

Opinnäytetyö (AMK)

Tieto- ja viestintätekniikka

2023

Jimi Keurulainen

Varastohallintajärjestelmän kehitys MEAN-pinolla



Opinnäytetyö (AMK) | Tiivistelmä

Turun ammattikorkeakoulu

Tieto- ja viestintätekniikka

2023 | 48 sivua

Jimi Keurulainen

Varastohallintajärjestelmän kehitys MEAN-pinolla

Tämä päiväkirjamuotoinen opinnäytetyö kuvaa web-kehittäjän päivittäistä varastohallintajärjestelmän jatkokehittämistä salolaiselle metallialan yritykselle nimeltään Jame-Shaft Oy. Tämän opinnäytetyön tavoitteena oli tutustua olemassa olevaan varastohallintajärjestelmään ja kehittää siihen Jame-Shaftin vaatimat ominaisuudet. Muita Jame-Shaftin sisäisiä sovelluksia kehitettiin myös tämän opinnäytetyöjakson aikana. Jakson pituus oli 7 viikkoa, ja työsuhte jatkui tämän jälkeen vielä 2 viikkoa.

Entinen varastohallintajärjestelmä oli MEAN-pinolla (MongoDB, Express.js, Angular, Node.js) kehitetty verkkosovellus lukuun ottamatta tietokantaa.

Tietokantana projektissa oli käytetty Googlen Firebase-verkkotietokantaa. Uusi varastohallintajärjestelmä kehitettiin myös näillä samoilla teknologioilla, ja opinnäytetyön tavoite saavutettiin oppimalla ja soveltamalla olemassa olevaa tietoa näistä teknologioista.

Kaikki työnantajan pyytämät ominaisuudet kehitettiin onnistuneesti tarkastelujakson loppuun mennessä. Näiden myötä varastohallintajärjestelmä pystyttiin siirtämään tuotantoon, ja järjestelmä tehosti tuotannossa käsiteltävien töiden siirtämistä eri työvaiheiden välillä.

Asiasanat:

MEAN-pino, web-kehitys, Firebase

Bachelor's Thesis | Abstract

Turku University of Applied Sciences

Information and Communication Technologies

2023 | 48 pages

Jimi Keurulainen

Development of a Warehouse Management System With MEAN-Stack

This diary-based thesis depicts a web developer's day-to-day tasks in further developing an already existing warehouse management system for a Salo-based metal industry company called Jame-Shaft Oy. This thesis' goal was to familiarise with the existing warehouse management system and develop features that Jame-Shaft required. Other Jame-Shaft's internal software was also developed during this thesis' reporting period. This period lasted for 7 weeks, and the employment period lasted for 2 weeks afterwards.

The previous storage management system had been developed with MEAN-stack (MongoDB, Express.js, Angular, Node.js), excluding the database, which was Google's Firebase. The new system was developed with the same technologies that were used in the old system, and the goal of this thesis was achieved by learning and applying existing knowledge about these technologies.

All the employer's requested features to the warehouse management system were developed successfully by the end of this thesis' duration. The new warehouse management system was afterwards moved to production, which enhanced the flow of products between different work stages.

Keywords:

MEAN-stack, web-development, Firebase

Sisältö

Käytetyt lyhenteet ja sanasto	6
1 Johdanto	9
2 Nykytilanteen kuvaus	10
2.1 Varastohallintajärjestelmän alkuperäinen versio	10
2.2 Pääasialliset kehityskohteet	11
2.3 Sidosryhmät ja vuorovaikutus työpaikalla	11
3 Toiminnan raportti	13
3.1 Ensimmäinen raportointiviikko	13
3.2 Toinen raportointiviikko	18
3.3 Kolmas raportointiviikko	23
3.4 Neljäs raportointiviikko	27
3.5 Viides raportointiviikko	31
3.6 Kuudes raportointiviikko	35
3.7 Seitsemäs raportointiviikko	38
3.8 Kahdeksas raportointiviikko	41
3.9 Yhdeksäs raportointiviikko	43
4 Pohdinta	45
Lähteet	47

Kuvat

Kuva 1. Kuvakaappaus varastohallintajärjestelmän lavapaikkahallinta-sivulta.	17
Kuva 2. Automaattiajojen seurantasovelluksen käyttöliittymä.	19
Kuva 3. Ohjeet-sivun uusi käyttöliittymä.	25
Kuva 4. Ohjeet-sivun uusin käyttöliittymän versio.	40

Käytetyt lyhenteet ja sanasto

Angular	TypeScriptillä kirjoitettava ja komponentteihin perustuva modulaarinen verkkosivujen selainpuolen kehityskirjasto.
API	API eli rajapinta (Application Programming Interface) on kahden tai useamman eri ohjelmiston välille luotu väylä, jonka kautta ne pystyvät kommunikoimaan toistensa kanssa.
Augury	Selainlisäosa, jolla pystyy analysoimaan ja visualisoimaan Angularilla kehitettyjä verkkosovelluksia.
Bitbucket	Git-versionhallintajärjestelmään pohjautuva ja Atlassianin omistama versionhallintajärjestelmä.
Blob	Tietorakenne, jolla pystyy manipuloimaan tiedostojen raakaa dataa. Juontuu termistä iso binääriolio (a binary large object).
Express.js	Node.js:lle suunniteltu verkkosovelluskehys, jolla pystyy kehittämään esimerkiksi verkkosovelluksen API-kutsuja.
Ffmpeg	Komentoriviltä käytettävä työkalu, jolla pystyy muun muassa purkamaan ja muuntamaan kaikenlaisia multimediatiedostoja.
FileSystem	Node.js-suoritusympäristöön kuuluva moduuli, jonka avulla pystyy käsittelemään laitteen tiedostoja.
Firebase	Googlen kehittämä ja ylläpitämä verkkotietokanta.
Fragmentointi	Tiedoston pirstaloiminen tai jakaminen pienempiin osioihin. Esimerkiksi videotiedostoja voidaan fragmentoida, jotta näitä videoita pystytään suoratoistamaan.
Git	Yleisesti käytetty, hajautettu versionhallintajärjestelmä.

Hahmontaminen	Digitaalisen elementin, esimerkiksi kolmiulotteisen kappaleen tai verkkosivun elementin, näytölle kuvastaminen tai luominen.
ID	Yksilöivä tunniste (unique identifier), jota voidaan käyttää erottamaan esimerkiksi listojen eri alkiot toisistaan.
Kanban	Suosittu projektinhallinnan projektikehys, jossa työtehtävät on jaoteltu omiksi korteikseen, jotka sitten asetetaan eri työvaiheisiin.
MEAN	Yhdistelmä teknologioita, jolla pystyy kehittämään laajoja verkkosovelluksia. Lyhenne sanoista MongoDB, Express.js, Angular, Node.js.
Microsoft Teams	Microsoftin kehittämä yhteistyö- ja viestintäalusta.
MongoDB	Dokumenttipohjainen tietokanta, jossa jokainen tietokannan rivi muistuttaa rakenteeltaan JSON-oliota.
MP4	Yksi videoiden tiedostomuodoista. Se pystyy sisältämään videokuvaa, ääntä sekä tekstityksiä.
MS SQL	Microsoftin kehittämä relaatiotietokanta, jonka pääasiallinen tavoite on tallentaa ja jakaa tietoa paikallisverkon eri sovelluksien välillä.
MVC-arkkitehtuuri	Ohjelmistoarkkitehtuuri, jossa erotetaan tietokanta, tietojen käsittely ja sovelluksen selainpuoli eri osioikseen. MVC on lyhenne sanoista malli, näkymä ja käsittelijä (Model, View, Controller).
Node.js	Yleisesti verkkosovellusten taustana käytettävä JavaScriptiä käyttävä ajoympäristö.
npm	JavaScript-kielellä kirjoitettujen lisäosien eli "pakettien" hallintajärjestelmä.
Olio-URL	Selaimen tiedostosta luoma URL-osoite, jonka kautta tiedostoa pystyy katselemaan selaimessa.

QR-koodi	Kaksiulotteinen kuviokoodi, johon on koodattu tietoa. Tätä tietoa luetaan älylaitteen kameralla.
React	Verkkosovellusten selainpuolen kehittämiseen tarkoitettu komponenttipohjainen ja vapaan lähdekoodin JavaScript-kirjasto.
RxJS	Reaktiivisten lisäosien kirjasto JavaScriptille (Reactive Extensions Library for JavaScript).
Scope creep	Projekteissa ilmenevä ongelma, jossa projektin laajuus kasvaa liian suureksi uusien ominaisuuksien ja lisäysten myötä.
Socket.io	Node.js:lle kehitetty kaksisuuntainen kommunikaatiotyökalu, millä pystyy muun muassa ilmoittamaan toisille verkkosivun käyttäjien selaimille yhden käyttäjän tekemistä muutoksista.
TypeScript	JavaScriptiin perustuva ohjelmointikieli, joka eroaa JavaScriptistä vahvemmallalla tyyppityksellä sekä tiukemmalla tietorakenteella.
URL	Verkko-osoite, jolla selain pystyy löytämään sisältöä internetistä, kuten verkkosivuja. Lyhenne sanoista Uniform Resource Locator.
UUID	128-bittinen tunnistus, jota käytetään yleisesti yksilöivänä tunnisteenä. Lyhenne sanoista Universally Unique Identifier.
VPN	Virtuaalinen yksityisverkko (Virtual Private Network), jonka avulla laitteet pystyvät olemaan yhteydessä toisiinsa samoin kuin paikallisverkossa.

1 Johdanto

MEAN-pinolla tarkoitetaan verkkokehityksessä suosittua yhdistelmää eri teknologioita, jolla voi kehittää laajoja verkkosovelluksia. Termi MEAN on lyhenne sanoista MongoDB (tietokanta), Express.js (palvelimen toteutusympäristö), Angular (selainpuolen ohjelmistoympäristö) ja Node.js (suoritusympäristö).

Tulen tämän opinnäytetyön aikana kehittämään jo olemassa olevaa varastohallintajärjestelmää Jame-Shaft Oy -nimiselle metallialan yritykselle Salossa. Varastohallintajärjestelmä on kehitetty MEAN-pinolla lukuun ottamatta tietokantaa. Tietokantana projektissa on käytetty Googlen Firebase-verkkotietokantaa. Varastohallintajärjestelmän kehityksessä ei tarvita kovinkaan paljon MEAN-pinon taustaohjelmistoon tarkoitettuja teknologioita, koska Firebase-tietokanta on suorassa yhteydessä varastohallintajärjestelmän Angularilla tehtyyn selainpuoleen. Tulen kuitenkin tarvitsemaan näitä teknologioita myöhemmin projektissa varastohallintajärjestelmän ohjevideoiden lisäämisessä.

Tarkastelujakson aikana perehdyn tähän varastohallintajärjestelmään ja kehittämään siihen työnantajan pyytämät ominaisuudet. Teen myös tarvittaessa muita työnantajan pyytämiä kehitystöitä muiden yrityksen sisäisten sovellusten parissa.

Tavoitteena on saada tarkastelujakson päätteeksi valmiiksi kaikki työnantajan määrittelemät kehityskohteet.

2 Nykytilanteen kuvaus

Tarkastelujakson alussa aikaisempi varastohallintajärjestelmä on otettu pois käytöstä työntekijöiden ilmoittamien puutteiden ja ongelmien takia. Jotkin ominaisuudet ovat myös jääneet kesken edelliseltä kehitysjaksolta, joita on tarkoitus kehittää valmiiksi tällä tarkastelujaksolla. Tarkastelujakson alussa työnantaja ei ole vielä määritellyt kaikkia tarkastelujakson kehitystavoitteita. Alustavat tavoitteet ovat varastohallintajärjestelmän teknologioiden päivitys uudempiin versioihin sekä varastohallintajärjestelmän töiden lavapaikkojen valitsemiseen tarkoitetun pudotusvalikon uudelleentyylittely.

2.1 Varastohallintajärjestelmän alkuperäinen versio

Työpaikkaesittelyn myötä selvisi, että Jame-Shaft tuottaa monen eri alan yrityksille muun muassa holkkeja ja niveltappeja, ja Jame-Shaftin tuotanto tuottaa kappaleet käsittelemättömästä raakamateriaalista aina valmiiseen tuotteeseen asti. Nämä kappaleet kulkevat monen eri työvaiheen välillä, jotka sijaitsevat tuotantoalueen eri osissa. Varastohallintajärjestelmän tarkoitus on seurata kappaleiden kulkemista työvaiheelta toiselle ja hallita niiden väliaikaista tai pysyvää pysähdystä tuotantoalueen varastohyllyissä.

Alkuperäinen varastohallintajärjestelmä on Jame-Shaftin paikallisverkossa toimiva verkkosovellus. Sovellus jakaantuu kolmeen eri sivuun: lavaliiikenteeseen, taksijonoon sekä lavapaikkahallintaan. Uudet työt kirjataan ensin lavaliiikenteeseen, ja jokaiseen työhön kuuluu vähintään työnnumero sekä nykyinen lavapaikka. Jos työ on valmiina siirtoon toiselle lavapaikalle, siihen lisätään seuraava lavapaikka. Tällöin kyseinen työ ilmestyy varastohallintajärjestelmän taksijono-sivulle. Taksijono-sivulta trukkien kuljettajat pystyvät valitsemaan kuljetukseen työt nykyiseltä lavapaikaltaan seuraavaan lavapaikkaan. Lavapaikkahallinta-sivulta pystyy näkemään kaikki varastohallintajärjestelmässä olevat lavapaikat sekä hallinnoimaan niitä.

2.2 Pääasialliset kehityskohteet

Uusien töiden lisääminen varastohallintajärjestelmään tapahtuu lavaliikennesivulta sijaitsevasta ”Lisää työ” -nappulasta. Nappulaa painettaessa avautuu lomakkeen sisältävä ponnahdusikkuna, johon lisätään vähintään uuden työn numero ja nykyinen lavapaikka. Nykyisen lavapaikan valitsemiseen on tällä hetkellä pelkkä pudotusvalikko, jossa näkyy jokaisen lavapaikan nimi.

Lavapaikan nimen ja sen fyysisen sijainnin korrelaatio ei ole kuitenkaan kovinkaan intuitiivinen, joten tavoitteena on tehdä tästä pudotusvalikosta jollain tavalla helppolukuisempi.

Lavapaikkahallinta-sivulla näkee kaikki olemassa olevat lavapaikat ruudukkona. Tällainen ruudukko kävisi myös lavapaikan valitsemiseen.

Muita alustavia kehityskohteita on eri käyttäjien lisääminen varastohallintajärjestelmään, varastohallintajärjestelmän käyttöohjeiden lisääminen sivustolle sekä jonkinlainen tietokannan paikallinen varmuuskopiointijärjestelmä.

2.3 Sidosryhmät ja vuorovaikutus työpaikalla

Minä tulen kehittämään varastohallintajärjestelmää yksin. Tulen työsuhteeni aikana käymään toimistolla kerran viikossa, ja muuten tulen työskentelemään etänä. Viikoittaisten toimistokäyntien aikana tulemme myös käymään läpi viikon etenemiseni työnantajani kanssa. Työnantajani tulee lisäämään työtehtäviä Microsoft Teamsiin sitä mukaa kun hän saa niitä määriteltyä. Microsoft Teams on Microsoftin kehittämä yhteistyö- ja viestintäalusta. Minulle ei ole sovittu mitään muita tapaamisia tai palavereita muiden kollegoiden kanssa työsuhteen ajalle, joten tulen suurimmaksi osaksi työskentelemään itsenäisesti. Minuun voi kuitenkin olla yhteydessä muulloin esimerkiksi Microsoft Teamsin, sähköpostin tai tekstiviestien kautta.

Jame-Shaft ei ole ohjelmistokehitykseen erikoistunut yritys, joten ainoa henkilö, joka pystyisi osallistumaan kehitystyöhön minun lisäksi on Jame-Shaftilla

täysipäiväisesti työskentelevä IT-asiantuntija. Hänellä on kuitenkin tälle ajanjaksolle muita tehtäviä, joten hän ei osallistu suoraan projektin kehittämiseen. Hän kuitenkin tietää enemmän varastohallintajärjestelmän alkuperäisestä kehityksestä, joten voin tarvittaessa kysyä häneltä tietoa sovelluksesta. Tämä tulee olemaan hyödyllistä, etenkin koska aikaisemmasta kehityksestä ei ole kauheasti dokumentaatiota.

3 Toiminnan raportti

3.1 Ensimmäinen raportointiviikko

Maanantai 16.1.2023

Työsuhteen ja opinnäytetyön ensimmäisenä päivänä keskityin lähinnä eri ympäristöjen, dokumenttien sekä pääasiallisen projektin määrittämiseen.

Kehitystyön kohteena on tällä hetkellä Jame-Shaftilla toiminnassa oleva varastohallintajärjestelmä. Tätä järjestelmää tarvitsee kuitenkin kehittää monin eri tavoin, ja pääasiallinen kehityskohde on sovelluksen käyttöliittymä. Sovellus on kehitetty muuten MEAN-pinolla, paitsi että tietokantana käytetään Googlen Firebase-pilvitietokantaa MongoDB:n sijaan. Syynä tähän on pääasiassa se, että Firebase viestittää kaikille sovellusta käyttäville tietokantaan tapahtuvista muutoksista samalla tavoin kuin esimerkiksi Socket.io. Socket.io on Node.js:lle kehitetty kaksisuuntainen kommunikaatiotyökalu.

Varastohallintajärjestelmässä ei ole myöskään niin paljoa verkkoliikennettä, että Jame-Shaftin tarvitsisi ostaa suurempaa kapasiteettia Firebaseen Googelta.

Versionhallintajärjestelmänä tulen käyttämään Bitbucketia, sillä Jame-Shaftilla sitä on käytetty kaikkien muidenkin projektien versionhallintajärjestelmänä.

Pääasiallisena viestintäkanavana projektissa osallisina olevien kollegoiden kanssa tulen käyttämään projektille luotua Microsoft Teams -kanavaa. Tulen myös tallentamaan kaikki mahdolliset projektinhallintaan liittyvät dokumentit tälle kanavalle. Tein tälle kanavalle myös tehtävät-välilehden, johon pystyn Kanban-taulukon omaisesti luomaan tehtäviä erilaisiksi korteiksi, joita voin sittemmin siirtää eri valmistumisvaiheiden välillä. Kanban on suosittu projektinhallinnan projektikehys, jossa työtehtävät on jaoteltu omiksi korteikseen, jotka sitten asetetaan eri työvaiheisiin. Vaikkakin kanavalla on

useampi henkilö, niin kehitystyö on pelkästään minun työni. Muut henkilöt voivat kuitenkin seurata etenemistäni tämän Teams-kanavan kautta.

Tiistai 17.1.2023

Toisena päivänä aloitin työstämään ensimmäistä käyttöliittymäparannusta. Kyseessä on lavan siirtämiseen tarkoitettu ponnahdusikkuna, jonka avulla valitaan siirrettävän lavan nykyinen hylly ja sen paikka siinä hyllyssä. Tällä hetkellä tämän valintaan on käytössä pelkästään yksi pudotusvalikko, jossa on listattuna jokainen lavapaikka erikseen. Tavoitteena on saada tästä valikosta visuaalisesti helppolukuisempi, jotta tämänhetkinen ja tuleva lavapaikka on helpompi valita. Tämä tarkoittaa, että valikosta voisi tehdä esimerkiksi samankaltaisen kuin sivuston lavapaikkahallinta-välilehdestä.

Tällä hetkellä ponnahdusikkunana käytetään Angular Material -käyttöliittymäkirjaston Mat Dialog -komponenttia. Päätin hyödyntää samaa komponenttia uudestaan, mutta tällä kertaa käyttäisin dialogin datana jo valmista lavapaikkahallintakomponenttia.

Keskiviikko 18.1.2023

Tämän päivän aikana toteutin suurimman osan lavapaikan valinnan käyttöliittymän muutoksesta. Visuaalisena käyttöliittymäelementtinä toimii Mat Dialog -komponentti Angular Material -kirjastosta. Tässä tilanteessa ponnahdusikkunassa käytettävä lavapaikkahallinta-komponentti muuttuu lavapaikka-komponentin lapseksi. Pystyn välittämään tähän komponenttiin attribuuttina merkkijonon, joka kertoo sen hahmontamisen, eli näytölle kuvastamisen tai luomisen, alkuperän. Alkuperän mukaan lavapaikkahallinta-komponentin lavapaikkoja esittävät kortit tekevät eri asioita. Jos komponentin attribuuttina tuotu merkkijono on muuta kuin tyhjää, käyttäjän painama kortti välittää omat attribuutinsaan lavapaikka.service-komponentissa sijaitsevaan RxJS-kirjaston Subject-elementtiin. RxJS on reaktiivisten lisäosien kirjasto

JavaScriptille. Tällöin kyseinen Subject-elementti myös välittää tietoa kaikille sitä tilaaville muuttujille. Jos merkkijono on tyhjä, komponenttia käytetään alkuperäisessä käyttötarkoituksessaan, ja sen kortit käyttävät alkuperäisiä funktioitaan.

Jos attribuuttina tuotu merkkijono ei ole tyhjä, se on joko ”nykyinen” tai ”seuraava”. Näillä merkkijonoilla on väliä vasta takaisin lavapaikka-komponentissa, jossa se kertoo, painoiko käyttäjä nykyisen lavapaikan vai seuraavan lavapaikan painiketta. Tämän perusteella tieto välittyy lomakkeen eri attribuutteihin.

Torstai 19.1.2023

Tämän päivän aikana aloitin tekemään ohjeet-välilehteä varastonhallintasivustolle, josta työntekijät voivat käydä katsomassa kuinka sivustoa käytetään. Aloitin luomalla sivustolle uuden komponentin Angularin ”ng generate component”-komennolla, ja annoin sille nimeksi ”ohjeet”.

Päätin jakaa ohjeet kolmeksi eri kategoriaksi sivuston välilehtien mukaan. Loin aluksi ohjeet-komponentin HTML-tiedostoon navigaatiopalkin kolmella napilla mitkä vaihtaisivat ohjeiden kategorialle, sekä sen TypeScript-tiedostoon kategoria-muuttujan. TypeScript on vahvemmin tyyhitetty versio JavaScriptistä, mitä muun muassa Angular käyttää JavaScriptin sijaan. Sitten loin HTML-elementin jokaiselle kategorialle jossa olisi itse ohjeiden sisältö. Näissä elementeissä on valittuun kategoriaan perustuva Angularin ngIf-ehto, joka rajoittaa elementtien hahmontamisen valittuun kategoriaan. Tämän myötä navigaatiopalkin napin painallus vaihtaa komponentin kategoria-muuttujan, joka taas vaihtaa hahmonnetun ohjeen.

Jaoin jokaisen kategorian ohjeen eri vaiheisiin, jotka kertovat sivuston välilehtien eri ominaisuuksista ja toiminnallisuuksista. Jokainen vaihe koostuu kategorian käyttöliittymän kuvasta sekä sen selityksestä. Tulen korostamaan kuviin käyttöliittymän keskeisimmät toiminnot.

Perjantai 20.1.2023

Tänään aloitin työstämään ohjeiden kuvien korostamista. Alun perin ajattelin vain käsitteleväni kuvat kuvankäsittelyohjelmalla, mutta päätin kuitenkin tehdä tästä korostamisesta hieman interaktiivisemman. Päätin luoda kuvan päälle toisen "div"-HTML-elementin, johon sitten sijoitin muita elementtejä kehyksiksi. Nämä kehykset korostuvat sen mukaan, jonka ohjeiden päällä käyttäjä pitää hiiren kursoria. Käytin tämän toiminnon kehittämiseen ehkä hieman liikaa aikaa siihen nähden, että työnantaja ei ole vielä hyväksynyt esittämäni ohjeiden toimintatapaa.

Ensimmäisen viikon yhteenveto

Ensimmäisellä viikolla sain alustettua projektin viestintäkanavat sekä tutustuttua aikaisempaan varastohallintajärjestelmään. Sain aloitettua varastohallintajärjestelmän kehitystä aloittamalla kahta työnantajan antamaa tehtävää. Ensimmäinen oli varastohallintajärjestelmän töiden siirtämisessä käytettävän lavapaikkavalintavalikon visuaalinen päivitys ja toinen oli uuden ohjeet-sivun suunnittelu ja luominen.

Aikaisemmin töiden siirtämisessä käyttäjän tarvitsi valita yhdestä pudotuslaatikosta suoraan tulevan lavapaikan tunnus. Lavapaikat on kuitenkin jaettu eri ryhmiin niiden tehtaassa sijaitsevien hyllyjen mukaan, joten olisi luontevampaa ensin valita lavapaikan hylly, ja sen jälkeen vasta lavapaikka tästä hyllystä. Samankaltainen ratkaisu on jo kehitetty sovelluksen lavapaikkojen hallintaan tarkoitetulla sivulla, josta on kuvakaappaus kuvassa 1. Kuva on sumennettu yrityksen tietojen turvaamiseksi. Tämän myötä lähdin kehittämään samanlaista ratkaisua myös töiden lavapaikkojen valitsemiseen.



Kuva 1. Kuvakaappaus varastohallintajärjestelmän lavapaikkahallinta-sivulta.

Ohjeet-sivun tarkoituksena on ohjeistaa työntekijöitä varastohallintajärjestelmän käytöstä. Tein ensin erillisen ohjeet-sivun, johon pääsisi käsiksi navigaatiopalkin kautta. Työnantajan palautteen perusteella tein kuitenkin ensimmäisen version sijaan jokaiselle olemassa olevalle varastohallintajärjestelmän sivulle ohjeet ponnahdusikkunaksi. Näitä ohjeita pystyisi katsomaan painamalla jokaisen sivun otsikkopalkin oikeassa reunassa sijaitsevaa infonappulaa. Ponnahdusikkunana käytin Material-kirjaston Mat Dialog -komponenttia. Halusin myös katsoa ohjeet-sivun tekemiseen vinkkejä siltä varalta, että olisiko ohjeet-sivuun hyvä sisältää tiettyjä ominaisuuksia, joita ei tule välttämättä mieleen. Löysin Riddhi Patelin kirjoittaman blogin aiheesta (Patel, 2021), jossa hän analysoi, milloin sivusto tarvitsee ohjeita ja millaisia nämä ohjeet kuuluisivat olla. Hänen mainitsemansa syyt ohjeiden luomiseen pätevät hyvin myös varastohallintajärjestelmään, kuten myös suuri osa hänen mainitsemistaan kriteereistä hyvien ohjeiden luomiseen.

3.2 Toinen raportointiviikko

Maanantai 23.1.2023

Tämän päivän aikana kehitin ohjeet-sivun esittelykelpoisiksi. Tämä tarkoitti muutaman esimerkkiohjeen tyyllittelyä sekä ohjetekstien keksimistä. Tein myös ohjeen kuvista interaktiiviset, eli kun käyttäjä laittaa kursorin jonkin ohjeen osion päälle, se korostuu punaisella ääriviivalla ohjeen kuvan päällä. Tämä kertoisi käyttäjälle tietyn sivun osion tai painikkeen, josta halutun ominaisuuden voi löytää. Voi olla, että suurin osa käyttämästäni ajasta tämän tyylisten ohjeiden kehittämiseen on turhaan, mutta halusin tehdä nämä ohjeet kuitenkin edes toiminnallisiksi.

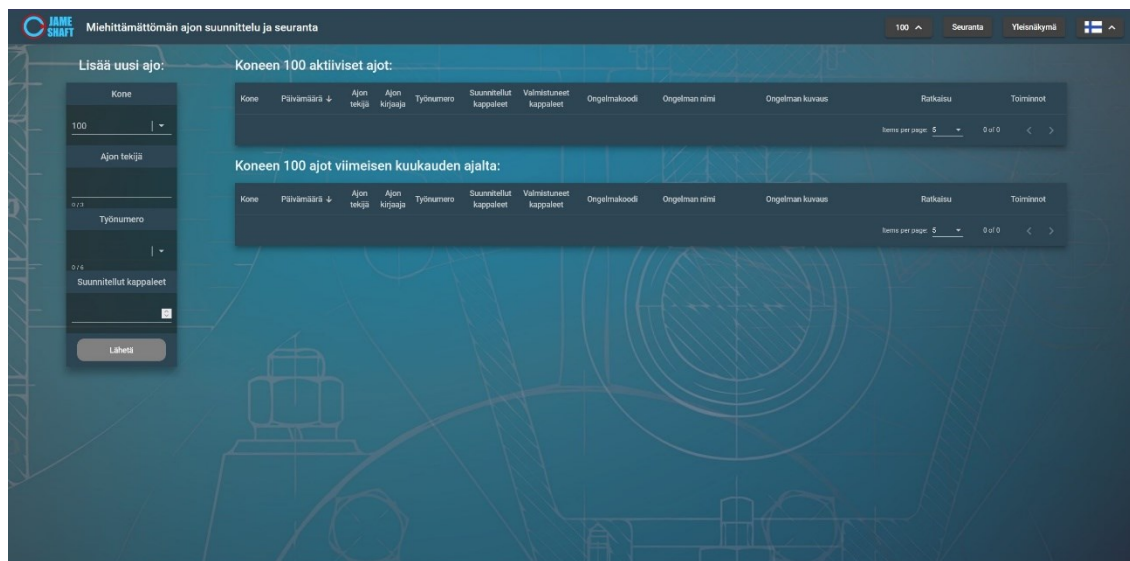
Tiistai 24.1.2023

Tänä päivänä sain uuden toimeksiannon aikaisemmin Jame-Shaftille luomastani ohjelmasta. Ohjelman tarkoituksena on kerätä tietoa koneistuksessa käytettävien robottien automaattisesti suorittamista työvaiheista, eli automaattiajoista. Työntekijä kirjaa ohjelmaan ennen lähtöään automaattiajon työnumeron, koneen sekä odotetun kappalemäärän. Kun automaattiajo on suoritettu, seuraava työntekijä kirjaa ajon aikana suoritettua kappalemäärän ja kirjoittaa virhekoodin ja kuvauksen virheestä, jos lopullinen kappalemäärä ei ole sama kuin odotettu kappalemäärä.

Tämän sovelluksen kehityspyynnössä pyydettiin lisäämään automaattiajojen kirjaukseen myös jokaisen työntekijän numero, jotta voitaisiin paremmin seurata kunkin työntekijän automaattiajojen kirjaamista. Niinpä otin tämän ominaisuuden suurimman osan kehittämisen tämän päivän tavoitteeksi.

Kaikki automaattiajot tallennetaan MS SQL -tietokantaan, joten ensiksi lisäsin tietokannan ajot-tauluun sarakkeet "planner" ja "recorder" merkitsemään työn aloittajaa ja työn kirjaajaa. MS SQL on Microsoftin kehittämä relaatiotietokanta. Sovelluksen tausta on kehitetty MVC-arkkitehtuurilla, joten minun tarvitsi

muuttaa pelkästään ajoModel-tiedostosta ajoModel-oliota. MVC-arkkitehtuuri on ohjelmistoarkkitehtuuri, jossa erotetaan tietokanta, tietojen käsittely ja sovelluksen selainpuoli eri osioikseen. Seuraavaksi aloitin kehittämään sovelluksen selainpuolta, joka koostuu kolmesta pääasiallisesta komponentista: ajon lisäämisestä, aktiivisista ajoista sekä kaikista ajoista viimeisen kuukauden aikana. Kuvassa 2 näkyy sovelluksen tämänhetkinen selainpuoli.



Kuva 2. Automaattiajojen seurantasovelluksen käyttöliittymä.

Ajojen lisääminen tapahtuu lomakkeella, jossa on kolme input-kenttää, ja loput ajon sarakkeista täyttyvät automaattisesti. Minun tarvitsee ainoastaan lisätä tähän lomakkeeseen yksi sarake, johon käyttäjän tarvitsee lisätä hänen työntekijänumeronsa.

Aktiiviset ajot sekä viimeisen kuukauden ajot ovat molemmat Angular Material -kirjaston Material Table -komponentteja. Tällöin minun tarvitsee ainoastaan lisätä näihin tauluihin "planner" ja "recorder"-sarakkeet, ja Material Table -komponentti osaa lisätä tiedot tauluun. Tämän lisäksi tauluja tarvitsee tyyllitellä uudelleen, ja tästä jätin suurimman osan seuraavalle päivälle.

Keskiviikko 25.1.2023

Tänä päivänä tavoitteenani oli suorittaa automaattiajojen seurantasovelluksen taulujen ja ajojen lisäyslomakkeen tyyllittely. Tyyllittelyssä meni suhteellisen kauan, koska olen viimeksi muuttanut sovellusta noin puoli vuotta sitten, enkä muistanut enää, kuinka Material Table -komponenttien tyyllittely toimi tarkalleen. Minun olisi pitänyt tätä sovellusta tehdessäni laittaa muistiin edes ne lähteet, joista katsoin ohjeita Material-kirjaston komponenttien tyyllittelemiseen, sillä uskon tulevani käyttämään kyseistä kirjastoa myös tulevaisuudessa.

Sain kuitenkin taulut ja lomakkeet tyyllitetyä, ja jäljelle jäi tehtäväksi ainoastaan varmistusponnahdusikkunoiden tietojen järjestyksen muuttaminen. Nämä ponnahdusikkunat kysyvät käyttäjältä varmistusta aina kun sovellukseen lisätään, muokataan tai poistetaan ajoa. Ponnahdusikkuna näyttää käyttäjälle ajon tiedot taulukossa, sekä vaihtoehdot tietojen hyväksymiseen tai hylkäämiseen. Tällä hetkellä nämä ponnahdusikkunat saavat tietoa väärässä järjestyksessä, joten siinä esiintyvän taulukon sarakkeet eivät vastaa oikeita ajon tietoja. Kun tämä järjestys on korjattu, voin jatkaa varastohallintajärjestelmän kehittämistä.

Torstai 26.1.2023

Tänä päivänä päätin kehittää hetken varastohallintajärjestelmää ennen seurantasovelluksen loppuun korjaamista. Syy tähän oli varastohallintajärjestelmän QR-koodilukijan rikkoutuminen lavapaikkahallintavalintaa muuttaessani. QR-koodilukija käyttää käyttäjän laitteen kameraa lukeakseen työn lavassa sijaitsevan QR-koodin, joka sisältää työn numeron. QR-koodi on neliön muotoinen ruutukoodi, joka voi sisältää tietoa kuten verkkosivun osoitteen. QR-koodia käytetään sovelluksessa vaihtoehtoisena lavapaikkojen valintamenetelmänä, joten sen rikkoutuminen lavapaikkavalinnan tekniikan muuttuessa käy järkeen.

Päätin kuitenkin ennen tämän ongelman korjaamista päivittää projektin npm-paketit sekä Angular-version. Npm on pakettienhallintajärjestelmä, jonka avulla pystyy käyttämään JavaScriptillä kirjoitettuja lisäosia eli "paketteja". Projektia on viimeksi päivitetty muutama vuosi sitten, ja se käyttää Angular 9 -versiota. Löysin päivittämiseen hyvältä vaikuttavan verkkosivun (Angular, 2023a) suoraan Angularin kehittäjiltä, joka kertoo askel askeleelta version päivittämisprosessin.

Päivittämisprosessi ei tosin ollut niin helppo mitä tämä sivusto antoi olettaa. Projektin monien vanhentuneitten pakettien takia suurin osa päivityksistä ei halunnut asentua. Näihin ongelmiin päätin pakottaa päivitykset, mistä aiheutuvat todennäköiset ongelmat joudun selvittämään jälkikäteen. Varastonhallintajärjestelmässä on paljon ylimääräisiä tai vanhoja komponentteja ja ominaisuuksia, joten en ole varma, tarvitaanko kaikkia näitä paketteja loppujen lopuksi.

Perjantai 27.1.2023

Päivitettyäni projektin Angular-version 9:stä 15:een, aloitin korjaamaan päivityksestä aiheutuneita ongelmia. Suurimmat ongelmat aiheutuivat Angularin Material-kirjastosta, sekä Angularin uuden Firebase-paketin integroinnista.

Angular Material -kirjaston pystyin hyvin pitkälti integroimaan uuteen Angularin versioon käyttämällä Angularin tarjoamaa mdc:migration-työkalua (Angular, 2023b). Tämän prosessin jälkeen minun tarvitsi ainoastaan siirtää pakettien tuonnit erillisestä material.module-tiedostosta takaisin app.module-tiedostoon, jotta sovellus osaisi hyödyntää näitä paketteja.

Varastonhallintajärjestelmän vanhemmassa versiossa käytettiin vanhaa angularfire2-nimistä pakettia. Nykyään Firebase-tietokanta on paremmin integroitu itse Angulariin, ja sen löytää jo suoraan @angular-pakettiryhmästä. Tämä paketti pitää ensin asentaa pakettienhallintajärjestelmän kautta. Itse käytän npm-pakettienhallintajärjestelmää, joten käytin sen asentamiseen komentoa "npm i @angular/fire". Kun npm sai asennettua paketin, kirjauduin sisään konsolissa

käyttämällä "firebase login" -komentoa. Kirjautumisen jälkeen asensin paketin vielä erikseen Angularin omalla "ng add @angular/fire" -komennolla. Tämän asennuksen yhteydessä tarvittiin kirjautumistunnuksia, ja tämä hoitui automaattisesti, koska kirjauduin aiemmin paketin kautta käyttämälleni laitteelle.

Toisen viikon yhteenveto

Toisella viikolla sain kehitettyä aikaisemmin Jame-Shaftille luomaani sovellusta, päivitettyä varastohallintajärjestelmän Angularin ja npm-pakettien version sekä jatkettua ensimmäisellä viikolla aloittamani varastohallintajärjestelmän ominaisuuksia.

Aikaisemman kehittämäni sovelluksen, automaatioseurantasovelluksen, läpikäynnin aikana huomasin käyttäneeni ohjelmoinnissa funktioita ja tekniikoita, joita en muistanut osanneeni käyttää. Tämä luultavasti johtuu hyvin pitkälti suoraan kopioidusta koodista, minkä takia pyrinkin käyttämään suoraan kopioitua koodia suhteellisen vähän. Koen oppivani enemmän referoidessani ja kirjoittaessani itse internetistä löytämäni koodia verrattuna suoraan kopioimiseen.

Olen kerran aikaisemmin päivittänyt yhden toisen Jame-Shaftin sovelluksen Angular-version. Tämä prosessi oli paljon helpompi kuin varastohallintajärjestelmän päivittäminen, mutta siinä projektissa ei myöskään ollut läheskään yhtä montaa npm-pakettia ja niiden välisiä riippuvuuksia kuin varastohallintajärjestelmässä. Päivittämiseen käyttämälläni ohjesivustolla ei mainita mitään mahdollisista ongelmista päivityksen aikana, joten sellaisen kohdatessaan joutuu käyttäjä turvautumaan joihinkin toisiin tietolähteisiin (Angular, 2023a). Tämän takia käyttämäni ohjesivusto ei riittänyt itsessään Angularin version päivittämiseen, vaikka se onkin itse Angularin kehittäjien tekemä ohjesivusto.

Muutkin kehittäjät ovat kohdanneet samankaltaista ongelmaa Angularin oman päivitysohjeen kanssa, missä on jouduttu turvautumaan pakotettuihin Angular-päivityksiin käyttämällä "—force"-tunnistetta (Yigit, 2021). Oman käsitykseni

mukaan tämän tunnisteen käyttämisessä ei pitäisi olla muuta ongelmaa, kuin että muut ladatut npm-paketit täytyy päivittää erikseen tai poistaa, jos ne eivät ole enää yhteensopivia Angularin päivityksessä tapahtuneiden muutoksien kanssa.

Päivitin mitä luultavammin MEAN-pinoa käyttävien verkkosovelluksien Angular-versioita opinnäytetyöni jälkeenkin, ja osaan nyt käyttää enemmän työkaluja ja metodeja näiden sovelluksien päivittämiseen.

3.3 Kolmas raportointiviikko

Maanantai 30.1.2023

Tämän päivän tavoitteena oli korvata vanha Firebase-paketti uudella. Tämä vanha paketti on aika tiiviisti integroitu varastohallintajärjestelmään, joten arvioisin tämän tehtävän olevan suhteellisen laaja. Sain uuden Firebase-paketin asennettua varastohallintajärjestelmään jo perjantaina, mutta käyttämäni esimerkki ei käyttänyt kaikkia Firebasen ominaisuuksia, kuten käyttäjän todentamista. Niinpä etsin uuden resurssin käyttäjän todentamista varten (Firebase, 2023a).

Uuden paketin integroinnissa käynnistin ensin sovelluksen "npm start" -komennolla, jonka jälkeen aloin käymään läpi kaikkia tiedostoja, joissa ilmeni ongelmia käynnistyksen yhteydessä. Suurin osa näistä tiedostoista oli virheellisiä johtuen vanhasta Firebase-paketista, joten tässä vaiheessa uskon, että ohjelma alkaa toimimaan, kun olen päivittänyt vanhat Firebase-paketin funktiot.

Päivitysprosessi oli suhteellisen vaivaton, ja minun ei edes tarvinnut muuttaa tiedonhakufunktioita, jota vanhassa paketissa käytettiin. Uudessa paketissa käytettiin vielä samoja funktioiden nimiäkin.

Tiistai 31.1.2023

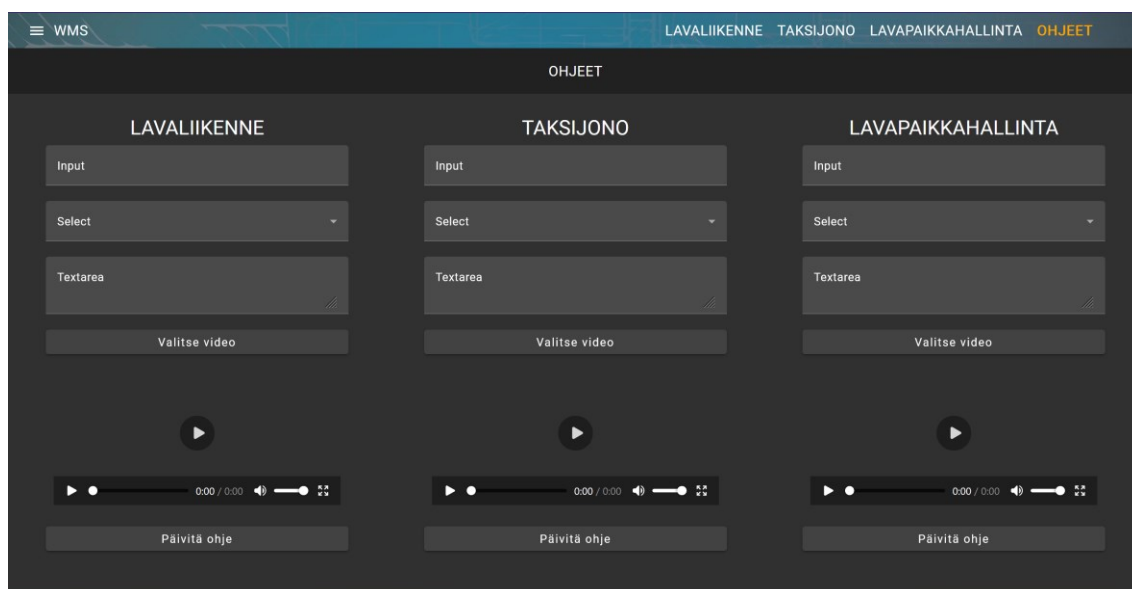
Tämän päivän tavoitteena on korjata päivityksen jälkeinen käyttöliittymä. Osa käyttöliittymän komponenteista ei enää noudattanut aikaisempaa tyyllittelyä. Näiden korjaaminen oli suhteellisen helppoa, sillä minun tarvitsi ainoastaan selvittää päivitettyjen elementtien uudet nimet, ja vaihtaa tyylitiedostosta luokkien nimet niiden nimisiksi.

Loin tämän päivän aikana myös varastohallintajärjestelmän Firebase-tietokantaan uusia käyttäjiä. Näillä käyttäjillä olisi eri oikeudet sovelluksen eri osioihin, ja työntekijöille annettaisiin oikeudet näihin käyttäjiin riippuen heidän työnkuvistaan. Esimerkiksi kesätyöntekijät, jotka hoitavat vain lavojen kuljetusta lavapaikkojen välillä saisivat oikeudet pelkästään varastohallintajärjestelmän taksijono-sivulle.

Tämän lisäksi aloitin ohjeet-sivun uudelleenohjelmoinnin. Tällä kertaa tämän sivun tarkoitus olisi toimia ohjeiden muokkausalustana, josta olemassa olevia ohjeita voisi päivittää tai lisätä. Käytännössä tämä tarkoittaa eri lomakkeiden luomista jokaiselle eri varastohallintajärjestelmän sivulle, jossa pyydetään käyttäjää lisäämään ohjeista tekstiä ja ohjevideon. Tälle sivulle pääsisi käsiksi pelkästään ylläpitäjä tai mahdollisesti jälkikäteen lisättävä ohjeiden muokkaamiseen tarkoitettu käyttäjä.

Keskiviikko 1.2.2023

Tämän päivän aikana kehitin ohjeet-sivun esittelykuntoon. Kuvassa 3 näkyy tekemäni ohjeet-sivun uusi käyttöliittymä, jota tulen esittelemään työnantajalleni.



Kuva 3. Ohjeet-sivun uusi käyttöliittymä.

Ohjeet-sivun lomakkeiden tärkein osio on ohjevideon lataus, ja tämän päivän aikana tein input-painikkeet videon lataamista varten. Kun käyttäjä lataa sivulle videon, se ilmestyy myös alapuolella sijaitsevaan mediasoittimeen.

Tällä hetkellä "Päivitä ohjeet" -painike ei tee mitään. Videotiedostot pitäisi saada tallennettua johonkin paikkaan, mihin kaikilla paikallisverkossa olevilla tietokoneilla olisi pääsy. Tämä tarkoittaisi näiden tiedostojen tallentamista paikalliselle palvelimelle. Tämä ei kuitenkaan onnistu pelkästään Angularin avulla, sillä JavaScriptillä ei ole käyttöoikeuksia käyttäjän tietokoneen tiedostoihin turvallisuussyistä. MEAN-pinoon kuuluvalla Node.js-suoritusympäristöllä kuitenkin on siihen kuuluvan FileSystem-kirjaston avulla pääsy käyttäjän tiedostoihin, tässä tapauksessa palvelimen tiedostoihin. Tämä kuitenkin merkitsisi sitä, että minun tarvitsee luoda tälle ohjelmalle oma Node.js-sovellus tiedonkulkua ja komponenttien välistä tiedonsiirtoa varten.

Aloitin tänään myös varastohallintajärjestelmän komponenttien kartoituksen. Työnantajani mainitsi Angularilla kehitettyjen verkkosovellusten kartoitukseen käytetystä selainlisäosasta nimeltään Augury. Augurya tutkiessani huomasin kuitenkin, että tämä lisäosa on korvattu Angular DevTools -selainlisäosalla. DevToolsilla pystyy nauhoittamaan Angular-sovelluksen käyttöä, ja tästä

nauhoitteesta pystyy katsomaan sovelluksen tiedonsiirtoa, kunkin komponentin käyttöä ja tiedon kulun kestoa. En kuitenkaan löytänyt tästä lisäosasta samankaltaista Component Tree -ominaisuutta, joka näyttäisi eri komponenttien keskinäiset suhteet selvästi. Tällainen ominaisuus on Auguryssä, joka on tarkoitettu aiemmille Angularin versioille. Kokeilin Augurya varastohallintajärjestelmän komponenttien visualisointiin, mutta en saanut sitä toimimaan johtuen varastohallintajärjestelmän liian uudesta Angular-versiosta.

Torstai 2.2.2023

Tämän päivän aikana tein sekalaisia sovelluksen koodin siistimisiä, ja asensin Angular DevToolsin selaimeeni. Sekalaiseen siistimiseen kuului muun muassa vanhan kommentoidun koodin poistamista ja erinäisten funktioiden koodin uudelleenjärjestelyä.

Perjantai 3.2.2023

Tämän päivän aikana jatkoin sekalaisia sovelluspäivityksiä. Tähän kuului muun muassa lavaliiikenteen töiden Mat Dialog -varmistusponnahdusikkunan uudelleentyylittelyä ja eilisestä siistimisestä johtuneiden ohjelmavirheiden korjaamista.

Kolmannen viikon yhteenveto

Kolmannella viikolla sain päivitettyä varastohallintajärjestelmän käyttämän Firebase-paketin, korjattua Angular- ja Firebase-paketista johtuvat selainpuolen tyylittelyvirheet, kehitettyä uudenlaisen ohjeet-sivun, kehitettyä suurimman osan automaattiajojen seurantasovelluksen päivityksestä sekä kartoitettua varastohallintajärjestelmän komponentit. Tällä viikolla oli paljon tehtävää, ja pystyin kirjaamaan monta tehtävää valmiiksi Teamsin tehtävät-taulukosta.

Suurin osa kehittämistäni ominaisuuksista tällä viikolla onnistuivat hyvin, enkä usko, että minun olisi kannattanut tehdä näitä asioita eri tavoin.

Tyylittelyvirheiden korjaamisessa ja uusien ohjeet-sivujen kehittämisessä olisi kuitenkin ollut muitakin tapoja, ainakin selainpuolen ohjelmoinnissa. Olisin voinut esimerkiksi käyttää joidenkin elementtien keskittämiseen joko CSS:n flex-attribuuttia tai padding-attribuuttia. Itse käytän lähes aina flex-attribuuttia align-items -attribuutin ja justify-content -attribuutin kanssa, sillä olen huomannut sen olevan joustavampi ja luotettavampi tapa keskittää HTML-elementtejä.

Kehittämäni ratkaisut toimivat joka tapauksessa hyvin, joten pysyttelen niissä. Varastohallintajärjestelmän komponenttien kartoittamisen olisin tosin voinut tehdä paremmin, sillä kartoittamalla saadut kaaviot eivät olleet yhtä helppolukuisia kuin Augury:n luomat, eivätkä ne kuvastaneet sovelluksen komponenttisuhteita samalla tavoin kuin Augury.

Varastohallintajärjestelmän komponenttien visuaalinen kartoittaminen miellekartan tavoin olisi helpottanut järjestelmän komponenttien välisien suhteiden hahmottamista.

Miellekartat voivat olla hyödyksi esimerkiksi projektin osa-alueiden muistamisessa ja laajojen projektien pilkkomisessa pienempiin osioihin. Nämä miellekartan ominaisuudet olisivat voineet olla hyödyksi myös komponenttien kartoituksessa (Indeed Editorial Team, 2023).

3.4 Neljäs raportointiviikko

Maanantai 6.2.2023

Tämän päivän aikana korjasin automaatioajojen seurantasovelluksen tekstejä, Loin raportit varastohallintajärjestelmän tiedonkulusta ja lisäsin varastohallintajärjestelmän taksijonoon lavapaikkojen uuden valintaponnahdusikkunan.

Seurantasovelluksen komponenteissa näkyvät tekstit määritellään erillisessä palvelutiedostossa. Tässä tiedostossa on kaksi eri merkkijonolistaa kullekin ohjelmassa olevalle kielelle, suomalaiselle ja englannille. Sovellus käyttää jommankumman listan merkkijonoja riippuen käyttäjän valitsemasta kielestä. Seurantasovelluksen suunnittelijan ja kirjaajan lisäämisen jälkeen näihin merkkijonolistoihin täytyi lisätä myös nämä tiedot molemmille kielille.

Varastohallintajärjestelmästä luomani raportit olivat Angularin DevToolsilla tehdyt komponenttien kartoitukseen tarkoitetut raportit. Nämä raportit ovat yksinkertaisia JSON-tiedostoja, joihin DevTools tallentaa käyttäjän sivustolla tekemiä toimintoja, sovelluksen käyttämiä komponentteja sekä niiden välisiä riippuvaisuuksia. Loin jokaiselle varastohallintajärjestelmän sivulle oman raportin, jossa kävin läpi sen kaikki toiminnallisuudet. Näiden raporttien avulla pystytään myöhemmin selvittämään, mitä komponentteja ei enää tarvita varastohallintajärjestelmässä.

Taksijonon lavapaikkojen valinta tapahtuu nyt myös samantapaisesti kuin lavaliikenteessä.

Tiistai 7.2.2023

Tämän päivän aikana sain suoritettua loppuun automaatioajojen seurantasovelluksen päivityksen. Tämäkin ongelma oli hieman vaikeampi kuin alun perin luulin, johtuen MS SQL -tietokannan ajot-tauluun lisättyjen "planner" ja "recorder" sarakkeiden sijainnista. Koska sarakkeet lisättiin vasta jälkikäteen, tulevat ne olemaan taulussa loppupäässä indekseissä 11 ja 12. Sovelluksen selainpuolessa nämä tiedot tulevat kuitenkin näkymään sijoilla 4 ja 5. Näiden sarakkeiden sijaintia pystyy siirtämään MS SQL -tietokannan taulussa, mutta ei kovinkaan helposti. Koska taulussa on jo niin paljon tärkeää tietoa Jame-Shaftin automaatioajojen kulusta, olisi tämä prosessi vaatinut ensin taulun varmuuskopioinnin, uuden taulun luomisen oikealla sarakejärjestyksellä, ja sitten varmuuskopion liittämisen uuteen tauluun. Tämän sijaan päätin tehdä uuden lajittelun selainpuolessa näkyviin taulukoihin sekä ponnahdusikkunoihin

TypeScriptillä, mikä oli paljon helpompi ja nopeampi ratkaisu. Tietokannan uudelleen järjestäminen olisi ollut niin sanotusti ”oikeampi” ratkaisu, mutta automaattiajajien seurantasovelluksen aikaisempi uudelleenpäivitys oli tärkeämpää kuin sovelluksen täydellinen tietorakenne.

Keskiviikko 8.2.2023

Tämän päivän aikana aloitin varastohallintajärjestelmän töiden logiikan korjaamisen. Tällä hetkellä olemassa olevaa työtä muokatessa ilmestyvä ponnahdusikkuna ei näyttänyt työn nykyistä ja seuraavaa lavapaikkaa oikein. Tämä johtui siitä, että aikaisemmat lavapaikkojen valinnat näyttävät painikkeet näyttivät lavapaikan vain, jos sen muuttujan arvo ja tyyppi oli muuta kuin 0. Tämän myötä lavapaikkapainikkeet näyttivät tyhjää myös, jos lavapaikan muuttujan arvona ei ollut numeroa. Kaikki lavapaikat ovat numeron sijaan merkkijonoja, sillä ne sisältävät myös kirjaimen kuten ”V”.

Torstai 9.2.2023

Tänään jatkoin varastohallintajärjestelmän töiden tiedon logiikan korjaamista. Tällä hetkellä työntekijä ei voi lisätä uutta työtä, mikä pidemmän analyysin jälkeen johtuikin aikaisemmin kommentoidusta koodista, jonka olin tehnyt taas toiminnalliseksi koodiksi. Tämän korjattuani töitä pystyi taas lisäämään, ja seuraava ongelma oli näiden lisättyjen töiden muokkaamisessa. Töitä muokatessa ilmestyvän ponnahdusikkunan lomake ei täytynyt automaattisesti työn tiedoilla. Tämä johtui lähinnä uudesta lavapaikan valintaponnahdusikkunasta ja sen tallentamasta tietorakenteesta. Ponnahdusikkunaa hahmottaessa komponentti myös uudelleenalusti ponnahdusikkunassa käytettävää lomaketta, ja tämän toiminnallisuuden poistamiseen riitti pelkästään uudelleenalustamisfunktion poistaminen. Ponnahdusikkunassa myös näkyi pelkästään lavapaikan ID, eli yksilöivä tunniste. Minun täytyi syöttää tämä tunniste funktioon, joka hakee lavapaikan nimen sen tunnisteiden perusteella.

Automaatioajojen seurantasovelluksessa myös ilmeni yksi ohjelmavirhe, jossa ajon kirjaajan numero korvautui ajon tekijän numerolla. Tämä johtui yksinkertaisesta if-lauseen väärästä parametrasta, jossa tekijän numero korvaantui kirjaajan numerolla, jos ajoa kuitattiin valmiiksi joko OK-nappulan tai muokkauksen kautta.

Perjantai 10.2.2023

Tämän päivän aikana jatkoin varastohallintajärjestelmän logiikan korjaamista ja loin sovelluksessa käytettäville Angular Material -kirjaston komponenteille oman teeman. Tavoitteena oli saada etenkin lavapaikkahallinnan Material Tab -komponenteille takaisin aikaisempi oranssinkeltainen aksenttiväritys.

Aksenttivärityksen korjaamiseen minun täytyi ainoastaan korjata värityksen määrittämiseen käytetty ngClass-ehto. Aikaisempi ehto käytti valmiiksi määriteltä Angular Material -luokkaa, jota ei luultavasti ollut enää sen päivittämisen jälkeen. Tämän takia tein taksijono ja tilaa-kuljetus -komponenttien tyylittelytiedostoihin uuden luokan, joka lisättäisiin ngClass-ehdon avulla komponenteissa olevien taulujen riveille.

Sain myös tämän päivän aikana korjattua suurimman osan varastohallintajärjestelmän logiikasta. Nyt lavapaikkaa valittaessa lavapaikkavalintaponnahdusikkunassa kyseinen lavapaikka muuttuu keltaiseksi osoittamaan, että se on odottamassa tilausta. Tällä tavoin muut työntekijät, jotka ovat tilaamassa kuljetuksia, eivät voi valita samoja lavapaikkoja yhtä aikaa.

Huomasin varastohallintajärjestelmän logiikkaa korjatessani, että suuri osa funktioista ei ole järkevästi lajiteltu, ja tietoa kulkee paljon turhaan eri komponenttien välillä.

Neljännän viikon yhteenveto

Neljännellä viikolla sain viimeisteltä automaattiajojen seurantasovelluksen päivityksen, korjattua varastohallintajärjestelmän töiden logiikkaa ja korjattua varastohallintajärjestelmän tyylittelyä.

Tällä viikolla merkittävin oppimani asia oli Angularin Material-kirjaston teemat, ja ylipäättään teemapohjainen tyylittely. Tekemäni tyylittely toimi pelkästään Material-kirjaston komponentteihin, sillä niiden tyyli määritellään niiden omalla color-attribuutilla, mutta tällainen teemapohjainen tyylittely on itseasiassa paljon järkevämältä vaikuttava vaihtoehto tähän asti käyttämäni, pelkästään CSS-luokkiin pohjautuvaan tyylittelyyn.

Olen Angularin lisäksi käyttänyt Reactia komponenttipohjaisena selainpuolen kehityskirjastona, ja teemapohjainen tyylittely voisi olla seuraava kehityskohteeni Reactin opettelemisessa. Teemapohjaiseen tyylittelyyn mitä luultavammin löytyisi myös npm-paketteja, mutta Tapas Adhikary (2021) on tehnyt hyvin joustavan ja kustomoitavan vaihtoehdon blogissaan. Uskon kehittäväni samankaltaisen järjestelmän omiin sovelluksiini tulevaisuudessa.

3.5 Viides raportointiviikko

Maanantai 13.2.2023

Tämän päivän aikana jatkoin varastohallintajärjestelmän logiikan korjaamista. Viime viikolla sain korjattua tilaa kuljetus -välilehden lavapaikkavalinnan, töiden lisäämisen sekä muokkaamisen, joten tämän päivän aikana keskityin taksijono-sivun toiminnallisuuden varmistamiseen. Tällä hetkellä taksijonossa näkyvien töiden kuljetusta ei voi merkata aloitetuksi tai lopetetuksi. Taksijonon listan riveillä näkyy kuljetuksen aloitukseen käytettävä painike, ja se avaa tarvittavan ponnahdusikkunan oikein. Ponnahdusikkunan lähetä-painike ei kuitenkaan merkitse kuljetusta aloitetuksi, vaan kuittaa sen suoraan valmiiksi. Ongelma lopulta aiheutui tämän ponnahdusikkunan sulkeutuessa, jolloin

ohjelma suoritti aikaisemmin onClose-funktion. Tämä funktio vapauttaa työssä varatut lavapaikat ja resetoi lavapaikkalomakkeen, kun sen ainoastaan kuljetusta aloittaessa tarvitsisi sulkea ponnahtusikkuna ja vaihtaa työn statukseksi ”kuljetuksessa”.

Tiistai 14.2.2023

Tämän päivän aikana jatkoin vielä varastohallintajärjestelmän logiikan korjaamista. Tällä kertaa korjauksen alla oli lavapaikoissa ilmenneet epäsäännöllisyydet, niiden tyylittelyä sekä funktioiden muuttamista. Epäsäännöllisyydet ilmenivät tiettyjä lavapaikkoja valitessa, jolloin lavapaikan tunnus ei täytynyt lavapaikkavalinnan ponnahtusikkunaan. Tätä virhettä oli myös hankala toistaa, sillä näiden virheellisten lavapaikkojen välillä ei ollut mitään yhteneväisyyksiä. Kaikissa valituissa lavoissa oli samanlainen oliorakenne, sekä kaikkien lavapaikkojen tunnukset olivat samassa formaatissa. Varastohallintajärjestelmän Firebase-tietokantaa katsoessani huomasin kuitenkin, että näille lavapaikoille oli ilmestynyt kopiot, joissa ei ollut mitään lavapaikkatietoja. Näitä paikkoja valittaessa ohjelma valitsi nämä versiot lavapaikoista, eivätkä lavapaikkatiedot tämän takia tallentuneet lisättyyn tai muokattuun työhön. Tämänkin jälkeen kuitenkin huomasin joidenkin lavapaikkojen olevan silti virheellisiä samantlaisilla oireilla. Firebase-kannassa ei kuitenkaan näkynyt tällä kertaa olevan mitään väärin, joten ratkaisu tähän ongelmaan oli lopulta tuotannossa käytettävän Firebase-kannan uudelleenkopiointi testitietokantaan. Tällöin valitut virheelliset lavapaikat antoivat täysin samat tiedot kuin aikaisemmin, mutta tällä kertaa ne toimivat kuten pitikin. Loppupäätelmänä voin vain kuvitella, että ongelma oli jokin outo ohjelmavirhe Firebase-tietokannan päässä.

Muutin tämän lisäksi lavapaikkahallinnan funktioita niin, että kun lavapaikkahallinta hahmonnetaan ponnahtusikkunassa, siinä ilmestyviä lavapaikkoja ei voi enää muokata. Tämän sijaan koko kortti laukaisee saman toiminnon kauttaaltaan. Ainoa muutos lavapaikkavalintaan, kun se hahmonnetaan normaalisti, on jokaisen lavapaikkaa esittävän kortin otsikon

korostus, kun kursori asetetaan sen päälle. Näiden lisäksi sain tietää, että nykyinen Firebase-tietokanta on julkisesti nähtävissä internetistä. Tämä oli mielestäni outoa, mutta tietokannan osoitteeseen mentäessä se palauttaa suoraan JSON-muodossa koko tietokannan. Päätin kuitenkin perehtyä tähän ongelmaan seuraavan päivän aikana.

Keskiviikko 15.2.2023

Tämän päivän aikana keskityin varastohallintajärjestelmän Firebase-tietokannan yksityistämiseen. Asiaa tutkiessani kävi ilmi, että estääkseen julkisen pääsyn Firebase-tietokantaan, täytyy pääsy asettaa manuaalisesti käyttäjätunnusten taakse Firebase-konsolista. Ohjelmistossa on jo käyttäjätunnukset, mutta pääsyä tietokantaan ei ollut asetettu näiden käyttäjien taakse. Säännöt, joilla Firebase määrittelee tietokannan käyttöoikeudet sekä tiedon suodattamisen, on oma JavaScript-oliota muistuttava osio tietokannan konsolissa. Tämän olion kautta sääntöjen määrittely on helppoa, mutta niinkin yksinkertainen sääntö, kuin tietokannan luku- ja kirjoitusoikeuden siirtäminen pelkästään tietokantaan kirjatuille käyttäjille, vaati jonkin verran omaa kokeilemistä. Käytin sääntöjen kirjoittamiseen Googlen omaa ohjesivustoa (Firebase, 2023b), mutta sielläkään ei mainittu toimivaa ratkaisua, johon lopulta päädyin.

Torstai 16.2.2023

Tämän päivän aikana kehitin ohjevideoiden tallentamista ja hakemista palvelimelta. Tähän tarkoitukseen käytän Node.js:n FileSystem-kirjastoa. En kuitenkaan saanut videoiden hakemista toiselta levyltä toimimaan, joten päätin jättää ominaisuuden kehittämisen myöhemmälle.

Perjantai 17.2.2023

Tämän päivän aikana jatkoin varastohallintajärjestelmän taksijonon logiikan korjaamista. Yritin tätä ennen myös löytää Auguryn tyyppistä reittipuun visualisointia varastohallintajärjestelmälle. Taksijonon komponentissa käytetään paljon eri funktioita, joissa tieto kulkee alustavasti katsoen turhaan komponentista toiseen. Reittipuun visualisointi auttaisi näkemään missä kaikkialla kyseisiä komponentteja käytettäisiin, ja tämän myötä selviäisi olisiko tiedon kuljettaminen näiden komponenttien läpi turhaa. Augury on kuitenkin vanhentunut lisäosa, ja uudemmille Angularin versioille ei ole tehty samankaltaista lisäosaa. Augury on korvattu Angular DevTools -nimisellä selainlisäosalla, mutta siinä ei ole yhtä helposti luettavaa visualisointia kuten reittipuuta. Päätin tämän takia jättää visualisoinnin myöhemmälle, ja pyrin saamaan taksijonon lavapaikkavaihdon toimimaan nykyisellä varastohallintajärjestelmän tiedonkululla. Sain tämän päivän aikana tehtyä taksijonon kuljetuksen aloituksen ja lopetuksen lavapaikkavaihdon toimivaksi. Sain myös näiden ponnahdusikkunoiden tekstit kertomaan kyseisen työn numeron.

Viidennen viikon yhteenveto

Viidennellä viikolla sain korjattua varastohallintajärjestelmän taksijono-sivun ponnahdusikkunoita, taksijono-sivun logiikkaa ja Firebase-tietokantaa. Sain myös suojattua Firebase-tietokannan tiedot käyttäjätunnuksien taakse, jotta tietoja ei pystyisi näkemään enää tietämällä pelkästään tietokannan URL-osoitteen, eli verkko-osoitteen, jolla selain pystyy löytämään sisältöä internetistä, kuten verkkosivuja.

Firebase-tietokannan yksityistäminen ja siihen liittyvät tietokannan säännöt myös herättivät kysymyksiä tämänkaltaisen verkkotietokannan turvallisuudesta ylipäätään. Tom Colvin (2023) analysoi blogissaan Firebasen todennuksen turvallisuutta ja sen heikkoja puolia. Loppupäätelmänä hän kuitenkin toteaa Firebasen olevan turvallinen, kunhan sen määrittää ja käyttää oikein.

3.6 Kuudes raportointiviikko

Maanantai 20.2.2023

Tämän päivän aikana jatkoin varastohallintajärjestelmän ohjevideoiden kehittämistä. Viime kerralla jäin jumiin oikean tiedostopolun löytämiseen, mutta tällä kertaa tiedosto löytyi Jame-Shaftin palvelimelta helposti. Muistaakseni olin kokeillut nyt toimivaa polkua aikaisemminkin, mutta voi olla, että olin jättänyt esimerkiksi pari vinoviivaa pois polusta. Kun tiedosto löytyi, aloitin sen palauttaman tiedoston käsittelyn ohjelman taustassa. Ensin päätin kokeilla readStreamin laittamista, sillä videotiedostot ovat isoja. Kun palvelimella kesti kuitenkin hyvin kauan palauttaa kaikki data videotiedostosta, mietin että eihän videoita käsitellä tällä tavoin millään muullakaan sivustolla. Sen sijaan että sovelluksen selainpuoli pyytää taustaa palauttamaan koko tiedoston, pitäisi taustan vain antaa tiedostopolku palvelimella sijaitsevaan videoon. Tämän jälkeen päätin kokeilla Node.js:n oman URL-paketin pathToFileURL-funktiota, joka luo annetusta tiedostopolusta version, jonka avulla sovelluksen selainpuolikin voi haetun ohjevideon löytää ja toistaa. Kun sain videon URL-osoitteen luonnin toimimaan, en voinut kuitenkaan testata sitä suoraan sovelluksessa, koska tätä versiota tästä taustaohjelmasta ei ollut käännetty palvelimelle aktiiviseksi versioksi. Kollegani ei myöskään enää ollut tähän aikaan toimistolla, joten en voinut pyytää häntä kääntämään ja lataamaan uutta versiota palvelimelle.

Tiistai 21.2.2023

Tämän päivän aikana jatkoin ohjevideoiden URL-osoitteen keksimistä, ja sain selville, että Node.js:n FileSystemin readStream-funktionaalisuutta käytetään kuitenkin videoiden toistamisessa verkkopalveluiden kautta. Tämän päivän aikana myös lisäsin varastohallintajärjestelmän lavapaikkavalintaponnahdusikkunan lavapaikkoihin merkit, mitkä merkitsevät nykyistä valittua lavapaikkaa. Tämä merkki on lavapaikkaa esittävään korttiin

ilmestyvä valkoinen, 5 pikselin levyinen kehys, joka kertoo käyttäjälle nykyisen valitun lavapaikan. Tämän ominaisuuden lisääminen vaati uuden attribuutin lisäämistä ponnahtusikkunan määrittävään olioön. Sen lisäksi että nykyinen lavapaikka korostuu, myös lavapaikkahallinnan ponnahtusikkunan hylly vaihtuu nykyisen lavapaikan ryhmään. Tämän tiedon pystyi välittämään aiemmin lisäämän attribuutin kanssa, mutta minun täytyi lisätä lavapaikkahallintakomponenttiin uusi taulukko, jossa on kaikkien hyllyjen tunnukset. Lisäsin tämän sen takia, koska Material Tab -komponentin oletusvälilehti määritetään välilehtien indeksin mukaan. Tämän myötä pystyin lavapaikkahallintaponnahtusikkunaa hahmottaessa määrittämään oletusvälilehden JavaScriptin IndexOf-funktiolla.

Näiden lisäksi aloitin sovelluksen siistimistä. Aloitin lisäämällä kaikki varastohallintajärjestelmässä käytettävät ponnahtusikkunat samaan tiedostoon – dialog.serviceen.

Keskiviikko 22.2.2023

Tämän päivän aikana jatkoin dialog.servicen kehittämistä, ja laitoin kaikki varastohallintajärjestelmässä esiintyvät ponnahtusikkunat sinne.

Tämän lisäksi jatkoin ohjevideoiden hakemista taustasta. Lähteinä tähän tehtävään toimivat Mark Vassilevskiyn (Vassilevskiy, 2022) ja Samuel Martinsin (Martins, 2022) blogit. Käytin ensin Martinsin luomaa taustaohjelmakoodia ja Vassilevskiyn selainpuolta. Tämän kautta taustaohjelmisto ensin luo readstreamin videosta, ja sen jälkeen selainpuoli pystyy tilaamaan tämän streamin ja toistamaan videota sitä mukaa kuin tausta sitä antaa.

Torstai 23.2.2023

Jatkoin tämän päivän aikana ohjevideoiden päivittämistä. Koodin pitäisi toimia, mutta selainpuoleen ilmenevä koodi ei kuitenkaan ole käytettävissä. Lopulta kävi ilmi, että ongelma oli haettavassa MP4-muotoisessa videossa. Jotta MP4-

muotoista videota pystyisi suoratoistamaan, sen täytyy olla niin sanottu "fragmented MP4". Esimerkkivideon käyttämäni Jame-Shaftin työturvallisuusvideo ei ollut "fragmented MP4", joten ensiksi pyrin muuntamaan sen sellaiseksi. Tutkimisen jälkeen sain selville, että MP4-videot pystyy "fragmentoimaan", eli paloittelemaan ffmpegillä. Ffmpeg on komentoriviltä käytettävä työkalu, millä pystyy muun muassa purkamaan ja muuntamaan kaikenlaisia multimediatiedostoja. Minulla on jo asennettuna ffmpeg, mutta minun piti vielä lisätä se tietokoneeni PATH-ympäristömuuttujaan, jotta voisin käyttää sitä terminaalissa. Videotiedoston konvertointiin käytin seuraavaa komentoa: "ffmpeg -re -i Jameshaft.mp4 -g 52 -c:a aac -b:a 64k -c:v libx264 -b:v 448k -f mp4 movflags output.mp4".

Konvertoinnin jälkeen latastin videon takaisin palvelimelle, mutta se ei kuitenkaan toiminut. Videon toisto toimi sen jälkeen, kun laitoin palvelimen API-osoitteen, eli rajapintaosoitteen, suoraan videon lähteeksi.

Perjantai 24.2.2023

Sain eilisen päivän aikana kehitettyä ohjevideoiden hakemisen sovelluksen selainpuoleen get-pyynnöillä, joten tämän päivän aikana päätin kehittää uusien ohjevideoiden päivittämisen post-pyynnöt palvelimelle. Tämäkin prosessi paljastui työlääksi, sillä tiedon hankkiminen aiheesta oli yllättävän hankalaa. Löysin kuitenkin lopulta Petr Večeřan kirjoittaman blogin (Večeřa, 2018), jonka mukaan aloitin sovelluksen taustan kehittämisen.

Kuudennen viikon yhteenveto

Kuudes viikko ei ollut yhtä tehtävärikas kuin aikaisemmat viikot, sillä käytin suurimman osan siitä pelkästään ohjevideoiden tekemiseen. Ohjevideoiden kehityksen lisäksi siirsin kaikki varastohallintajärjestelmässä käytettävät ponnahdusikkunat omaan palvelutiedostoonsa.

Jälkikäteen ajateltuna minun olisi pitänyt luultavasti keskittyä enemmän varastonhallintajärjestelmän muihin ongelmiin, kuten esimerkiksi töiden logiikan testaamiseen. Ohjeet ovat kuitenkin toissijainen ominaisuus verrattuna itse varastonhallintajärjestelmän satavarmaan toiminnallisuuteen, ja testaamalla olisi voinut selvittää mahdollisia ongelmia töiden logiikassa. Sanoisin kuitenkin oppineeni paljon tällä viikolla isojen tiedostojen siirtämisestä verkkosovelluksien selainpuolen ja taustan välillä, tässä tapauksessa ohjevideoiden. Uskon, että tulen käyttämään videoiden lataamisessa käytettyjä `readStream`- ja `writeStream`-funktioita tulevaisuudessa. Nämä funktiot myös vaativat paljon asynkronista ohjelmointia, mitä olen tehnyt vähemmän kuin synkronista ohjelmointia. Tämä sai minut kiinnostumaan oman asynkronisen ohjelmoinnin tietoni kertaamisesta. Löysin hyvältä vaikuttavan blogin, jossa kerrottiin yleisesti asynkronisesta ohjelmoinnista (Indeed Editorial Team, 2022). Tiesin suurimman osan tämän blogin kertomista asynkronisen ohjelmoinnin hyödyistä, mutta asynkronisen ohjelmoinnin mahdollistama visualisointi verkkosovelluksissa, esimerkiksi latauspalkit yms., oli ominaisuus, jota en ole ainakaan aikaisemmin hyödyntänyt.

3.7 Seitsemäs raportointiviikko

Maanantai 27.2.2023

Sain tämän päivän aikana vihdoin videoiden lataamisen taustaan toimimaan. Jatkoin kehitystä viime viikolta käyttäen samaa tietolähdettä, Petr Večeřan kirjoittamaa blogia (Večeřa, 2018).

Tiistai 28.2.2023

Tämän päivän aikana parantelin kaikennäköisiä pienempiä ongelmia:

- Vaihdoin tiettyjen nappuloiden värejä keltaiseksi käyttäen aikaisemmin määrittelemiäni teemoja.

- Korjasin ohjelmavirheen, jossa lavapaikkavalinnan onClose-funktio vapautti lavapaikat, joissa oli jo valmiiksi töitä.
- Ohjelmoin kaikki työt, joilla ei ole nykyistä lavapaikkaa poistumaan tietokannasta.

Tämän päivän aikana pyrin työnantajani kanssa myös laittamaan testiversion tuotantoon. Ongelmia oli kuitenkin ympäristömuutujissa, joissa ne eivät vaihtaneet ohjevideoiden hakuosoitetta.

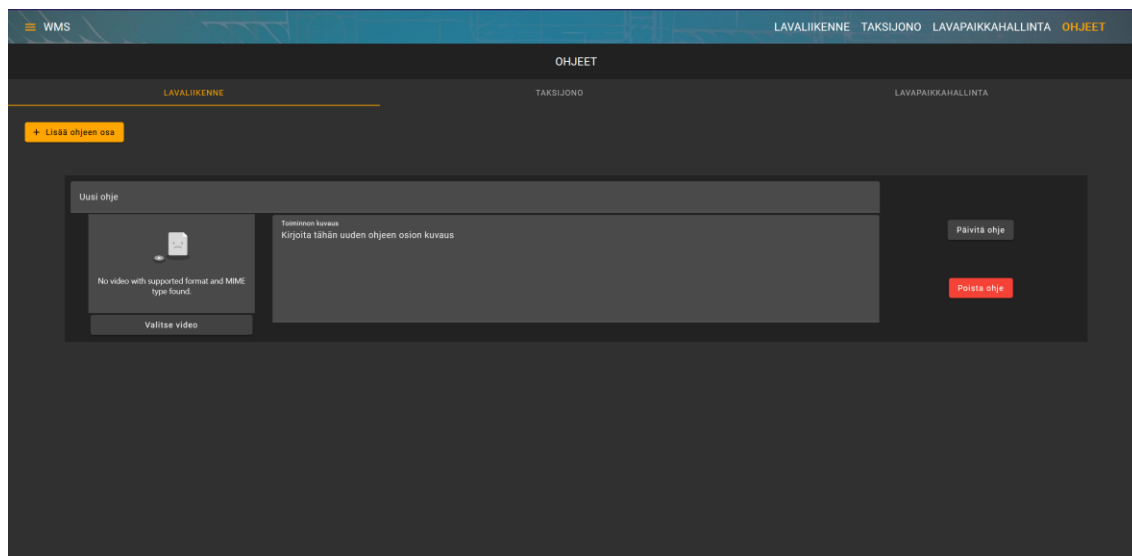
Keskiviikko 1.3.2023

Tämän päivän aikana muokkasin ohjeiden muokkaussivua uusien ohjeiden mukaisesti. Uusiin ohjeisiin pääsee käsiksi samoista paikoista, ja on jaoteltu samalla tavoin kuin aikaisemminkin. Uudet ohjeet koostuvat useasta lyhyemmästä videosta/kuvasta, mitkä kertovat jostain tietystä kategorian ominaisuudesta. Näitä erilaisia "askeleita" pystyy lisäämään, muokkaamaan ja poistamaan.

Torstai 2.3.2023

Tämän päivän aikana jatkoin ohjesivujen muokkaamista. Sain tämän päivän aikana suurimman osan ohjesivuston selainpuolesta kuntoon, ja sain ohjeosioden automaattisen generoinnin taustasta tuleville videotiedostoille toimimaan. Käytän alasvetolaatikon sijaan Angular Material -kirjaston mat-tab-group -välilehtiryhmäelementtiä näyttämään kaikki sivuston eri kategoriat. Kun käyttäjä vaihtaa välilehtielementin välilehteä, sovellus hakee sen kategorian videoiden osoitteet palvelimelta, ja palauttaa ne sovelluksen selainpuoleen. Sen sijaan, että sovellus hakisi sitä hahmontaessa kaikkien kategorioiden videot samanaikaisesti, päätin kehittää sovelluksen hakemaan valitun kategorian jokaisen videon kategoriaa vaihdettaessa. Tämä voi lisätä kyselyitä palvelimelle, jos käyttäjä vaihtaa paljon valitsemaansa kategoriaa, mutta en usko tämän olevan todennäköistä. Käyttäjä on ohjeet-välilehdellä

muokkaamassa ohjeita, joten hän tulee olemaan vain yhdessä ohjeiden kategoriassa kerrallaan. Kuvassa 4 näkyy kehittämäni uusi ohjeet-sivun käyttöliittymä.



Kuva 4. Ohjeet-sivun uusin käyttöliittymän versio.

Perjantai 3.3.2023

Tämän päivän aikana kehitin lisää ohjesivuja. Sain tämän päivän aikana valmiiksi ohjeen osan JSON ja MP4 tiedostojen lukemisen ja tuomisen Node.js:n kautta sovelluksen selainpuoleen. Jäljellä ohjeista on vielä tiedostojen muokkaaminen ja tallentaminen taustaan, ja tietojen näyttäminen ohjeiden ponnahdusikkunoissa.

Seitsemännen viikon yhteenveto

Tällä viikolla sain kehitettyä ohjeet-sivulle uuden selainpuolen, korjattua useamman pienemmän ohjelmavirheen sekä kehitettyä toimivan videoiden latausjärjestelmän.

Ohjeet-sivujen kehittäminen on tähän mennessä vienyt paljon enemmän aikaa, kuin alun perin olin ajatellut. Tämä ominaisuus ei ole myöskään yhtä

merkityksellinen kuin muut varastohallintajärjestelmän ominaisuudet, mutta tämänkaltainen videoiden muokkausjärjestelmä vaatii laajan kehitystyön johtuen sen monesta eri toiminnallisuudesta. Tällaisen ongelman huomaamatta jättäminen ohjeet-sivujen kehityksen alussa muistuttaa minua projektinhallinnasta tutusta termistä ”scope creep”, mikä tarkoittaa projektin paisumista liian suureksi uusien ominaisuuksien ja lisäysten takia. Tässä tapauksessa tämä tosin pätee vain yhdelle sovelluksen ominaisuudelle eikä koko sovelluksen ominaisuuksien määrälle. Tutkiessani sopivaa termiä tälle ilmiölle, päätin lukea artikkelin Project Management Institutelta (Larson & Larson, 2009). ”Scope creep” määritellään muuten erilaisena, koko sovelluksen paisumisena ylimääräisten ominaisuuksien lisääntyessä, toisin kuin yhden ominaisuuden paisumisena. Artikkelissa mainitut syyt projektin paisumiseen tuntuvat kuitenkin pätevän myös ohjeet-sivun kehittämisen pituuteen, esimerkiksi projektin määrittelyvaiheessa oleva liian vähäinen projektin selkeys ja määrittely.

Tämä on opinnäytetyön seurantajakson viimeinen viikko, mutta työsuhteeni Jame-Shaftin kanssa tulee jatkumaan vielä kaksi viikkoa 19.3.2023 asti. Tulen kirjoittamaan näistä viikoista keskittyen työn etenemiseen, jotta voin tuoda esille loput varastohallintajärjestelmän kehityksestä.

3.8 Kahdeksas raportointiviikko

Maanantai 6.3.2023

Tämän päivän aikana keskityin ohjeiden kehittämiseen. Suurin osa tästä päivästä meni kuitenkin VPN-yhteyden, eli virtuaalisen yksityisverkkoyhteyden, aukenemiseen. Kävi ilmi, että Jame-Shaftin VPN-yhteydessä on tilaa vain kahdelle samanaikaiselle käyttäjälle. Tällä kertaa kaksi etänä työtä tekevää työntekijää olivat ehtineet minua ennen ottamaan yhteyden VPN:n kautta.

Tiistai 7.3.2023

Tämän päivän aikana jatkoin töiden logiikan testaamista ja korjaamista. Lisäsin ohjelmaan niin sanotun välimuistin, joka tallentaa kuhunkin työhön sen aikaisemmat lavapaikat. Ongelmana oli työtä sulkiessa tapahtuva lavapaikkojen varauksen jääminen tietokantaan. Tämän myötä lavapaikkahallintaan pystyi jäämään varattuja lavapaikkoja ilman työtä, jos käyttäjä asetti johonkin työhön lavapaikat, mutta sen jälkeen sulki työn, eikä tallentanut muutoksia. Kun tietokanta tallentaa kunkin työn tämänhetkiset ja vanhat lavapaikat, pystyy se muuttamaan työn vanhan lavapaikan statuksen, jos käyttäjä ei tallenna muutoksiaan.

Keskiviikko 8.3.2023

Tämän päivän aikana testasin ja korjasin eilen kehittämäni töiden välimuistia. Testauksessa ilmenneet ongelmat liittyivät lähinnä lavapaikkavalinnan ponnahdusikkunaan, jossa ponnahdusikkuna ei auennut oikeaan lavapaikkaryhmään tai hyllyyn. Sain tämän ratkaistua hakemalla kaikkien lavapaikkojen group-attribuutit ja lisäämällä ne groups-taulukkoon.

Työn muokkaamisen tai luomisen ponnahdusikkuna ei myöskään näyttänyt valittuja lavapaikkoja oikein, johtuen väärästä HTML-tiedoston ngIf-parametristä.

Torstai 9.3.2023

Tämän päivän aikana jatkoin töiden välimuistin integrointia taksijonon ponnahdusikkunoihin. Lisäsin tarvittavat funktiot töiden vanhojen lavapaikkojen käyttämiseen ja tarkistin tämän ponnahdusikkunan toiminnallisuuden. Tein nykyisen työn lavapaikan valitsemisesta taksijonon "aloita siirto" -ponnahdusikkunassa mahdotonta, jotta mahdolliset kesätyöläiset eivät pystyisi vaihtamaan töiden nykyisiä lavapaikkoja, vaan laittamaan ne

pelkästään siirtoon. Tämän jälkeen poistin kokonaan funktiot, mitkä vaihtoivat työn nykyistä lavapaikkaa.

Perjantai 10.3.2023

Tämän päivän aikana aloitin selaimen taaksepäin siirtymiseen tarkoitetun napin käytön estämistä varastohallintajärjestelmän työtä muokattaessa. Syy tähän oli se, että työn muokkaamisen ponnahdusikkuna ei palauttanut vanhoja lavapaikkoja, jos käyttäjä painoi ennen työn tallentamista selaimen taaksepäin-näppäintä. Ratkaisu tähän ongelmaan oli käyttää JavaScriptin onbeforeunload-funktiota, joka laukeaa silloin kuin sivun hahmontaminen on päättymässä. Sisällytin tähän funktioon samat funktiot, joita käytin ponnahdusikkunan sulkunäppäimessä.

3.9 Yhdeksäs raportointiviikko

Tämä viikon aikana sain viimeisteltyä ohjeet-sivun, lisättyä järjestelmään JSON-varmuuskopiot sekä paljon muita ominaisuuksia, mitä tarvitsi saada valmiiksi ennen työsuhteen päättymistä.

Maanantai 13.3.2023

Sain tämän päivän aikana ohjeiden päivityksen ja luonnin toimimaan. Suurin ratkaisu ohjeiden luonnissa oli vietävän tiedon vaihtaminen videon blob-muotoiseen dataan. Olin aikaisemmin vienyt vahingossa videon selaimessa katselemista varten luodun olio-URL:n sovelluksen taustaan, vaikka itse videon sisältämä data säilytetään blob-muodossa.

Tiistai 14.3.2023

Tämän päivän aikana muokkasin ohjeiden taustaa. Vaihdoin esimerkiksi ohjeiden tiedostojen automaattisesti generoidut nimet UUID-muotoon ohjeen

otsikon tilalle. UUID on 128-bittinen tunniste, jota käytetään yleisesti yksilöivänä tunnisteena. Ohjeiden tiedostojen nimien muuttaminen tarkoitti myös ohjeiden hakemisen parametrien muuttamista. Ohjeiden muokkaamisen lisäksi siistisin varastohallintajärjestelmän koodia poistamalla vanhoja kommentoituja koodipätkiä ja ylimääräisiä console.log-funktioita.

Keskiviikko 15.3.2023

Tämän päivän aikana korjasin vahingossa tapahtuneen Git-haarojen yhdistymisen BitBucketissa ja taksijono-sivun ponnahdusikkunan välimuistin, jossa nykyinen lavapaikka ei vapautunut, kun käyttäjä aloitti työn siirron.

Torstai 16.3.2023

Tämän päivän aikana yritin selvittää lavapaikan valitsemiseen liittyvää ongelmaa, jossa lavapaikkojen statukset muuttuvat satunnaisesti, tai ainakin hyvin epäsäännöllisesti.

Perjantai 17.3.2023

Suurin ratkaistu ongelma tämän viikon aikana oli lavapaikkavalinnan logiikan RxJS-tilausten korjaamiset. Ongelmana lopulta oli, että useat lavapaikkoja ja työjonoa tilaavat muuttujat jatkoivat niiden tilaamista, vaikka niiden tarvitsi näissä tilanteissa hakea vain ensimmäinen muuttuva arvo. Ratkaisu oli lisätä RxJS:n subscribe-muuttujan eteen pipe(x)-funktio, joka määrittää kuinka monta päivitystä muuttuja tilaa, ennen kuin se lopettaa tilauksen.

4 Pohdinta

Opinnäytetyöni tavoitteena oli kehittää varastohallintajärjestelmää ja muita Jame-Shaftin intrasovelluksia. Pääasiallinen kehityskohde oli varastohallintajärjestelmä, ja tavoitteena oli kehittää varastohallintajärjestelmään kaikki työnantajan pyytämät ominaisuudet. Näiden myötä tavoitteena oli myös pystyä käyttämään oppimaani tietoa ja oppia lisää MEAN-pinosta sekä Firebaseesta.

Opinnäytetyöjakson aikana sain suoritettua kaikki työnantajan antamat työtehtävät. Näihin työtehtäviin kuuluivat:

- varastohallintajärjestelmän Angular-version sekä kaikkien siihen kuuluvien pakettien päivittäminen
- varastohallintajärjestelmän töiden lavapaikkojen valinnan käyttöliittymäpäivitys
- muiden Jame-Shaftin sovellusten mahdollinen kehittäminen tai ohjelmavirheiden korjaus
- varastohallintajärjestelmän töiden siirtymisen logiikan testaaminen ja toiminnallisuuden varmistaminen
- varastohallintajärjestelmän ohjeiden lisäämis- ja lukujärjestelmän kehittäminen.

Sanoisin näiden työtehtävien myös onnistuneen hyvin. Suurimmat ongelmat olivat varastohallintajärjestelmän ohjeet-sivujen työmäärän suuruus verrattuna sen tärkeyteen ja varastohallintajärjestelmän lavapaikkahallinnan toimivuuden varmistaminen.

Varastohallintajärjestelmän kehittämisen myötä tämä järjestelmä pystyttiin ottamaan käyttöön osaksi Jame-Shaftin tuotantoa, ja se tehosti tuotteiden kulkua varaston hyllyjen ja työpisteiden välillä.

Kuluneen yhdeksän viikon aikana olen kehittynyt sovelluskehittäjänä monin eri tavoin. Olen oppinut uusia asioita suurimmasta osasta tekemistäni

varastohallintajärjestelmän ominaisuuksista, vaikka osa näistä oli jo aikaisemmasta kehitystyöstä tuttuja. Toisto on kuitenkin yksi tärkeimpiä osia oppimista, joten minun ei tarvitse seuraavalla kerralla luottaa yhtä paljon internetin hakutuloksiin.

Sanoisin, että merkittävin yksittäinen sovelluskehityksen osa-alue, josta olen opinnäytetyöjakson aikana oppinut, on asynkroninen ohjelmointi. Olen käyttänyt tätä varastohallintajärjestelmän kehittämisessä muun muassa ohjevideoiden viemisessä ja tuomisessa ja Firebase-tietokannan tapahtumien tilaamisessa. Olen käyttänyt asynkronisen ohjelmoinnin hoitamiseen enimmäkseen RxJS-kirjastoa TypeScriptin omien lupausten sijaan. Tämä kirjasto toimii myös muissa käyttämissäni selainpuolen kehitysympäristöissä, kuten Reactissa.

Tekemäni tehtävien lisäksi kokemuksen kerryttäminen olemassa olevien sovellusten jatkokehityksestä ja korjaamisesta oli hyödyllistä, ja uskon tämän olevan hyödyllistä myös tulevaisuudessa.

Lähteet

Adhikary, T. 2021. Theming and Theme Switching with React and styled-components. Viitattu 26.4.2023 <https://css-tricks.com/theming-and-theme-switching-with-react-and-styled-components/>.

Angular. s. a. Angular Update Guide. Viitattu 26.1.2023 <https://update.angular.io/?l=2&v=9.1-15.0>.

Angular. s. a. Migrating to MDC-based Angular Material Components. Viitattu 27.1.2023 <https://rc.material.angular.io/guide/mdc-migration>.

Colvin, T. 2021. Is Firebase Auth Secure? Viitattu 26.4.2023 <https://tdcolvin.medium.com/is-firebase-auth-secure-dace0563d41b>.

Firebase. s. a. Get Started with Firebase Authentication on Websites. Viitattu 30.1.2023 <https://firebase.google.com/docs/auth/web/start>.

Firebase. s. a. Understand Firebase Realtime Database Security Rules. Viitattu 15.2.2023 <https://firebase.google.com/docs/database/security>.

Gechev, M. 2022. Building a web application with Angular and Firebase. Viitattu 27.1.2023 <https://developers.google.com/codelabs/building-a-web-app-with-angular-and-firebase#9>.

Indeed Editorial Team. 2022. What Is Asynchronous Programming? (And When To Use It). Viitattu 26.4.2023 <https://www.indeed.com/career-advice/career-development/asynchronous-programming>.

Indeed Editorial Team. 2023. How To Use Mind Mapping in Project Management (Plus Tips). Viitattu 25.4.2023 <https://www.indeed.com/career-advice/career-development/mind-mapping-project-management>.

Larson, R. & Larson, E. 2009. Top five causes of scope creep ... and what to do about them. Viitattu 26.4.2023 <https://www.pmi.org/learning/library/top-five-causes-scope-creep-6675>.

Martins, S. 2022. Build a video streaming server with Node.js. Viitattu 22.2.2023 <https://blog.logrocket.com/build-video-streaming-server-node/>.

Patel, R. 2021. How to Create User Manual for Websites: Guide with Benefits & Steps. Viitattu 28.4.2023 <https://www.thecloudtutorial.com/how-to-create-user-manual-for-website/>.

Vassilevskiy, M. 2022. How to Stream Videos Using Javascript and HTML5. Viitattu 22.2.2023 <https://img.ly/blog/how-to-stream-videos-using-javascript-and-html5/>.

Večeřa, P. 2018. How to handle large file upload with NodeJS express server. Viitattu 24.2.2023 <https://medium.com/@vecera.petr/how-to-handle-large-file-upload-with-nodejs-express-server-7de9ab3f7af1>.

Yigit, C. 2021. The Issues I Faced When Upgrading The Angular Versions. Viitattu 25.4.2023 <https://medium.com/piateamtech/issues-i-faced-when-upgrading-the-angular-versions-99ba8700eb8e>.