkoitetaan seuran jäsenille tarkoitettuja sähköisiä palveluita. myClub on SaaS-periaatteella toimiva palvelu, joka sisältää myös mobiilisovellukset harrastajille sekä valmentajille. myClub tarjoaa välineet jäsenrekisterin hoitoon, laskutukseen, viestintään sekä tapahtumiin ilmoittautumiseen. (myClub ominaisuudet 2021.)

## 3.1 Laskutus ja maksutavat

myClubin yhtenä suurena etuna asiakkaiden kannalta on se, että palvelu tarjoaa laskutusominaisuuden suoraan myClub-palvelusta. myClubissa on mahdollista luoda rajaton määrä sähköpostilaskuja, jotka lähetetään jäsenille joko sähköpostin tai sovelluksen kautta. Palvelua käyttäville seuroille on tarjolla Pankkiyhteys-palvelu, joka mahdollistaa seuran pankin yhdistämisen suoraan myClubiin. Pankkiyhteys-palvelun avulla kaikki seuralle tehdyt maksusuoritukset tulevat automaattisesti näkyviin myClubiin. Näin ollen myClubia käyttävä seura pystyy seuraamaan vaivattomasti tilannettaan maksetuista laskuista ja taloudellisesta tilanteesta. (myClub ominaisuudet 2021.)

Saatavilla on myös Kirjanpito-lisäpalvelu, jonka avulla seuran on mahdollista saada täsmälliset kirjanpitoaineistot joko seura-, joukkue- tai ryhmäkohtaisesti. Nämä aineistot ovat myös automaattisesti vietävissä Procountoriin, Netvisoriin tai Excelin kautta kirjanpito-ohjelmaan. Laskutuksen lisäpalvelut mahdollistavat kokonaisvaltaisen laskutuskokemuksen, joka on toimiva niin hallittavuudeltaan kuin seurannan suhteen. (myClub ominaisuudet 2021.)

Lisäpalveluiden avulla myClub tarjoaa seurojen jäsenille myös erilaisia tapoja maksaa laskuja suoraan myClubista. Maksutapavaihtoehtoina on saatavilla verkkolaskutus Maventan kautta, verkkomaksut Paytrailin tai Svean kautta sekä MobilePay Invoice. MobilePay Invoice on muihin palveluihin nähden siten erityinen, että sen avulla seura voi lähettää laskut suoraan jäsenen mobiilisovellukseen, jossa jäsen voi maksaa laskunsa suoraan MobilePayllä. Mobiilimaksaminen onkin nopeimmin kasvava maksutapa. (myClub ominaisuudet 2021.)

### 3.2 Lisäominaisuudet

myClub sisältää integraatioita myös muihin palveluihin. Tällainen on esimerkiksi Palloliitto Pelipassi -lisäpalvelun kautta yhteys Pelipaikasta ostettuihin pelipasseihin. Kun seuralla on myClubissa käytössä Kortit-lisäpalvelu, voidaan pelipasseista tehdä kortteja myClubiin. Korttien avulla seura pystyy hoitamaan maksullisia tapahtumia. Tapahtumaan osallistuttaessa kortti näytetään mobiililaitteesta, ja ohjaaja tai valmentaja tarkistaa kortin myClub Coach -mobiilisovelluksella. Lisäpalvelu tuo näin ollen huomattavaa jouhevuutta tapahtumien kulunvalvontaan. Toisena vaihtoehtona tapahtumien kulunvalvonnalle on Aulamoodi, joka ei edellytä ketään ihmistä valvomaan tapahtumiin kulkua. Aulamoodi toimii viivakoodinlukijalla, jonka seura voi laittaa esimerkiksi tapahtumapaikan aulaan. Paikalle saapuessaan seuran jäsenet voivat helposti kuitata itsensä paikallaoleviksi viivakoodilukijan avulla. Aulamoodi-lisäpalvelu mahdollistaa siis kulunvalvonnan muuttamisen itsenäiseksi. (myClub ominaisuudet 2021.)

Lisäksi myClub tarjoaa verkkokaupan sekä kotisivut maksullisina lisäpalveluina. Personalisoidut kotisivut ovat joka urheiluseuran potentiaalisten jäsenten ensimmäinen kosketus seuraan, joten niiden on oltava vaikuttavat ja toiminnalliset. Vuorosuunnittelu tulee olemaan uusin lisäys lisäpalveluiden joukkoon, joka tarjoaa uudenlaisia ominaisuuksia seuralle.

## 4 Teknologiat ja kehitysympäristö

Tässä luvussa käsitellään Vuorosuunnittelu-lisäpalvelun kehityksessä käytettyjä teknologioita ja kehitysympäristöä. Projektissa käytettiin jo ennestään yrityksessä käytössä olleita teknologioita, sillä ne soveltuivat nykyaikaiseen web-kehitykseen. Projektin kehitykseen käytettiin macOS-käyttöjärjestelmää.

### 4.1 Git-versionhallinta

Git on yleisesti käytössä oleva hajautettu versionhallintajärjestelmä. Tämä tarkoittaa sitä, että jokaisella kehittäjällä on oma paikallinen kopio lähdekoodista, johon tehdään muu-

toksia. Näitä muutoksia tehdään kehityshaaroihin, jotka voivat olla tehty projektin päähaarasta tai esimerkiksi uuden ominaisuuden kehityshaarasta. Uuden ominaisuuden tai yksittäisen pienemmän osan valmistuessa haara voidaan liittää projektin vakaaseen haaraan. Tällä tavoin pystytään varmistamaan paremmin, etteivät uudet ominaisuudet riko projektin muita ominaisuuksia. Kuvattu työskentelymalli mahdollistaa myös usean kehittäjän yhtäaikaisen työskentelemisen samassa projektissa, koska muutokset eivät sotke toisiaan. Lisäksi Git tarjoaa mahdollisuuden tarkastella versionhallinnan muutoshistoriaa. (Git 2020.) Muutoshistoria tuo erittäin suuren helpotuksen etenkin silloin, jos myöhemmin huomataan, että muutos on rikkonut jonkin ominaisuuden. Muutoshistoriasta voi nähdä, minkä muutoksen jälkeen ongelmia on esiintynyt, ja näin ollen ne on helpompi korjata.

Vuorosuunnitteluprojektin lähdekoodin säilyttämiseen käytettiin Bitbuckettia, joka on Gittiin pohjautuva pilvipalvelu, jossa voi säilyttää repositorioita. Bitbucketin avulla voi myös hoitaa koodin automaattisen testauksen, ja lisäksi se sisältää integraation Jiraan. (A brief overview of Bitbucket 2021.)

### 4.2 Visual Studio Code

Visual Studio Code on Microsoftin kehittämä avoimen lähdekoodin tekstieditori, joka on saatavilla Windowsille, Linuxille ja Macille. Visual Studio Codella on tuki suoraan Javascriptille, TypeScriptille ja Node.js:lle. Näiden lisäksi editori tukee erilaisten lisäosien avulla useimpia yleisesti käytössä olevia ohjelmointikieliä. (Getting Started 2021.) Visual Studio Codeen on myös itse mahdollista kehittää omia lisäosia, koska sen lähdekoodi on avoin kaikille. Visual Studio Code oli vuoden 2019 Stack Overflown kyselyn mukaan suosituin tekstieditori (Visual Studio Code 2021).

### 4.3 Ruby

Ruby on vuonna 1995 julkaistu puhdas oliopohjainen ohjelmointikieli, jonka uusin versio on julkaistu joulukuussa 2020. Ruby on saanut vaikutteita erityisesti Lisp, Smalltalk ja Perl -ohjelmointikielistä, ja sen syntaksi muistuttaa paljolti Pythonia (About Ruby 2021). Rubya voidaan pitää suhteellisen helposti opittavissa olevana ohjelmointikielenä, ja sen

kieltä ja syntaksia helposti luettavana. Alla on annettu esimerkki yksikertaisesta Ruby-koodista (esimerkkikoodi 1).

```ruby
[2, 1, 6, 0].sort.each do |i|
   print i*i
end
# tulostaa "01436"
```

Esimerkkikoodi 1. Esimerkki Ruby-koodista. Koodi järjestää taulukon numeerisen arvon mukaiseen järjestykseen ja tulostaa lukujen neliöt.

Ruby saavutti suuren suosion vuonna 2006, mutta sen jälkeen suosio on laskenut ja tällä hetkellä Ruby on 15. suosituin ohjelmointikieli (TIOBE Index for January 2021). Ruby ei välttämättä ole yksinään paras työkalu web-kehitykseen, minkä takia sitä käytetään tyypillisimmin yhdessä Ruby on Rails -ohjelmointikehyksen kanssa, josta kerron enemmän seuraavassa luvussa.

### 4.4 Ruby on Rails

Ruby on Rails on vuonna 2005 julkaistu Ruby-ohjelmointikieleen perustuva MVC-mallin mukainen ohjelmistokehys, jota käytetään lähinnä web-sovellusten kehittämiseen (What's Rails 2021). MVC-malli on perinteinen ohjelmistokehityksen suunnittelumalli, jossa ohjelmiston rakenne erotellaan kolmeen osaan: malliin (Model), näkymään (View) ja ohjaimeen (Controller). Ruby on Rails tarjoaa kaikki työkalut moderniin web-kehitykseen pienistä projekteista jättimäisiin sovelluksiin, joista yksi hyvä esimerkki on GitHub (Rails has everything you need 2021).

Rails kannustaa käyttämään web-ohjelmoinnin standardeja kuten JSON:n ja XML:n tiedonsiirrossa sekä HTML-, CSS- ja JavaScript-näkymien toteutuksessa. Rails korostaa myös muiden yleisesti tunnettujen suunnittelumallien käyttöä, joita ovat esimerkiksi CoC ja DRY. CoC on lyhenne englanninkielisistä sanoista convention over configuration ja tarkoittaa vapaasti käännettynä sitä, että määrittelemisen sijaan suositaan konventiota eli yleisesti sovittua tapaa. CoC:sta yksi esimerkki on esimerkiksi se, että mallille annetaan nimeksi user, joten tietokannan taulun nimen oletetaan olevan users. Toimittaessa

tämän tavan mukaan kaikki toimii niin kuin pitää, mutta jos poiketaan tavasta ja annetaan mallille nimeksi club_user, joudutaan tekemään ylimääräisiä määrityksiä, jotta Rails osaa yhdistää tietokannan taulun ja mallin nimen. DRY tulee sanoista don't repeat yourself ja tarkoittaa, että kunkin ohjelmistoon liittyvä seikka tulee olla vain kerran määritetty yhdessä paikassa. Tällä pyritään välttämään koodin toistuvuutta, mikä puolestaan parantaa koodin laatua. (Ruby on Rails 2021.)

Ruby on Railsin suurimpana etuna voidaan pitää sen tuotteliaisuutta ja kykyä saada paljon aikaan vähäisellä määrällä koodia ja vähällä konfiguroinnin tarpeella. Ruby on Rails soveltuu erinomaisesti uusien projektien aloittamiseen, koska se on nopeaa ja tehokasta.

## 4.5 React

React on avoimen lähdekoodin JavaScript-kirjasto, jota käytetään käyttöliittymien rakentamiseen erityisesti yhden sivun applikaatioissa. React on alun perin julkaistu vuonna 2013 ja on Facebookin ylläpitämä. React on maailman suosituin JavaScript-kirjasto. (Christian Ström 2019.)

### JSX

React käyttää JSX-syntaksia, joka tulee sanoista JavaScript XML. JSX muistuttaa paljolti HTML-syntaksia. JSX:llä on samat säännöt kuin XML:llä, eli jokainen tagi täytyy sulkea. JSX:n etuja on, että kehittäjä voi sijoittaa JSX:n tagien sisälle normaalia JavaScript koodia silloin, kun se on annettu aaltosulkeiden sisällä. Tämä mahdollistaa esimerkiksi virheilmoitusten näyttämisen helposti. Alla on esimerkki React koodista, jossa on hyödynnetty JSX:n ominaisuuksia (esimerkkikoodi 2).

```
function fullName(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'John',
  lastName: 'Doe'
};

return (
  <h1>
    Hello, {fullName(user)}!
  </h1>
);
```

Esimerkkikoodi 2. React-koodiesimerkki, jossa kutsutaan JavaScript-funktiota JSX-tagin sisällä, joka palauttaa user-olion, kokonaisen nimen.

### Komponentit ja propsit

Reactissa sivun rakennusosia kutsutaan komponenteiksi. Kaikki Reactilla rakennetut sivut koostuvat useasta komponentista. Komponentit sallivat käyttöliittymän jakamisen itsenäisiin osiin, ja ne ovat uudelleenkäytettäviä. Reactissa voidaan tehdä joko luokkapohjaisia tai funktionaalisia komponentteja, eli JavaScript-funktioita. (Christian Ström 2019.) Seuraavissa koodiesimerkeissä 3 ja 4 on esitelty molemmat tavat.

```
function Title(props) {
  return <h1>Hello, {props.title}</h1>;
}
```

Esimerkkikoodi 3. Funktionaalisen komponentin luominen.

```
class Title extends React.Component {
  render() {
    return <h1>Hello, {this.props.title}</h1>;
  }
}
```

Esimerkkikoodi 4. Luokkapohjaisen komponentin luominen.

Niin kuin esimerkkikoodissa 4 näkyy, komponenteille voidaan antaa mielivaltaisia syötteitä, joita kutsutaan propseiksi. Props on lyhenne englannin kielen sanasta ”Properties”. Propsien avulla tietoa voidaan välittää komponenttien välillä. Propseja voidaan myös välittää ”parent”-komponentilta ”child”-komponentille, mikä helpottaa uudelleen käytettävien komponenttien luomista.

Yksi tärkeä sääntö Reactissa on, että komponentti ei saa muuttaa omia propsejaan suoraan, vaan muuttaminen tehdään komponentin tilaa, eli “Statea” muuttamalla. Komponentin tila voidaan antaa suoraan vain konstruktorissa, eikä sitä saa muuttaa suoraan, vaan sitä hallitaan setState()-funktiolla tai State-hookilla. Tilat ovat luokkapohjaisten React-komponenttien ominaisuus, mutta hookien avulla Reactin funktionaalisiin komponentteihin on saatu luokista tuttuja ominaisuuksia kuten tila ja sen päivitys. Reactissa tilan päivitys aiheuttaa uuden renderöinnin, joka päivittää sivun uudella tiedolla. (Christian Ström 2019.) Komponenttien määrän kasvaessa tilojen hallitsemisesta tulee entistä haastavampaa. Tätä ongelmaa ratkaisemaan on kehitetty Redux, josta kerron lisää seuraavassa luvussa.

## 4.6 Redux

Redux on Dan Abramovin ja Andrew Clarkin vuonna 2015 julkaisema avoimen lähdekoodin JavaScript-kirjasto. Se on luotu nimenomaan sovellusten tilan hallitsemiseen. Reduxia käytetään yleensä Reactin tai Angularin kanssa (Redux (JavaScript Library) 2021).

### Store

Redux tarjoaa yhden paikan, missä sovelluksen koko tila sisällytetään yhden objektin sisään. Tästä käytetään käsitettä ”store”. Sovelluksella on suositeltavaa olla vain yksi store, mutta on myös mahdollista luoda useampi store. Jokaisella React-komponentilla on mahdollisuus päästä käsiksi sovelluksen tilaan ilman, että propseja täytyy välittää komponenttien välillä. Tämä yksinkertaistaa ja helpottaa sovelluksen tilan hallitsemista (Neo Ighodora 2020).

### Actionit

Actionit ovat tapahtumia, joiden avulla tietoa voidaan lähettää sovelluksesta storeen. Actioneita voidaan lähettää käyttämällä store.dispatch()-funktiota. Actionit ovatkin vain JavaScript funktioita, joille on annettava type, jotta tiedetään, millainen toiminto on kyseessä, sekä payload, joka sisältää lähetettävän tiedon. Action creator luo ja palauttaa actionin (Neo Ighodora 2020). Esimerkkikoodissa 5 on esitetty esimerkki action creatorista, joka palauttaa actionin.

```
const showNotification = (content) => ({
  type: SHOW_NOTIFICATION,
  payload: content
});
```

Esimerkkikoodi 5. Redux action creator, joka palauttaa actionin, joka sisältää payloadin sekä actionin tyypin.

### Reducerit

Reducerit ovat puhtaita funktioita, joiden avulla storen tilaa voidaan päivittää. Puhdas funktio tarkoittaa funktiota, jolla ei ole mitään sivuvaikutuksia ja joka palauttaa aina saman arvon, jos se saa samat parametrit (Fortune Ikechi 2020). Alla on esimerkki puhtaasta funktiosta (esimerkkikoodi 6).

```
const multiply = (x, y) => x * y;

multiply(5, 6);

// tulostaa 30
```

Esimerkkikoodi 6. Esimerkki puhtaasta JavaScript-funktiosta, joka palauttaa tuloksena parametreina annettujen numeroiden tulon.

Reducerit ottavat vastaan sovelluksen tilan ja actionin, joista muodostetaan uusi tila. Tämän jälkeen reducer palauttaa uuden tilan, joka sisältää tehdyt muutokset. (Neo Ighodora 2020.) Tärkeä sääntö reducereiden kohdalla onkin, että ne palauttavat aina uuden sovelluksen tilan eivätkä muuta tilaa suoraan. Alla on esimerkki reducer-funktiosta (Esimerkkikoodi 7).

```
function (state = {}, action) {
  switch (action.type) {
    case SHOW_MODAL:
      return { ...state, open: true };
    case HIDE_MODAL:
      return { ...state, open: false };
  }
  return state;
}
```

Esimerkkikoodi 7. Reducer-funktio, joka palauttaa uuden tilan, joka sisältää tilaan tehdyt muutokset.

### Hyödyt ja haitat

Redux on hyvä työkalu sovelluksen tilan hallitsemiseen, varsinkin kun käytössä on React. Redux lisää melko paljon koodin kirjoittamista projektiin, mutta samalla se yksinkertaistaa tilojen hallitsemista. Pienessä projektissa Reduxin käyttö voi kuitenkin tuntua vaivalloiselta, koska pienen muutoksen aikaansaamiksesi täytyy kirjoittaa paljon koodia. Aloittelijalle, joka vasta tutustuu Reactiin, voi Redux olla hieman hankala ottaa mukaan, sillä sen opettelu vie oman aikansa.

# 5 Vuorosuunnittelu – aloitus ja suunnitelma

Ohjelmiston kehityksen suunnitteluun liittyy monia ratkaisevia seikkoja. Jokainen projekti aloitetaankin suunnitelman laatimisella. Ihan ensimmäiseksi on määritettävä, onko ohjelmistolle kysyntää ja onko sille paikka markkinoilla. Kun kysynnän tarve on todettu, ruvetaan työstämään palvelun kehittämissuunnitelmaa. Kehittämissuunnitelmassa luonnostellaan sitä, miten ohjelmiston pitäisi toimia ja miltä sen kuuluisi näyttää. Suunnitelmaa laadittaessa on myös mietittävä tuotteen hinta tulevalle käyttäjälle, jotta voidaan arvioida projektin kannattavuutta ja kuinka paljon projektiin on järkevää käyttää resursseja.

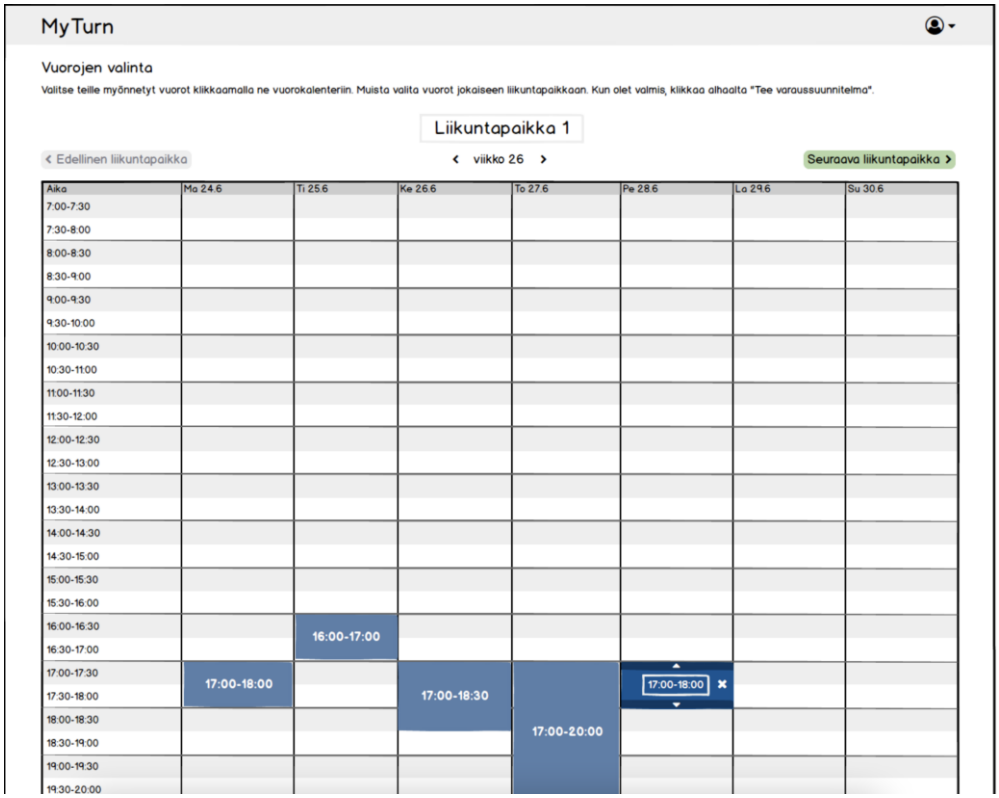
Projektin tavoitteena on luonnollisesti kehittää mahdollisimman toimiva ohjelmisto käyttäjien näkökulmasta. Käyttäjätutkimus on yksi hyvä tapa kartoittaa sitä, mitä tulevat käyttäjät odottavat ja toivovat tulevalta ohjelmistolta. Tämän takia myös Vuorosuunnittelulisäpalvelun kehittämisprojektia varten tehtiin käyttäjätutkimus, jossa haastatelluilta urheiluseuroilta kysyttiin, miten he hoitavat vuorojen suunnittelun, varausten tekemisen, laskutuksen ja ilmoittamisen. Kyselyn tuloksena oli, että suurin osa seuroista saa vuoronsa kaupungilta tai muulta tilojen vuokraajalta. Kun seura on saanut vuoronsa, se merkitsee vuorot Exceliin, josta ne jaetaan PDF-muodossa seuran joukkueille ja jäsenille. Excel-taulukosta seuran toimihenkilöt voivat syöttää joukkuevuorot myClubiin tapahtumiksi. Tulokset vahvistivat tarpeen palvelulle, joka sisältää kaiken tarpeellisen vuorojen suunnitteluun sekä helpon tavan viedä ne myClubiin tapahtumiksi ilman turhia välivaiheita.

## Käyttöliittymän suunnittelu

Käyttöliittymä ja käytettävyys ovat erittäin tärkeitä kaikissa sovelluksissa. Hyvä käyttökokemus onkin nykyään web-palveluissa ensisijaista. Vaikka järjestelmä olisi kuinka monipuolinen ja ominaisuuksiltaan rikas, ei järjestelmä ole mitään ilman käytettävää ja selkeää käyttöliittymää. Suunnittelussa onkin pohdittava, miten palvelu saadaan toimimaan niin, että sen käyttö on käyttäjälle mahdollisimman luontevaa ja vaivatonta. Toimivan käyttöliittymän suunnitteleminen ja luominen on kuitenkin erittäin haastavaa, sillä suunnittelijan on mahdotonta tietää, miten loppukäyttäjä oikeasti kokee palvelun käytön. Tämän takia yhteistyö tulevien palvelun käyttäjien kanssa on tärkeää. Palvelun suunnittelijan tulee tehdä käytettävyystestejä, joiden avulla tulevilta käyttäjiltä saadaan palautetta

palvelun käytettävyydestä. Käytettävyystesteistä saadun palautteen perusteella palvelua voidaan muokata ja parannella, jolloin käytettävyys paranee käyttäjien palautteiden ansiosta. Hyvä käyttökokemus voi myös antaa edun kilpailijoiden suhteen, jos kilpailevissa palveluissa ei ole annettu käyttökokemukselle tarpeeksi huomiota tai arvostusta. Monet suuryritykset ovatkin tehneet massiivisia sijoituksia käyttökokemuksen parantamiseksi. (Joonas Virtanen 2016.)

Projektin etenemistä helpottaa ja nopeuttaa yleisen mallin laatiminen. Vuorosuunnittelun kehityksen alussa hahmoteltiin, miltä palvelun käyttöliittymän tulisi näyttää ja miten vuoroja suunnitellaan. Kuvassa 1 on ensimmäinen konsepti, jossa on esitetty, miltä palvelun viikkonäkymä voisi näyttää ja mitä toiminnallisuuksia se sisältäisi.

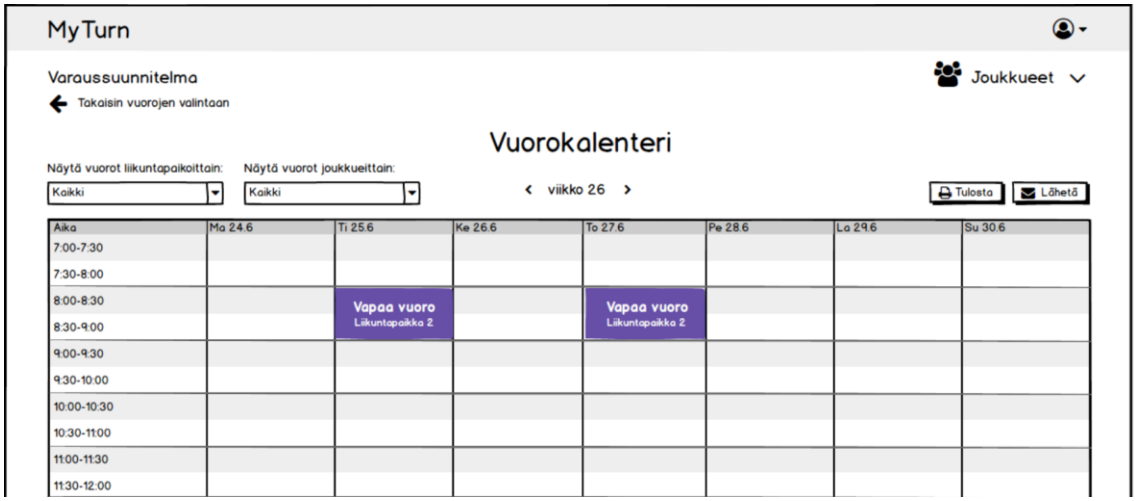



Kuva 1. Konsepti Vuorosuunnittelun viikkonäkymästä.

Viikkonäkymässä (kuva 1) nähdään vuoroja sinisinä laatikkoina, jotka ovat lukujärjestyksen mallisessa käyttöliittymässä. Kuvan 1 yläosassa on ilmoitettu liikuntapaikka, jota voi vaihdella, sekä viikon numero (esim. viikko 26), jonka molemmin puolin olevista nuolista voi vaihtaa näkyvää viikkoa. Kuvan 1 näkymän ajatuksena on, että käyttäjä voi luoda vuoroja liikuntapaikalle “maalaamalla” haluamansa ajankohdan. Tarvittaessa varausta

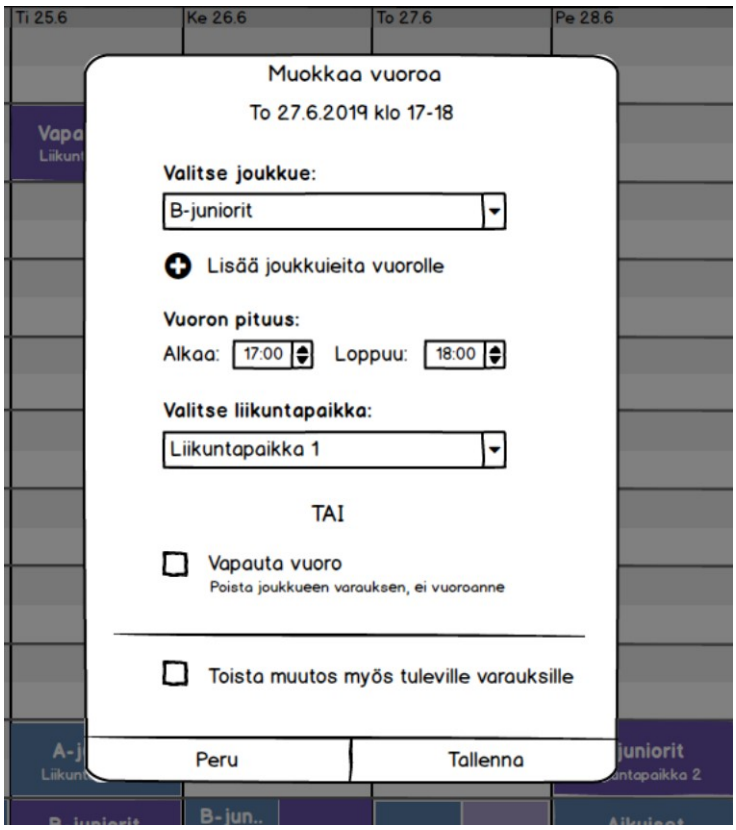
pystyisi muokkaamaan “vetämällä” tai klikkaamalla varausta, jolloin aukeaisi muokkausmodaali.

Kalenterin viikkonäkymään oli aluksi hahmoteltu myös erilaisia suodattimia, millä käyttäjä voisi rajata kalenterissa näytettäviä vuoroja. Kuvassa 2 on esitetty, millaisilta suodattimet voisivat näyttää. Vuoroja olisi siis mahdollista suodattaa liikuntapaikan ja joukkueen mukaan. Kuvasta 2 näkyy myös, että kalenteri olisi mahdollista lähettää sähköpostilla tai tulostaa. Verrattaessa kuvaa 2 kuvaan 1 huomataan myös, että vuorojen väri on eri. Tarkoituksena oli, että kaikilla liikuntapaikoilla voisi olla oman väriset vuorot.




Kuva 2. Konsepti Vuorosuunnittelun viikkonäkymän suodattimista.

Vuoroille olisi myös oltava mahdollisuus muokkaamiseen. Tähän suunniteltiin, että vuoroa klikkaamalla avautuisi muokkausmodaali, josta yksittäisen vuoron tietoja voisi helposti muokata. Kuvassa 3 on konsepti tästä. Vuorolle voitaisiin valita joukkue sekä lisätä joukkueita. Vuoron alkamis- ja päättymisaikaa voitaisiin muokata syöttämällä aika joko kirjoittamalla tai klikkailemalla nuolia. Vuoron voisi myös siirtää eri liikuntapaikkaan valitsemalla sen valikosta. Vuoro olisi myös mahdollista vapauttaa eli poistaa joukkueen varaus vuorolta. Toistuville vuoroille olisi mahdollista toistaa yksittäiseen vuoroon tehdyt muutokset, jolloin kaikkia vuoroja ei tarvitse yksitellen muokata.




Kuva 3. Konsepti vuoron muokkaamisesta.

Suunnitelman valmistumisen jälkeen projektin varsinainen toteutus voitiin aloittaa, kun tiedettiin, mitä lähdetään ensimmäiseksi tekemään ja mitä ominaisuuksia on tarkoitus tehdä. Projektin tyyleiksi ja väreiksi valittiin myClubista jo löytyvät tyylisuunnat sekä värit.

# 6 Työn toteutus

Tässä luvussa kerrotaan Vuorosuunnittelu-lisäpalvelu projektin toteutuksesta, eli millä tavoin kehityksen prosessi eteni. Ensin perehdytään siihen, mitä projektinhallintaa oli käytetty, minkä jälkeen kuvataan Vuorosuunnittelun toimintaa, ominaisuuksia sekä teknistä toteutusta.

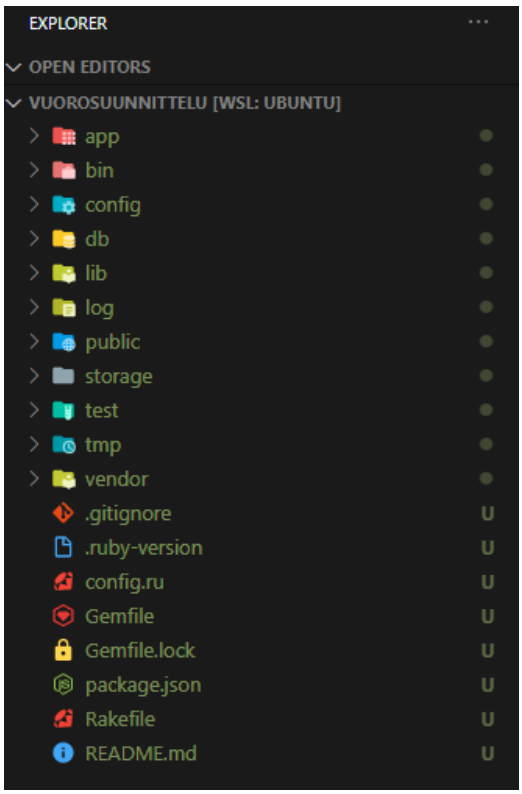
## 6.1 Toteutustapa

Vuorosuunnittelu-projekti toteutettiin Scrum-projektinhallinnan viitekehyksen mukaan, jota käytetään ketterässä ohjelmistokehityksessä. Scrumissa työskennellään 1–4 viikon mittaisissa sprinteissä, joissa tuotetta kehitetään valmiimmaksi ja paremmaksi useissa lyhyissä kehitysjaksoissa. Vuorosuunnitteluprojekti toteutettiin kahden viikon mittaisissa sprinteissä. Sprinttien välissä järjestettiin sprinttipalaverit, joissa käytiin läpi tehdyt parannukset ja määriteltiin seuraavassa sprintissä tehtävät asiat sekä sprintin tavoite.

## 6.2 Rails-projektin luominen

Vuorosuunnittelun ensimmäinen vaihe oli Rails-projektin luominen. Rails-projektin luominen vaatii, että kehittäjän koneelle on asennettu Ruby. Koska kehitys tapahtui macOS-käyttäjärjestelmällä, Ruby oli valmiiksi asennettuna. Eri projekteissa voi tosin olla eri Ruby-versio, minkä takia on hyvä olla tapa vaihtaa käytössä olevaa Ruby-versiota helposti komentoriviltä. Tätä varten on kehitetty RVM eli Ruby Version Manager. RVM on komentorivityökalu, jonka avulla on mahdollista hallita ja asentaa eri Ruby-versioita (Ruby Version Manager (RVM) 2017). Projekti kehittäminen vaatii myös, että kehittäjän koneelle on asennettuna Node.js ja yarn, joiden avulla sovelluksen JavaScriptejä voidaan hallita. Node.js on Google Chromen V8 JavaScript-moottoriin pohjautuva avoimen lähdekoodin ajoympäristö, jonka avulla palvelimella voidaan suorittaa JavaScript-koodia (Node.js 2021). Yarn on JavaScript-pakettien hallintatyökalu, joka sisältää kaikki projektiin liittyvät JavaScript-paketit tiedostossa nimeltä package.json (Yarn Introduction 2021). Monissa projekteissa voi myös olla eri Node.js versio. Node.js-versioiden hallitsemiseksi on kehitetty nvm eli Node Version Manager.

Rails tarjoaa erilaisia apuvälineitä skriptien muodossa. Näitä kutsutaan generaattoreiksi. Yksi näistä generaattoreista on uuden projektin luomisgeneraattori. Generaattorin käyttö mahdollistaa uuden Rails-projektin luomisen yhdellä komennolla komentorivillä. ”rails new vuorosuunnittelu” luo uuden projektin, jossa on valmiiksi kaikki tarvittavat tiedostot ja kansiot projektin aloittamiseksi. Kuvassa 4 on nähtävissä uuden Rails-projektin tiedostorakenne.




Kuva 4. Uuden Rails-projektin tiedostorakenne.

Projektin luomisskriptin jälkeen Vuorosuunnittelu-projektin pitäisi olla valmis toimimaan kehitysympäristössä. Ajamalla komentorivillä ”rails server” voidaan käynnistää sovellus. Kuvassa 5 nähdään, miltä uuden Rails-projektin vakiosivu näyttää.


Kuva 5. Uuden Rails-projektin vakiosivu.

Projektin luonnin ja kehitykseen tarvittavien vaatimusten täyttämisen jälkeen oli mahdollista alkaa kehittämään ensimmäisiä ominaisuuksia Vuorosuunnittelulle. Näistä kerrotaan lisää seuraavassa luvussa.

### 6.3 Kalenteri, vuorot ja varaukset

Vuorosuunnittelun kehittämisen ensimmäiseksi tavoitteeksi asetettiin toimivan kalenterin luominen. Kalenterissa olisi mahdollista luoda vuoroja ja varauksia, minkä takia päädyttiin kahden kalenterinäkymän luomiseen. Toisessa kalenterinäkymässä käyttäjä voisi tehdä vuoroja ja toisessa näkymässä kyseisten vuorojen päälle olisi mahdollista tehdä varauksia. Tarpeena oli myös, että vuoroja ja varuksia olisi mahdollista piirtää sekä vaihtaa vuorojen ja varausten pituutta venyttämällä varausta. Lisäksi tarpeena oli, että vuoroja ja varauksia olisi mahdollista siirrellä eri päivästä ja ajankohdasta toiseen. Näitä ominaisuuksia tarjoava avoimen lähdekoodin JavaScript-kirjasto oli jo olemassa, joten oli järkevää ottaa se pohjaksi kalenterien toiminnalle. Kalenterinäkymät tehtiin Reactilla, sillä kyseinen teknologia on ollut yrityksessä aiemminkin käytössä.

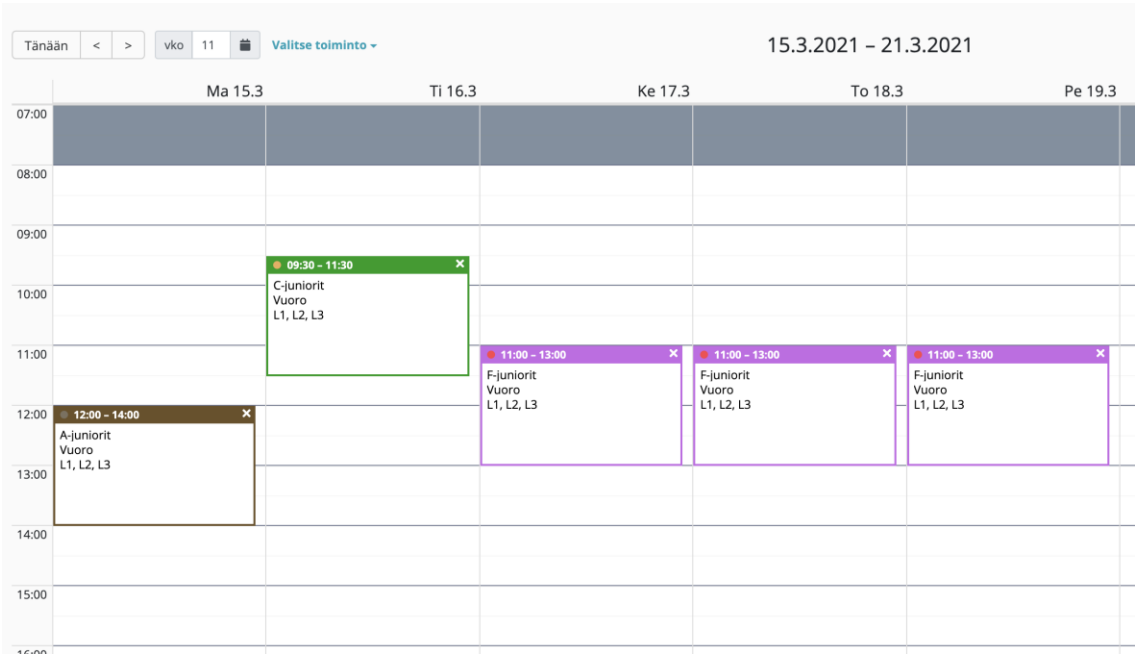
Ensimmäisenä kehitettiin varauksien kalenteri, jossa varauksia voidaan tehdä ja suunnitella viikkonäkymässä. Koodiesimerkissä 8 on esitetty kalenterikomponentti. Kuten esimerkkikoodista 8 nähdään, kalenterin komponentti vaati paljon propseja, joiden avulla voidaan hallita, mitä tapahtuu kunkin käyttäjän toiminnon seurauksena.

```
<DragAndDropCalendar
  selectable
  localizer={localizer}
  events={events}
  slotPropGetter={slotPropGetter}
  eventPropGetter={eventPropGetter}
  startAccessor="start"
  endAccessor="end"
  view={calendar.defaultView}
  views={['month', 'week', 'day']}
  date={calendar.weekDate}
  popup={false}
  min={new Date(2018, 0, 1, display.minTime, 0)}
  max={new Date(2018, 0, 1, display.maxTime, 59)}
  formats={calendarFormats}
  messages={Object.assign(I18n.t('calendar.messages'), { showMore: showMore })}
  onSelectSlot={({ start, end }) => handleEventChanges(null, start, end)}
  onSelectEvent={e => handleEventSelect(e)}
  onDoubleClickEvent={e => handleEventDoubleClick(e)}
  onEventResize={({ event, start, end }) => handleEventChanges(event, start, end)}
  onEventDrop={({ event, start, end }) => handleEventChanges(event, start, end)}
  onNavigate={(date, view) => onNavigate(date, view)}
  onView={view => handleViewChange(view)}
  onDrillDown={date => handleViewChange('day', date)}
  components={{
    month: { event: MonthReservation },
    event: (orgProps) => <Reservation {...orgProps} deleteEvent={deleteEvent} />,
    toolbar: (orgProps) => <CalendarToolbar {...orgProps} venue={props.venue} />,
    dateHeader: DateHeader,
    dateCellWrapper: DateCellWrapper
  }}
/>
```

Esimerkkikoodi 8. Varausnäkymän kalenterikomponentti.

Kalenterille on myös annettava localizer-objekti, joka hoitaa komponentin käännöksien ja lokalisaation tarpeet. Kalenterikomponentti vaati localizer-objektin, jotta kalenterissa näytettävät tekstit osataan kääntää sovelluksen kielen mukaan ja kellonajat ilmoitetaan oikeassa muodossa. Events-propsi sisältää listan varausobjekteista, joita kalenteri näyttää. Components-propsin avulla kalenterin omia komponentteja voi korvata itse tehdyllä komponenteilla. Tämä on usein tarpeellista, kun halutaan kustomoituja toimintoja, joita kalenteri ei suoraan tue, tai jos halutaan muokata sitä, miltä kalenterin eri elementit näyttävät. Esimerkkikoodista 8 nähdään, että varauskalenterille on tehty omat komponentit usealle eri osalle kalenteria. Kalenterin tärkein kustomoitu komponentti on yhden varauksen komponentti, jota klikkaamalla voidaan avata varauksen lisätiedot ja muokata varauksen eri arvoja omassa modaalissa. Kustomoitu varauksen komponentti mahdollistaa

myös, että sen ulkonäköä voidaan muokata tarpeen mukaan. Kuvassa 6 on kalenterin viikkonäkymä, jossa on useampi varaus tehtynä.



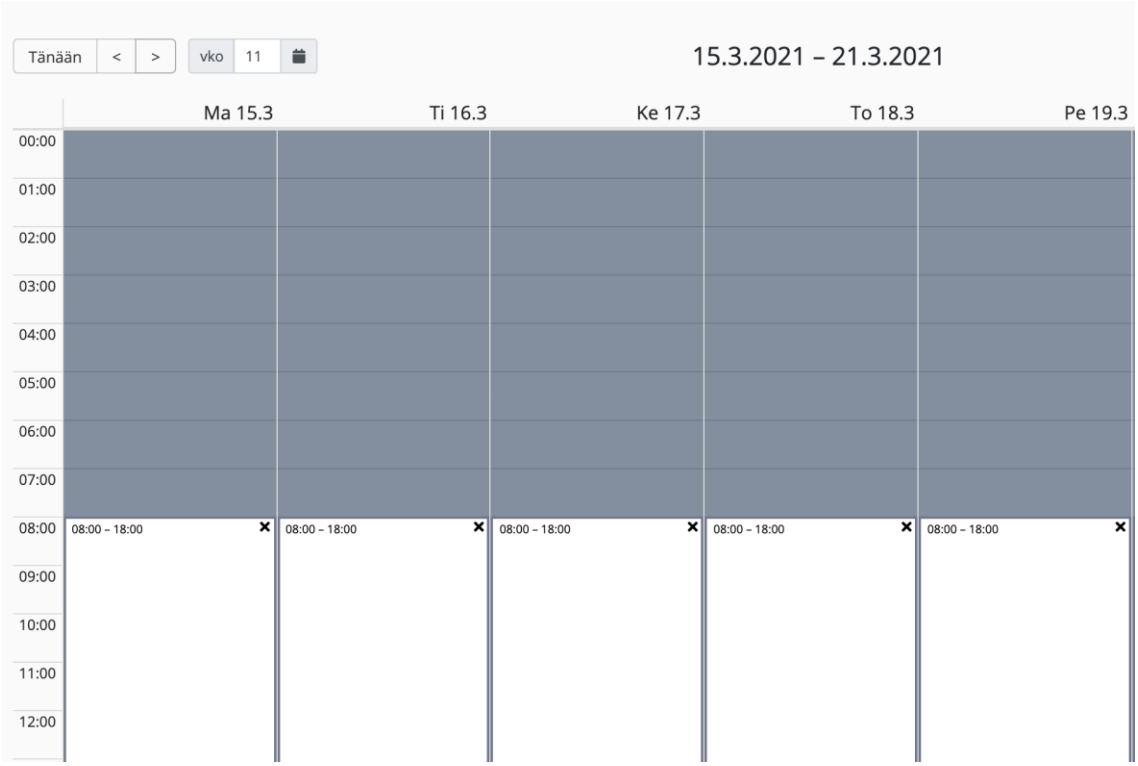

Kuva 6. Esimerkkikuva varauksien viikkonäkymästä.

Kuvassa 6 on kolmen eri ryhmän varauksia, jotka ovat samassa tapahtumapaikassa, esimerkiksi jäähallissa. Jokainen varaus on oma React-komponenttinsa, joka saa propseina varauksen tiedot, jotka voidaan näyttää käyttäjälle. Varauksen tyylit saadaan kalenterinkomponentin kautta eventPropGetter-funktion avulla, joka palauttaa tyylit ja komponentin luokan nimen varauskomponentille. Funktiossa tarkistetaan muun muassa varauksen ryhmän väri ja määritellään kyseinen väri varauksen reunojen ja yläpalkin väriksi, jos varausta ei ole viety myClubiin. MyClubiin viedyt varaukset väritetään kokonaan ryhmän värillä ja varauksen yläpalkki väritetään hieman tummemmalla värillä kuin ryhmän väri, jotta myClubiin viety varaus erottuu niistä varauksista, joita ei ole viety myClubiin. Varauksen sisällön tekstin väriin käytetään funktiota, joka vertaa varauksen ryhmän väriä tekstin väriin ja määrittää värin, joka paremmin erottuu varauksen taustaväristä (ryhmän väristä tai valkoinen). Esimerkiksi, jos ryhmän väri on tumman harmaa, on tekstin väri valkoinen. Mutta jos ryhmän väri on vaalean harmaa, on tekstin väri musta.

Jokainen varauksessa tietoa kuvaava kenttä on oma React-komponenttinsa. Oletuskenttiä ovat tapahtumatyypin väri ja aikaväli, ryhmä, nimi (varauksen nimi), lohkot (sama tapahtumapaikka on jaettu useampaan eri osaan) ja ohjaajat. Yksi varaus koostuu siis useammasta kentästä, kuten kuva 6 havainnollistaa. Varauksen kentät näkyvät oletusarvoisesti yllä kuvatussa järjestyksessä. Näin ollen ylimpänä, ensimmäisenä kenttänä, on oletusarvoisesti tieto tapahtumatyypistä värin avulla (pieni ympyrä varauksen vasemmassa ylänurkassa) sekä varauksen aikaväli. Varauskalenteri saa propsina tiedon varauksen kentistä ja niiden järjestyksestä, jotka voidaan varauksen komponentissa renderöidä missä tahansa järjestyksessä. Kenttien järjestystä on mahdollista muokata varauksien asetuksista. Näin ollen ensimmäiseksi kentäksi voidaan esimerkiksi määrittää tapahtumatyypin ja aikavälin sijaan ryhmän nimi. On myös mahdollista poistaa kenttiä näkyvistä kokonaan. Palaan tähän aiheeseen hieman tarkemmin vielä edempänä alaluvussa 6.5.

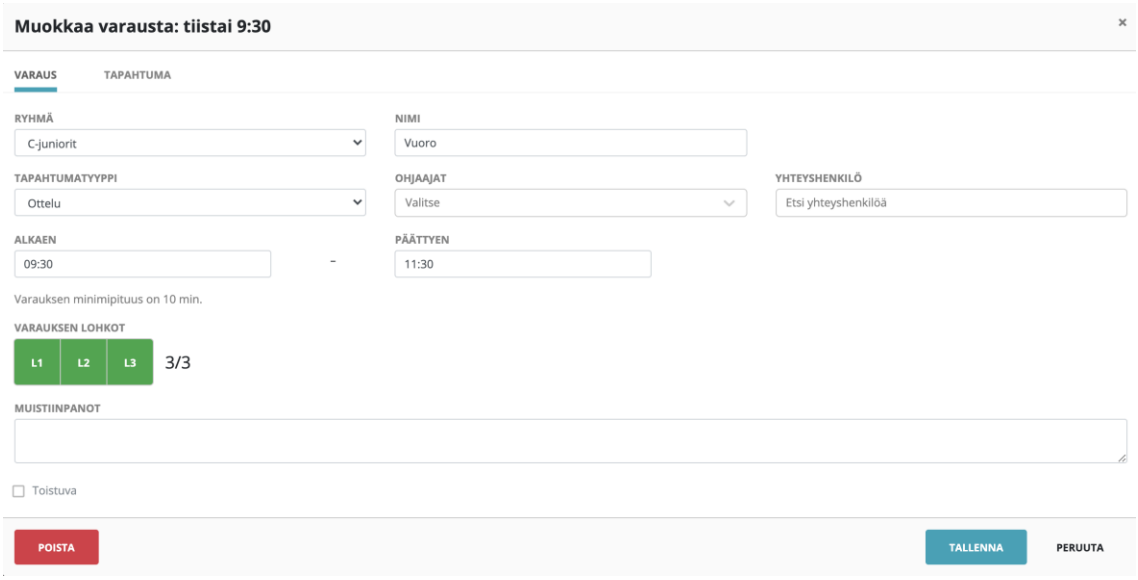
Valkoinen väri, joka on esitetty varauksien viikkonäkymässä (kuva 6), kuvaa varattavissa olevia vuoroja, joita Vuorosuunnittelu-palvelun käyttäjien on mahdollista varata. Vuoroja voidaan tehdä omassa näkymässään (kuva 7), ja ne näkyvät varausten näkymässä (kuva 6) valkoisina alueina, jotta varauksia ei vahingossa tule tehtyä ajankohdalle, jossa ei ole vuoroa (siniharmaa). Tämän tekninen toteutus onnistuu siten, että kalenterikomponentti hakee tiedot tehdyistä vuoroista API:n kautta, jonka jälkeen varatut vuorojen aikavälit piirretään eri värisiksi (valkoinen) muun kalenterin kanssa (siniharmaa). Sekä vuorot, että varaukset haetaan API:n kautta, aina tarvittaessa, esimerkiksi silloin, kun vaihdetaan seuraavaan viikkoon tai eri näkymään.

Kuvasta 7 nähdään, että vuorojen näkymä muistuttaa paljolti varausten näkymää, mutta vuoroilla ei ole ryhmiä tai muita arvoja asetettavina. Vuorot kuvaavatkin vain vapaita aikoja tapahtumapaikassa, johon voidaan tehdä varauksia. Vuoroja ei voi myöskään tehdä päällekkäin toisin kuin varauksia. Vuorojen tekninen toteutus on myös paljolti sama kuin varauksien, ja siihen on voitu käyttää samaa kalenterikomponenttia kuin varauksien kalenteriin. Vuorojen kalenteri kuitenkaan ei vaadi yhtä paljon propseja, koska sen toteutus on yksinkertaisempi.




Kuva 7. Esimerkkikuva vuorojen viikkonäkymästä.

Varauksia ja vuoroja voidaan muokata ottamalla muokattavasta varauksesta tai vuorosta kiinni ja vetämällä se halutulle paikalla. Varauksien ja vuorojen aikavälejä voi myös muokata vetämällä varausta hiirellä ylhäältä tai alhaalta ja venyttämällä se sopivan pituiseksi. Varauksien ja vuorojen aikavälien muokkaus laukaisee aina muokkausmodaalin, jossa käyttäjä voi hyväksyä muutoksen. Muokkausmodaalissa varauksien kaikkia arvoja voidaan helposti säätää. Kuvassa 8 on esitetty varauksen muokkausmodaali.




Kuva 8. Varauksen luomis- ja muokkausmodaali.

Varauksen muokkausmodaalissa varauksen ryhmää, nimeä, tapahtumatyyppiä, ohjaajia, yhteyshenkilöä, alkamisaikaa, loppumisaikaa, lohkoja sekä muistiinpanoja voidaan muokata. Vuorojen muokkaus tapahtuu samanlaisessa modaalissa kuin varauksienkin, mutta vuoroilla muokattavissa olevia arvoja ovat ainoastaan vuoron aloitusaika, päättymisaika ja toistuvuus. Varauksen ja vuoron voi asettaa myös toistuvaksi haluamallaan aikavälillä. Lisäksi tapahtuma-välilehdelle voidaan muokata myClub-tapahtumaan liittyviä tietoja, kuten tapahtuman ilmoittautumista ja näkyvyyttä. Kaikki yllämainitut tiedot kerätään reduxin stateen, josta ne lähetetään submitin yhteydessä palvelimelle API:n kautta, missä varaus luodaan tai sitä päivitetään. Yksittäisen kentän muokkaaminen tapahtuu kutsumalla handle-alkuista funktiota, joka kutsuu redux actionia, jonka kautta reducer voi päivittää sovelluksen tilaa. Seuraavana on annettu esimerkki, miten varauksen kentän muuttaminen tapahtuu. Esimerkkikoodissa 9 olevaa funktiota kutsutaan aina, kun käyttäjä kirjoittaa varaukselle nimeä, eli kun varauksen nimeä muutetaan.

```
handleTitleChange = event => {
  const title = event.target.value;
  this.props.changeTitle(title);
}
```

Esimerkkikoodi 9. Funktio, joka muuttaa sovelluksen tilaa, kun varauksen nimeä muutetaan.

## 6.4 Listanäkymä ja suodattimet

Varauksien ja vuorojen kalenterinäkymien valmistuttua siirryttiin varauksien ja vuorojen listanäkymään sekä suodattimiin. Listanäkymässä varauksia ja vuoroja voi tarkastella taulukossa ja niille voi suorittaa erilaisia toimintoja. Nämä toiminnot voivat kohdistua yhteen tai useampaan taulukon riviin. Tähän käytettiin myClubiin kehitettyä taulukkokomponenttia, joka sisälsi valmiudet näille ominaisuuksille. Taulukkokomponentin ja suodattimien vuoksi myClubista oli pakollista irrottaa kaikki uudelleenkäytettävät käyttöliittymän osat omaksi jaettavaksi kirjastokseen, millä vältettiin turhaa koodin toistamista ja tarvetta ylläpitää kahta erillistä kokonaisuutta.

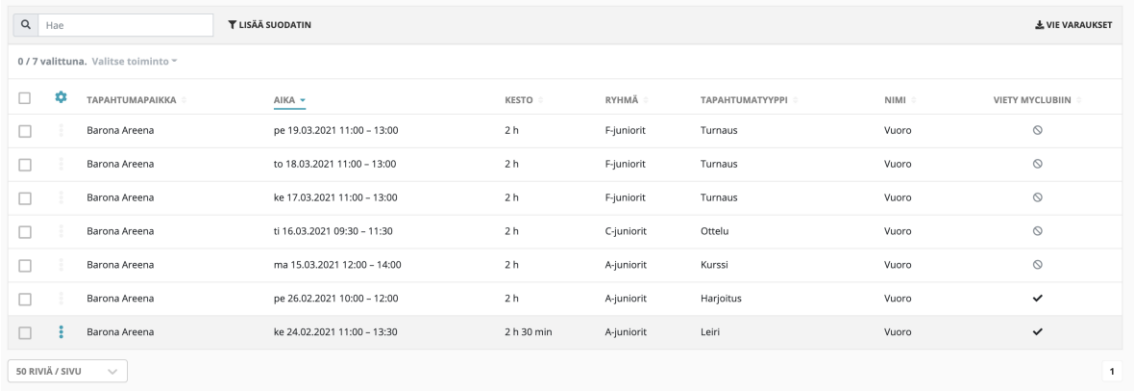
Irrottaminen toteutettiin tekemällä uudelleenkäytettävistä React-komponenteista oma JavaScript-paketti, joka oli mahdollista ottaa käyttöön Vuorosuunnittelussa käyttäen npm-pakettienhallintatyökalua Node.js:lle. Taulukkokomponentti ja suodattimet vaativat myös tietynlaisen palvelinpuolen toteutuksen ja omat tyylinsä, joten niistä oli järkevää tehdä uudelleenkäytettävä Ruby-paketti eli gemi. Gemit ovat paketteja, jotka sisältävät Ruby-koodia, jota voidaan suoraan käyttää eri projekteissa lisäämälle ne projektin Gemfile-tiedostoon. Gemfile-tiedosto sisältää Ruby on Rails-projektin kaikki käytössä olevat gemit. Näiden muutoksien jälkeen taulukkokomponentti oli mahdollista ottaa käyttöön Vuorosuunnittelussa. Esimerkkikoodissa 10 on esimerkki siitä, miten taulukomponentti voidaan sisällyttää Ruby-koodilla upotettuun HTML-tiedostoon.

```
<%= react_component 'McTable',
  ReservationListPresenter.new(
      params,
      @reservations,
      self
    ) %>
```

Esimerkkikoodi 10. React-taulukomponentti upotetussa Ruby-koodissa.

Taulukko-komponentti ei vaadi kuin yhden Ruby-luokan propsina, joka tuottaa taululle sopivassa muodossa datan, jota taulukko pystyy käyttämään ja näyttämään käyttäjälle.

Tämä data sisältää tiedon varauksista, näytettävistä kentistä, suodattimista sekä toiminnoista, joita taulukosta voidaan tehdä. Kuvassa 9 on esitetty varauksien listanäkymä.




Kuva 9. Varauksien listanäkymä.

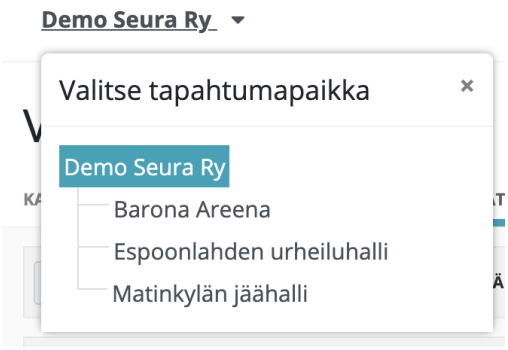
Kuvasta 9 nähdään, että taulukon rivejä on mahdollista suodattaa ja järjestää halutun arvon mukaan. Suodattimet mahdollistavat eri toimintojen tekemisen taulujen riveille tarkoilla säännöillä. Suodattimien avulla on esimerkiksi mahdollista julkaista myClubiin vain tietyn ryhmän varaukset. Suodattimia on myös mahdollista tallentaa suosikeiksi, jolloin eniten käytetyt suodattimet ovat aina nopeasti saatavilla. Suodattimet toimivat kaikissa näkymissä, joissa taulukkokomponenttia käytetään. Näitä ovat kalenteri-, lista- ja yhteenvetonäkymät.

## 6.5 Asetukset ja integraatio myClubin kanssa

Asetuksissa seura voi määrittää itselleen sopivia oletusarvoja sekä hakea tarvittavia tietoja, kuten ryhmiä, myClubista. Vuorosuunnittelu vaati toimiakseen myClubista tietoja, jotta varauksia voidaan viedä myClubiin tapahtumiksi. Tätä varten myClubista on mahdollista hakea API:n kautta tapahtumapaikkoja, ryhmiä, tapahtumatyyppejä ja muita tietoja. Vuorosuunnittelun pääkäyttäjät voivat hakea tietoja asetukset osiossa, jossa käyttäjä voi valita, mitkä tiedot myClubista tuodaan Vuorosuunnitteluun.

Tapahtumapaikat ovat paikkoja, joihin vuoroja ja varauksia voi tehdä. Vuorosuunnittelu vaatii toimiakseen vähintään yhden tapahtumapaikan, jotta vuoroja ja varauksia voi alkaa suunnittelemaan. Tapahtumapaikat ovat seuran määrittelemiä paikkoja, jotka on

tehty myClubissa. Vuorosuunnittelussa tapahtumapaikkoja voi hakea myClub-asetuksien alta, jossa käyttäjä voi itse valita, mitkä tapahtumapaikat tallennetaan Vuorosuunnitteluun. Tapahtumapaikkojen haku on toteutettu API:n kautta, jossa kaikki tapahtumapaikat haetaan myClubista valittavaksi. Kun Vuorosuunnittelupalvelun käyttäjä on valinnut kaikista tapahtumapaikoista haluamansa tapahtumapaikat, tallennetaan ne valinnan jälkeen tietokantaan. API vaatii toimiakseen API-avaimen, joka luodaan seuralle Vuorosuunnittelun tilauksen yhteydessä. Sekä vuorojen että varauksien näkymät kuvaavat aina yhden tapahtumapaikan tilannetta. Tilannetta voi myös katsoa usean tapahtumapaikan näkymästä, jolloin käyttäjä saa kuvan kaikkien tilojen kokonaistilanteesta. Näitä näkymiä voidaan hallita tapahtumapaikan valintakomponentilla, josta on esimerkki kuvassa 10.




Kuva 10. Tapahtumapaikan valitsin.

Kuvassa 10 valittuna näytetään seuran nimi, jolloin kaikki seuran alla olevat tapahtumapaikat ovat katseltavissa kerrallaan allekkain aseteltuna. Valittaessa tietty tapahtumapaikka seuran alapuolelta, voidaan tarkastella tätä tapahtumapaikkaa yksitellen. Valitusta tapahtumapaikasta pidetään tieto tallessa sessiossa, ja eri näkymissä näytetään varausten ja vuorojen tietoja valitun tapahtumapaikan perusteella.

Tapahtumapaikan lisäksi varauksella on oltava ryhmä, jotta se voidaan viedä myClubiin tapahtumana, sillä tapahtumat kohdistavat myClubissa aina ryhmään. Ryhmiä voidaan hakea vastaavasti myClubin asetukset-osiossa, jossa käyttäjä voi valita, mitkä ryhmät tuodaan myClubiin. Ryhmälle voi myös asettaa oman värin, joka näytetään varauksen värinä kalenterissa, kun ryhmä on valittu varaukselle. Eriväriset varaukset erottuvat paremmin toisistaan varsinkin silloin, kun varaukset ovat päällekkäisiä, eli ne on tehty samalle aikavälille.

Tapahtumatyyppejä on myös mahdollista hakea myClubista. Tapahtumatyypeille voi määritellä myClub-tapahtuman aloitusajan ennen varausta, lopetusajan varauksen jälkeen sekä näkyvyyden ja ilmoittautumisen oletusarvot. Varaus saa nämä arvot automaattisesti, kun varaukselle valitaan tämä tapahtumatyyppi. Varauksen React-komponentti saa serialisoidun tiedon mahdollisista tapahtumatyypeistä ja niiden oletusarvoista. Tapahtumatyypin vaihdon actionin seurauksena varaus saa valitulle tapahtumatyypille määritetyt oletusarvot. Seuralle voidaan myös antaa samat oletusarvot kuin tapahtumatyypille, mutta tämän lisäksi myös varauksen nimi. Näitä oletusarvoja käytetään varaukselle, jos tapahtumatyyppiä ei ole valittu.

# 7 Yhteenveto

Insinöörityön tavoitteena oli kehittää hyvin toimiva, vuoronvarausarkea helpottava työkalu seuran tapahtumapaikkojen varausten ja vuorojen hallintaan. Vuorosuunnittelu kehitettiin lisäpalveluksi olemassa olevaan myClub-palveluun.

Vuorosuunnittelu-lisäpalvelun kehitys oli monimutkainen ja aikaa vievä prosessi, jossa oli paljon huomioitavaa ja suunniteltavaa. Hyvän lopputuloksen saamiseksi oli oltava hyvä suunnitelma. Suunnitelmassa oli otettava huomioon käyttöliittymä, käytettävyys, resurssit sekä palvelun ominaisuudet ja hyödyt käyttäjille. Oli myös varmistettava, että palvelulle on todellinen tarve ja etteivät markkinat ole liian saturoituneet. Hyvin suunniteltu on kuitenkin vasta puoliksi tehty. Kehitysaika oli vuorosuunnitteluprojektin ylivoimaisesti suurin aikaa vievä osa.

On olemassa monia erilaisia vuoronvarausjärjestelmiä, mutta seurojen käyttöön tarkoitettuja palveluita, jotka sisältäisivät integraation myClubin kanssa, ei ole olemassa. Sähköiset ajanvarausjärjestelmät ovat yleisesti käytössä monilla toimialoilla ja kaupungeilla. Sähköiset varausjärjestelmät tuovat nopeuden ja seurattavuuden kaikkien saataville. Mahdollisuus laskutukseen palveluiden kautta tarjoaa helpon tavan pitää talous kunnossa.

myClub on seurojen jäsenpalvelu, joka tuo helppouden seuran jäsenrekisterin hoitamiseen, tapahtumiin ilmoittautumiseen, viestintään sekä laskutukseen. myClub on yksi palvelu, joka sisältää kaiken seurojen tarpeisiin. Lisäpalveluiden avulla seura voi laajentaa

myClubin toimintaa myös integraatioilla muihin sovelluksiin, esimerkiksi laskutukseen liittyen. Vuorosuunnittelu tuli osaksi muita tarjolla olevia lisäpalveluita.

Vuorosuunnittelun kehityksessä käytettiin Ruby on Rails -ohjelmistokehystä sovelluksen näkymien sekä palvelimen puolen toteutukseen. Ruby on Railsin avulla sovelluksen yksinkertaisien näkymien ja lomakkeiden tekeminen oli helppoa ja nopeaa. Ruby ja Ruby on Rails olivat itselleni täysin uusia, mutta niiden perusteiden opetteleminen ei vienyt kauaa, vaikka niiden täydellinen ymmärtäminen on pidempi prosessi. Vuorosuunnittelun käyttöliittymän monimutkaisempiin osiin käytettiin React JavaScript -kirjastoa sekä sovelluksen tilan hallitsemisen Redux-kirjastoa.

Vuorosuunnitteluun toteutettiin mahdollisuus hakea tapahtumapaikkoja, ryhmiä ja tapahtumatyyppejä myClubista. myClubin asetuksissa on mahdollista antaa erilaisia oletusarvoja niin seura-, tapahtumatyyppi- kuin varauskohtaisesti. Tapahtumapaikoille kehitettiin kalenteri, lista- ja yhteenvetonäkymät, jossa voidaan luoda, muokata ja tarkastella vuoroja ja varauksia ja viedä niitä tapahtumina myClubiin.

Insinöörityön tuloksena saatiin tavoitteita vastaava monipuolinen vuoronvarausjärjestelmä, jossa on kaikki toivotut ominaisuudet. Projekti oli hyvin pitkä, ja siihen kuului useita osakokonaisuuksia, joista kaikkia ei ole mainittu tässä insinöörityössä niiden laajuuden vuoksi. Näitä ominaisuuksia ovat muun muassa varauksien ja vuorojen tuonti Excel-taulukosta, varauskalenterin upotukset, varauksien vienti Excel-tiedostoon sekä varauksien laskutus. Vuorosuunnittelussa on runsaasti mahdollisuuksia jatkokehitykselle ja palvelua kehitetään aktiivisesti myös jatkossa myClub-palvelun rinnalla.

## Lähteet

About Ruby 2021 Verkkoaineisto <https://www.ruby-lang.org/en/about/> Luettu helmikuussa 2021.

A brief overview of Bitbucket 2021 Verkkoaineisto <https://bitbucket.org/product/guides/getting-started/overview> Luettu maaliskuussa 2021.

Asio Varausjärjestelmä 2021 Verkkoainesto <https://www.asio.fi/> Luettu maaliskuussa 2021.

Christian Ström 2019 React pähkinänkuoressa Verkkoaineisto <https://joinex.fi/react-pahkinankuoressa/> Luettu helmikuussa 2021.

Components and Props 2021 Verkkoaineisto <https://reactjs.org/docs/components-and-props.html#function-and-class-components> Luettu helmikuussa 2021.

Fortune Ikechi 2020 How Redux Recucers Work Verkkoaineisto <https://www.smashingmagazine.com/2020/12/how-redux-reducers-work/> Luettu helmikuussa 2021.

Getting Started 2021 Verkkoaineisto <https://code.visualstudio.com/docs> Luettu maaliskuussa 2021.

Git 2020 Verkkoaineisto <https://www.linux.fi/wiki/Git> Luettu maaliskuussa 2021.

Joonas Virtanen 2016 Mistä muodostuu loistava käyttökokemus eli User Experience (UX)? Verkkoaineisto <https://contrast.fi/blog/hyvan-kayttokokemuksen-ux-kolme-tarkeinta-elementtia> Luettu helmikuussa 2021.

myClub ominaisuudet 2021 Verkkoaineisto <https://www.myclub.fi/ominaisuudet/> Luettu tammikuussa 2021.

Neo Ighodora 2020 Why use Redux? A tutorial with examples Verkkoaineisto <https://blog.logrocket.com/why-use-redux-reasons-with-clear-examples-d21bffd5835/> Luettu helmikuussa 2021.

Node.js 2021 Verkkoaineisto <https://fi.wikipedia.org/wiki/Node.js> Luettu maaliskuussa 2021.

Rails has everything you need 2021 Verkkoaineisto <https://rubyonrails.org/everything-you-need/> Luettu helmikuussa 2021.

Redux (JavaScript Library) 2021 Verkkoaineisto https://en.wikipedia.org/wiki/Redux_(JavaScript_library) Luettu helmikuussa 2021.

Rowena Gungon 2021 What is an online reservation system? Verkkoaineisto < https://www.getomnify.com/blog/one-reservation-system> Luettu helmikuussa 2021.

Ruby on Rails 2021 Verkkoaineisto <https://en.wikipedia.org/wiki/Ruby_on_Rails> Luettu helmikuussa 2021.

Ruby Version Manager (RVM) 2017 Verkkoaineisto <https://rvm.io/> Luettu maaliskuussa 2021.

Tietoa varaamo.hel.fi –palvelusta 2021 Verkkoainesto <https://varaamo.hel.fi/about> Luettu maaliskuussa 2021.

TIOBE Index for January 2021 Verkkoaineisto <https://www.tiobe.com/tiobe-index//> Luettu helmikuussa 2021.

TIMMI Tilavarausohjelmisto 2021 Verkkoaineisto <http://timmi.fi/tuotteet/timmi-tilavarausohjelmisto/> Luettu helmikuussa 2021.

Visual Studio Code 2021 Verkkoaineisto <https://en.wikipedia.org/wiki/Visual_Studio_Code> Luettu helmikuussa 2021.

Whats Rails 2021 Verkkoaineisto <https://github.com/rails/rails> Luettu helmikuussa 2021.

Yarn Introduction 2021 Verkkoaineisto <https://yarnpkg.com/> Luettu maaliskuussa 2021.