Ammatillinen kehitys osana ohjelmointitiimiä

Leevi Valjakka

Haaga-Helia ammattikorkeakoulu

Tradenomin tutkinto, Tietojenkäsittely

Amk-opinnäytetyö

2025

Tiivistelmä

Tekijä(t)

Leevi Valjakka

Tutkinto

Tradenomi

Raportin/Opinnäytetyön nimi

Ammatillinen kehitys osana ohjelmointitiimiä

Sivu- ja liitesivumäärä

48

Tässä päiväkirjamallisessa opinnäytetyössä seurataan tekijän ammatillista kehitystä ohjelmointialan uran alkuvaiheessa. Seuranta toteutetaan päivätasoisin raportein, joista johdetaan viikoittainen yhteenvetoanalyysi käsitellyistä teemoista sekä kehityksestä. Ammatillisen kehityksen tavoitteiksi on valittu teknologian ja työkalujen käyttö työn tuottavuuden ja laadun parantamiseksi, sujuva yhteistyö ohjelmistokehitystiimissä, sekä työhyvinvoinnin ja yhteisöllisyyden edistäminen työssä jaksamisen tueksi.

Tekijän lähtötilanteena on ollut juuri ennen seurantajaksoa aloitettu työ pienessä ohjelmistoalan yrityksessä. Tekijällä ei ole ollut aiempaa kokemusta ohjelmistoalalla työskentelystä, ja ohjelmointikokemuksen pohja perustuu koulutöihin sekä tekijän omiin projekteihin. Tekijän työ on seurantajaksolla keskittynyt pääsääntöisesti mobiilisovelluskehitykseen React Native-teknologiaa hyödyntäen. Pienemmässä roolissa on ollut myös Microsoft Windowsille tarkoitetun sovelluksen kehitys Python-ohjelmointikielellä.

Merkittävin osa tekijän ammatillisesta kasvusta on tapahtunut teknologian ja työkalujen käytön tavoitteessa. Tekijä on oppinut sekä omaksunut useita menetelmiä tuottamansa koodin tehostamiseen, ja saavuttanut näin hyötyä tuotettavan koodin laadussa sekä työskentelyn tehokkuudessa. Näistä esimerkkinä toimii muun muassa funktioiden tulosten tallentaminen välimuistiin käyttämällä React Nativen sisältämiä metodeja. Ohjelmistokehitys on tapahtunut osana tiimiä, jonka vuoksi tekijä on myös edistänyt osaamistaan yhteistyötavoitteen osalta, etenkin versionhallinnan ja kommunikoinnin osa-alueilla. Yhteistyöllisen tavoitteen edistäminen on mahdollistanut tehokkaan ja joustavan yhteistyön ohjelmointitiimissä toimiessa. Tekijä ei ole saavuttanut merkittävää edistymistä työhyvinvoinnin ja yhteisöllisyyden tavoitteessa seuranta-aikana.

Opinnäytetyöprosessi on mahdollistanut tekijälle oman toiminnan analyyttisen tarkastelun, jonka ansiosta sitä on voitu kehittää johdonmukaisesti ongelmakohtiin reagoiden. Seurannan tulokset toimivat perustana sekä vertailupisteenä tekijän tulevaisuuden ammatilliselle kehitykselle.

Asiasanat

Ohjelmistokehitys, Versionhallinta, React Native, Python

Sisällys

1 Johdanto

Tulen raportoimaan tässä opinnäytetyössä ammatillisesta kehityksestäni ensimmäisessä tutkintoani vastaavassa työpaikassa. Opiskelen tietojenkäsittelyä, ja työnimikkeeni tällä hetkellä on Ohjelmistokehittäjä. Työnantajani on suhteellisen pieni tietotekniikan palveluita sekä tuotteita toimittava yritys.

Työtehtävieni kannalta keskeisin käytettävä teknologia on React Native, joka on mobiilisovellusten kehitykseen tarkoitettu JavaScript-kirjasto. Olen kuitenkin vielä melko alkuvaiheessa työtehtävieni osalta, ja mahdollisuuksia on myös kehittää itseään esimerkiksi PHP- sekä tietokantaosaamisen suhteen. Näistä en kuitenkaan varmuudella voi sanoa, että opinnäytetyöni seurantajaksolla saan niitä mukaan käsittelyyn. Näihin liittyvä tietoperusta tullaan keräämään pääsääntöisesti alan kirjallisuudesta, keskeisistä verkkojulkaisuista sekä blogeista.

Päiväkirjani seurantajakso alkaa maanantaista 24.1.2025, ja jatkuu tästä seuraavat 8 kalenteriviikkoa sunnuntaihin 23.3.2025. Käytännössä täten viimeinen työpäiväkohtainen merkintä tulee sijoittumaan perjantaille 21.3.2025. Laadin jokaisesta päivästä päiväkohtaisen lyhyen raportin, joista johdan viikoittaisen yhteenvedon ja analyysin.

1.1 Tavoitteet ammatilliselle kehitykselle

Ensimmäinen tavoitteistani on kehitys työlleni keskeisten teknologioiden ja työkalujen käytössä. Vaikka nämä ovat jo tuttuja opiskelun kautta, voi niiden käyttäminen tuottavassa työssä olla varsin erilaista. Ei voidakaan olettaa, että nimellisesti täysin samoja teknologioita käytettäisiin aina samoin tavoin eri toimijoiden kesken. Eri toimijoilla voi olla monia syitä käyttää hyödykseen samoja teknologioita varsin erilaisilla tavoilla. Joskus teknologiaa ei välttämättä ole valittu käyttöön vain sen yleisesti hyväksytyn sopivuuden tiettyyn ratkaisuun osalta, vaan syy voi olla esimerkiksi siinä, että projektin aloittaneet tekijät ovat yksinkertaisesti olleet sen käytöstä jo hyvin perillä, eivätkä ole tahtoneet vaivautua uuden teknologian opetteluun. Myös samasta teknologiasta voi olla monia eri versioita käytössä eri toimijoiden keskuudessa, esimerkiksi siitä syystä, että uudempaan päivittäminen edellyttäisi huomattavan osan olemassa olevasta koodiperustasta refaktorointia. Näin ollen luonnollisesti myös tämä työ tulee keskittymään siihen näkökulmaan, kuinka juuri työnantajani on valinnut teknologioita käyttöönsä ja kuinka niitä käytännössä käytetään. Tämän lisäksi teknologioiden käyttöä tullaan analysoimaan eri käyttötarkoituksissa, joita en pysty vielä raportoinnin alkuvaiheessa ennakoimaan.

Toinen tavoitteistani on yhteistyö ohjelmointitiimissä. Työtehtävät, joiden parissa ammatillista kehitystä seurataan, ovat pääsääntöisesti projektiluontoisia, ja niitä tehdään löysän hierarkian tiimissä. Tähän liittyviä käytännön haasteita ovat esimerkiksi työtehtävien jaot, yhteistyö sekä muiden auttaminen ongelmatilanteissa ja versionhallinta sekä tähän liittyvät mahdolliset ristiriidat. Yhteistyöaspektia tullaan myös tarkastelemaan eri toimijoiden välisen kanssakäymisen osalta, esimerkiksi kommunikoidessa asiakkaiden kanssa.

Kolmas työssä käsiteltävä tavoite on työhyvinvointi ja yhteisöllisyys. Työn tuottavuus kulkee käsikädessä työntekijän hyvinvoinnin kanssa. On sekä työnantajan että työntekijän vastuulla huolehtia siitä, että edellytykset pitkäaikaiseen tuottavaan työpanokseen pysyvät kunnossa. Kuormittava työtilanne heijastuu myös vapaa-aikaan ja yleiseen hyvinvointiin, aivan kuin vapaa-ajalla palautuminen ja onnellisuus heijastuu työhön ja sen tuottavuuteen. Työssä keskitytään tällä aihealueella siihen, mihin työntekijä voi itse vaikuttaa näiden tavoitteiden edistämiseksi.

Raportoinnin ulkopuolelle rajataan työmatkaan liittyvät asiat, vertailu mahdollisiin aiempiin työnantajiin, sekä työnantajan sisustusarkkitehdilliset ratkaisut.

Tavoitteet voidaan siis kootusti listata näin:

Tavoite 1: Teknologian ja työkalujen käyttö

Tavoite 2: Yhteistyö ohjelmistokehitystiimissä

Tavoite 3: Työhyvinvointi ja yhteisöllisyys

Taulukko 1. Peittomatriisi päiväkirjaopinnäytetyön tekstinsisäisistä kytköksistä

Oman ammatillisen kehittymisen tavoitteet	Seurantaviikko	Oman ammatillisen kehittymisen tulokset
Teknologian ja työkalujen käyttö	Viikot 1, 2, 3, 4, 5, 6, 7, 8	3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8
Yhteistyö ohjelmistokehitystiimissä	Viikot 1, 2, 3, 6, 8	3.1, 3.2, 3.3, 3.6, 3.8
Työhyvinvointi ja yhteisöllisyys	Viikko 1	-

Taulukko 1 sisältää peittomatriisin, josta on tarkasteltavissa seuranta-ajalle asetetut kehittymisen tavoitteet, seurantaviikot, joilla kutakin tavoitetta tarkastellaan, sekä luvut, joissa kehityksen toteutumista arvioidaan.

1.2 Keskeisimmät käsitteet ja tietoperusta

Kirjastot

Ohjelmoinnissa kirjastolla tarkoitetaan valmista koodia, jonka tarkoituksena on tuottaa ratkaisu johonkin tiettyyn tarkoitukseen taikka tavoiteltuun toiminnallisuuteen. Kirjastot virtaviivaistavat kehitystyötä vähentämällä sovelluskehittäjän tarvetta tuottaa omia ratkaisumenetelmiä eri ongelmiin (Meltzer 2023).

JavaScript

Ohjelmointikieli, jota tyypillisesti käytetään dynaamisten ja interaktiivisten toiminnallisuuksien toteuttamiseen verkkosivuilla (Mozilla s.a.).

TypeScript

JavaScript-laajennus, jonka lisäämiä ominaisuuksia ovat mm. luokat, rajapinnat sekä staattinen tyyppijärjestelmä (Haapalinna 2018). Tyypittämällä muuttujille, funktioille ja niiden palautusarvoille voidaan määrittää odotetut muodot (esimerkiksi numero, merkkijono). TypeScript mahdollistaa tehokkaan Olio-ohjelmoinnin JavaScript-syntaksilla (Deshpande 2024).

React

React on JavaScript ohjelmointikirjasto, jonka tarkoitus on virtaviivaistaa verkkokehitystyötä. Reactilla verkkosivun käyttöliittymä muodostetaan yksittäisistä komponenteista (React s.a.). React-kirjasto sisältää itsessään useita komponentteja, ja käyttäjä voi halutessaan ladata kolmannen osapuolen komponenttikirjastoja näitä täydentääkseen. Käyttäjä voi myös rakentaa omia komponenttejaan, ja hyödyntää näitä ohjelmiston eri osapuolissa. Visuaalisten komponenttien muodostukseen käytetään JSX-syntaksia (React s.a.), joka muistuttaa läheisesti HTML:ää.

React Native

React Native on kirjasto, joka mahdollistaa mobiilisovellusten kehityksen Reactia hyödyntäen. React Nativen etu mobiilikehityksessä on alustaperustaisen eriyttämisen minimoiminen: ohjelmakoodi kirjoitetaan JavaScriptillä, ja ajaessa tämä käännetään mobiililaitteen natiivikielelle. Näin esimerkiksi Android ja iOS-laitteille sovellukselle ei tarvita erillistä koodiperustaa. React Nativen JSX-syntaksi eroaa hieman Reactista, sen mukaillessa enemmän mobiilialustojen natiiveja UI-komponentteja HTML:n sijasta. (React Native s.a.).

Expo

Expo on mobiilikehitykseen käytettävä ohjelmointiviitekehys, jonka tarkoitus on helpottaa Android ja iOS sovellusten kehitystyötä (Expo s.a.).

GitHub & versionhallinta

GitHub on verkkoperustainen versionhallintapalvelu ohjelmointiprojekteille. Versionhallinnan tarkoitus on hallinnoida projektille tuotettua koodia, ja mahdollistaa usean kehittäjän itsenäinen työskentely. GitHubissa koodi on jaettu eri haaroihin, joita voidaan erottaa ja yhdistellä toisiinsa tarpeen mukaan (Kinsta 2023).

Ketterä kehitys (Agile development)

Ketterä kehitys on ohjelmoinnin projektinhallinnan menetelmä. Ketterillä menetelmillä on tarkoitus edesauttaa projektinhallinnan mukautuvuutta ja nopeaa vastetta muuttuviin tilanteisiin taikka vaatimuksiin. Ketterin menetelmin hallinnoituja projekteja toteutetaan iteratiivisesti, usein Sprinteiksi kutsutuissa ajanjaksoissa. Sprintin kesto on tyypillisesti 1–4 viikkoa. Suosittuja ketteriä menetelmiä ovat esimerkiksi Scrum ja Kanban. (Agile Alliance s.a.)

2 Lähtötilanteen kuvaus

Tässä luvussa esittelen työni sekä siihen liittyvän kokemukseni tilannetta seurantajakson alkaessa. Tämä luku toimii vertailupisteenä seurattavalle kehitykselle.

2.1 Oman nykyisen työ analysointi

Työtehtäväni tällä hetkellä keskittyvät pääsääntöisesti mobiilisovelluksen kehitykseen, johon viittaan jatkossa nimellä ”Sovellus A”. Sovellusta kehitetään alihankintana asiakasyritykselle, johon tulen jatkossa viittaamaan nimellä ”Asiakas A”. Sovellus on rakennettu React Native-teknologialla, ja sen backend on toteutettu käsitykseni mukaan pääosin PHP:llä. Toistaiseksi en ole koskenut backend-puoleen. Suurin osa työtehtävistäni on ollut tähän mennessä frontend-puolella, käyttäjälle näkyvissä ominaisuuksissa ja näitä tukevan logiikan rakentamisessa ja päivittämisessä (AirFocus s.a.). Tarpeet näille työtehtäville nousevat suurelta osin käyttäjien raportoimista bugeista, sekä vaatimuksista, joita Asiakas A välittää meille kehitettäväksi.

Kehitys tapahtuu ketterin menetelmin sprinteissä, jotka kestävät pääsääntöisesti kaksi viikkoa kerrallaan. Näiden päättyessä toteutetut muutokset toimitetaan Asiakas A:lle testattaviksi, ja ennen uutta sprinttiä on mahdollisuus vielä tehdä pieniä muutoksia asiakkaalta saadun palautteen perusteella.

Pelkän React Nativen yleisen tuntemuksen lisäksi työtehtävissä menestymisen edellytys on jatkuva oppiminen. Suuri työkuormaa lisäävä sekä sen virtaviivaisuutta rikkova tekijä on esimerkiksi React Nativen käyttämien pakettien päivitys, ja näiden muuttaessa toimintalogiikkaansa on myös usein sovelluksen rakennetta muutettava. Juuri tästä syystä tiedonhaku onkin keskeinen osa työtehtäviä: edes kokeneemmilla tekijöillä ei välttämättä ole suoraa vastausta siihen, kuinka uusimpien kirjastojen kanssa tulisi jatkossa jokin tietty toiminnallisuus toteuttaa. Keskiössä on kirjastojen virallisen dokumentaation opiskelu, mutta näiden laadun vaihtelevuuden vuoksi olennaista on myös hakukoneiden käyttö ja yhteisölliset palvelut, joista voi esimerkiksi lukea muiden samanlaisten ongelmien kanssa painivien ohjelmistokehittäjien ratkaisuja (StackOverflow, GitHub, yms.).

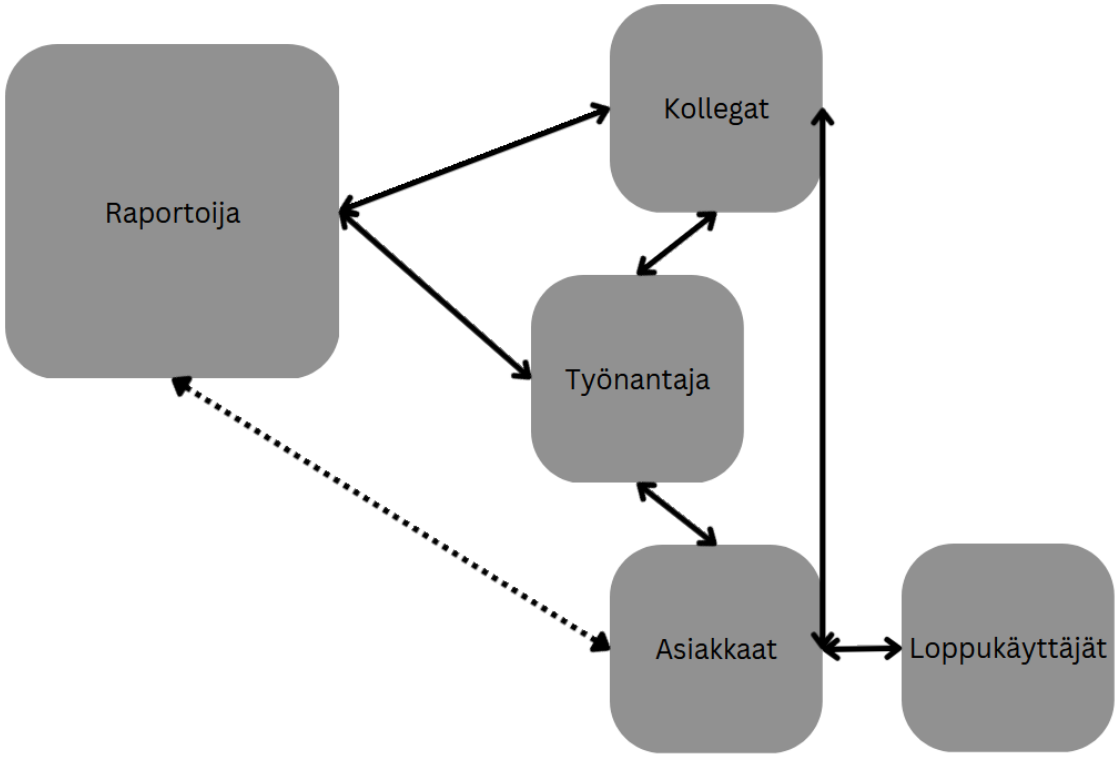
Tähän mennessä vaadittavaa osaamista on hankittu pääasiassa opiskelun ja pienimuotoisten omien projektien kautta. Sovellus on kuitenkin laajempi ja monimutkaisempi kuin mihin aiemmat projektit ovat yltäneet, ja täten sen ymmärtäminen on myös verrattain hankalampaa. Arvioin oman osaamiseni tällä hetkellä johonkin aloittelevan toimijan ja taitavat suoriutujan väliin, sillä vaikka pystyn itsenäisesti suoriutumaan suurimmasta osasta työtehtäviä, ovat tilanteet, joissa joudun kysymään neuvoja kuitenkin vielä melko yleisiä. Olen varsinkin huomannut hakevani ratkaisuja usein teknisiä ratkaisuja logiikkaan paikoissa, joissa pelkkä tyylimääritelmien muuttaminen voi jo

saada halutun tuloksen pienemmällä vaivalla. Tästä johtuen näiden tarkempi opiskelu tulee olemaan avainasemassa työni tehokkuuden kannalta.

2.2 Sidosryhmien esittely

Työhöni liittyvät sisäiset sidosryhmät ovat työnantajani sekä kollegani. Vastikkeellisessa työtehtävässä hankitun kokemukseni ollessa vielä vähäistä, työnantajalleni oletettavasti merkittävä intressi on tuottavuuteni kehitys. Myös kollegoideni osalta voidaan tunnistaa sama intressi: pystyessäni suoriutumaan paremmin töistäni, voin johdonmukaisesti keventää myös heidän työtaakkaansa ja antaa oman kontribuutioni työn sujuvuudelle.

Ulkoisista sidosryhmistä keskeisimmät ovat sekä työnantajan asiakasyritykset, joille palveluja tuotetaan, sekä näiden palveluiden loppukäyttäjät. Asiakasyrityksille merkityksellistä on toimiva ja ajallaan toimitettu tuote, joka mahdollistaa heidän oman liiketoimintansa taikka sen tukemisen. Loppukäyttäjille on myös oletettavasti hyötyä siitä, että tuottamamme palvelut helpottavat heidän elämäänsä joltain osa-alueelta.



Kuva 1: kaavio vuorovaikutuksesta sidosryhmien kanssa.

Kuva 1 havainnollistaa työn vuorovaikutussuhteitani raportoinnin lähtötilanteessa. Suoraa kommunikaatiota tapahtuu pääosin kollegoiden ja työnantajan kanssa, kun taas toistaiseksi itse asiakkaan kanssa kommunikoinnin hoitaa pääasiassa kollegani. Vuorovaikutusta asiakkaan

kanssa tapahtuu kuitenkin pienimuotoisesti, esimerkiksi vaatimuksia tarkentavien kysymyksien kautta sekä heille toimitettavien projektin työstämisen muodossa.

2.3 Työpaikan vuorovaikutustilanteet

En ole pääsääntöisesti työtehtävissäni ollut ulkoisiin sidosryhmiin toistaiseksi suorassa vuorovaikutuksessa, paitsi Asiakas A:lle kehitettävän Sovellus A:n osalta työtehtäviä koordinoidessa, ja palautetta ja siihen liittyvää lisätyötä käsiteltäessä.

Työtovereideni kanssa olen kuitenkin tiiviissä vuorovaikutuksessa esimerkiksi ongelmien ratkaisujen jakamisen muodossa. Toistaiseksi tämän osalta olen itse ollut useammin neuvoja vastaanottava osapuoli, mutta toisinaan olen päässyt myös jakamaan hyödyllisiä neuvoja ja ratkaisuja kokeneempien kollegoideni kohtaamiin ongelmiin. Vuorovaikutus on kollegoideni osalta suurimmaksi osaksi keskittynyt ohjelmointitiimiin, ja muiden kollegoideni kanssa vuorovaikutus onkin toistaiseksi ollut yleisemmällä tasolla suoraan työtehtäviin liittyvien asioiden sijasta.

3 Seurantajakson raportointi viikkoanalyyseineen

Tässä luvussa seuraan ammatillista kehitystäni konkreettisin, päiväkohtaisiin tehtäviin perustuvien päiväkirjamerkintöjen avulla. Päiväkirjamerkintöjen perusteella johdetaan viikoittainen syventyvä analyysi, jossa tarkastellaan päiväkirjamerkinnöissä käsiteltyjä teemoja ja niiden suhdetta henkilökohtaiseen ammatilliseen kehitykseen.

3.1 Seurantaviikko 1

Ensimmäisen seurantaviikon tavoitteet liittyvät työkalujen ja teknologian ymmärtämisen edistämiseen, sekä yhteistyötoimintaan ohjelmistokehitystiimissä.

Maanantai 27.1.2025

Päiväkohtainen tavoite: Työtilanteen kartoitus tiimipalaverissa, Sovellus A:n kehitystyö React Nativella

Työviikkoni alkaa tyypillisesti kaikkien paikalla olevien palaverilla, jossa käydään jokaisen henkilökohtainen työtilanne läpi. Tätä varten jokainen kirjoittaa yhteiseen tiedostoon yhteenvedon siitä, mitä edellisellä viikolla on saanut aikaan ja työstänyt, sekä arvion siitä mitä tulevalla viikolla tulee tekemään. Kirjaan tätä tarkoitusta varten oman osuuteni yrityksen jaettuun viikkopalaveritiedostoon. Palaverissa menee noin tunti, ja käymme suunnitellusti jokaisen työtilanteen läpi sekä keskustelemme yleisesti yrityksen tilanteesta. Palaverin avulla hahmotan paremmin kollegoideni työtilannetta, ja opin uutta eri tehtävistä mitä yrityksen toiminnot pitävät sisällään.

Olemme suunnitelleet päivityksiä Sovellus A:n käyttämiin kirjastoihin, mutta tästä on aiheutunut ennalta-arvaamattomia ongelmia. Minun tehtäväkseni on annettu näiden ongelmien selvittäminen. Kyseessä on useamman kirjaston versiopäivitys, ja näiden jäljiltä joidenkin kirjastojen toimintalogiikka on muuttunut. Etenkin ongelmalliseksi on osoittautunut sovelluksen sisäisten ruutujen väliseen navigaatioon tarkoitettu kirjasto, ja ilmeisesti päivityksen jälkeen muuttujien arvojen vienti näkymältä toiselle ei enää toimi kuten aiemmin. Tutkin ongelmaa käymällä sovelluskoodia läpi, ja epäilen ongelman johtuvan muuttuneesta tavasta välittää tietoa näkymästä toiselle siirtyessä. Muutan näiden tietojen käsittelyä päänäkymäsivulla, ja saan suurimman osan ongelmasta korjattua.

Tiistai 28.1.2025

Päiväkohtainen tavoite: Sovellus A:n sprintin päättäminen (yhteistyö ohjelmointitiimissä) ja siihen liittyvät tehtävät (työkalujen ja teknologian käyttö).

Sovellus A:n tämänkertainen sprintti on päättymässä pian, ja päätämme jättää ongelmallisen navigaatiokirjaston päivityksen vielä myöhempään versioon. Vaikka olen edellisenä työpäivänä edistynyt ongelmien ratkaisemisessa, eivät korjaukset ole vielä valmiita julkaistaviksi. Teimme sovelluksesta ns. staging buildin Asiakas A:n testattavaksi, ja seuraavaksi jäämme odottamaan heiltä palautetta mahdollisista bugeista, joita emme ole huomanneet kehitystyön ohessa. Työpäivään sattuu yllättävä kriisi: työtietokoneeni latauspiiri vaurioitui yllättäen. Olen lähdössä perjantaina Berliiniin kahdeksi viikoksi, tarkoituksena tehdä etätöitä sieltä käsin, joten tietokone on saatava kuntoon pikimmiten. Saan esihenkilöltäni käskyn lähteä välittömästi tietokoneeni toimittajan huoltotiskille selvittämään tilannetta. Saan uuden, alkuperäistä lähes vastaavan työkoneen, ja varmistan sen toimimisen vielä toimistolla ennen työpäivän päättymistä.

Keskiviikko 29.1.2025

Päiväkohtainen tavoite: Työvälineiden saattaminen kuntoon ja React Native kirjastopäivitysten selvitys

Työpäivän ensimmäinen puolisko kuluu tarvittavien sovellusten asennuksessa sekä asetusten säätämisessä. Minun täytyy myös kirjautua uusiksi erinäisiin verkkopalveluihin sekä tallentaa olennaisimmat kirjanmerkit verkkoselaimeeni. Olemme saaneet Asiakas A:lta listan heidän havaitsemistaan bugeista, ja koska näiden korjaaminen on ajallisesti kriittinen tehtävä, ottaa kokeneempi kollegani (tästä lähtien Kollega A) näistä vetovastuun. Tuen kuitenkin häntä parhaani mukaan etsimällä ratkaisuehdotuksia hakukoneella, ja saamme ongelmat pikaisesti korjattua. Tämän jälkeen jatkan Sovellus A:n navigointiongelmien selvittämistä, ja saan yhden siirtymän välitettävien tietojen käsittelyn taas toimimaan. Työpäivän päätteeksi ajan toteuttamani muutokset sovelluksen GitHub-repositorioon.

Torstai 30.1.2025

Päiväkohtainen tavoite: React Native navigaatio-ongelmien selvitys, sekä valmistautuminen tulevien viikkojen etätöihin yhteistyön sujuvuuden takaamiseksi.

Jatkan aamusta taas samojen ongelmien kanssa, mihin keskiviikkona lopetin. Nyt kyseessä on tilanne, johon käyttäjä päätyy erään käyttäjäpolun päätteeksi, mutta tällä hetkellä jää siihen jumiin prosessin lopuksi. Keksin tähän ratkaisun: koska olemme käyttäjäpolun lopussa, voimme huoletta

tuhota sen hetkisen näkymäpinon, ja sen jälkeen navigoida aloitusnäkymään. Toteutan tämän toiminnallisuuden sovelluksen koodiin, ja ongelma korjaantuu. Jatkan tämän jälkeen toisen tuotteemme parissa, johon viittaan tästä lähtien Sovellus B:nä. Sovellus B:lle tahdotaan tehdä laajempaa kirjastopäivitystä, ja lähden tätä toteuttamaan. Tavoitteena on päivittää Expo versioon 52, sekä kaikki sovelluksen käyttämät kirjastot joihin Expon uusi versio suosittelee päivitystä yhteensopivuuden vuoksi. Näistä merkittävin on itse React Nativen päivitys versioon 0.76, ja tältä osin päivitys hieman huolettaakin: kyseinen versio käyttää uutta arkkitehtuuria, ja on tiedossa, että tämä saattaa rikkoa joitain osia sovelluksen logiikassa. Päivitysten toteuttamisessa ja näiden testaamisessa menee muutama tunti, ja kaikki vaikuttaa toimivan kuten pitääkin. Tätä varten olen myös tehnyt Expon build-ominaisuudella uuden APK tiedoston sovelluksen testaamista varten Android-emulaattorilla. Olen tehnyt päivitystä varten erillisen haaran GitHubiin, ja koska kaikki vaikuttaa toimivan kuten pitääkin, ajan muutokset päähaaraan. Käymme vielä esihenkilöni kanssa läpi työnkulkua etänä, eritoten siihen liittyen, kuinka toimin, jos tekeminen loppuu kesken. Etätyöskennellessä spontaania kommunikaatiota ei tapahdu yhtä herkästi tämän vaatiessa esimerkiksi viestin kirjoittamista, joten tekijänä minun tulee olla entistä oma-aloitteisempi kysymään tehtäviä.

Perjantai 31.1.2025

Päiväkohtainen tavoite: Päivityksestä johtuvien ongelmien selvitystyö ensi viikolla tehtäviä ratkaisuja varten

Olen lähdössä lennolle Berliiniin n. 12:30, joten en tänään aloita mitään uutta vaativaa tehtävää. Teen taustatyötä parhaani mukaan navigaatio-ongelmien jatkoselvitystä varten, jotta pystyn ensi viikolla tarttua varsinaisiin korjauksiin tehokkaammin.

Viikkoanalyysi 1

Viikon tehtävissä on korostunut ns. debuggaus, eli ongelmakohtien tunnistus ja niihin johtaneiden syiden selvitys sekä korjaus. Tämä on tehtävänä usein turhauttavaa, sillä varsinaisesti tekijä ei pääse rakentamaan uutta toiminnallisuutta, vaan korjaamaan aiempaa sovelluslogiikkaa, joka on tässä tapauksessa kirjastojen päivityksistä johtuen vanhentunutta. Sovellus A, johon työ pääosin keskittyy, on tässä vaiheessa lähes kokonaan muiden kuin itseni kehittämä, ja sen koodikannasta on ilmeistä, että sitä on työstänyt useampi eri henkilö vuosien mittaan. Tämä ilmenee esimerkiksi vaihtelevana muuttujien nimeämisenä, suppeana taikka täysin puuttuvana kommentointina, sekä paikoittain pirstaloituneena ja lukuisiin eri tiedostoihin hajautettuna rakenteena. Tästä kuulin myös kollegoiltani, että heillä ei ole vahvaa kommentoimisen kulttuuria, ja siksi koodi voi toisinaan olla

vaikeaselkoista. Olen kuitenkin päättänyt kommentoida omaa koodiani, vähintään silloin kun tiedostan tietyn ohjelmoimani toiminnallisuuden taikka ratkaisun voivan olla toiselle tekijälle tarkoitusperältään vaikeasti hahmotettavissa. Täysin mahdollista on myös, että tämä toinen tekijä tulee esimerkiksi olemaan hieman kokeneempi itseni tulevaisuudessa, joten täysin altruistisesta motivaatiosta ei kuitenkaan ole kyse.

Vaikka tämän viikon työtehtävät ovat olleet yksipuolisia, ovat ne varmasti opettavaisia. Kokemuksieni perusteella tiedostan olevani enemmän kiinnostunut rakentamaan jotain uutta, kuin korjaamaan toimimattomia asioita. Todellisuus kuitenkin on, että olen kokemukseltani vielä urani alkuvaiheessa, ja oletan että harvassa työssä pääsisinkään heti rakentamaan uutta toiminnallisuutta ennen kuin olen muuten vakaasti perehtynyt työnantajan tuottamiin tuotteisiin ja/tai palveluihin. Tähän tarkoitusperään liittyen olen tietoisesti päättänyt olla kysymättä aivan jokaisessa ongelmatilanteessa kollegoiltani apua, sillä pienimuotoinen ‘yritys ja erehdys’’ lähestymistapa valmistaa minut tekijänä selviytymään näistä itsenäisemmin tulevaisuudessa (Mastorio 2018). Tätä menetelmää käyttäen tekijän täytyy määrätietoisesti pyrkiä selvittämään sovelluksen toimintalogiikkaa, ja pikkuhiljaa eri osien asiayhteydet alkavat selkeytymään. Itsenäinen ongelmanratkaisu auttaa kehittymään juuri itselleen olennaisimmissa ongelmissa.

Olen kaikesta huolimatta saanut viikon aikana paljon aikaiseksi, ja tunnen voivani olla tyytyväinen tuotoksiini sekä ammatilliseen kehitykseeni. Olen jo huomannut pientä tehokkuuden lisääntymistä työtehtävien suorittamisessa, ja tämän kasvutekijäksi oletan ennen kaikkea Sovellus A:n toimintalogiikan parempaa tuntemusta. Odotan seuraavalla viikolla pystyväni tarttumaan entistä haastavampiin tehtäviin.

3.2 Seurantaviikko 2

Toisen seurantaviikon päätavoitteina ovat teknologian ja työkalujen käyttö, sekä yhteistyö ohjelmistokehitystiimissä. Toista päätavoitetta tullaan tarkastelemaan erityisesti etätyöskentelyn näkökulmasta, sillä työskentelen väliaikaisesti toisesta maasta käsin.

Maanantai 3.2.2025

Päiväkohtainen tavoite: Sovellus A:n ongelmien ratkaisu, kommunikaatio kollegoiden kanssa työn fyysisestä suorituspaikasta.

Työpäivä käynnistyy koko yrityksen kattavalla viikkopalaverilla, jossa käydään läpi edellisen viikon aikaansaannokset sekä tulevalle viikolle ennakoidut tehtävät. Kertaan kollegoilleni olevani sekä kuluvan että sitä seuraavan viikon etäyhteyksin töissä Berliinistä käsin, täten toivoen välttäväni

mahdollisia ristiriitoja taikka väärinkäsityksiä työtehtävien ja näiden toteutuksen suunnittelun suhteen. Palaverin päätyttyä työ jatkuu siitä, mihin edellisellä viikolla on jääty. Olen saanut myös uuden tehtävän kollegalta: yrityksen oma tuote, Sovellus B, sisältää kirjoitusvirheitä käyttöliittymässään, ja nämä tulee korjata. Tartun toimeen, ja löydän oma-aloitteisesti samankaltaisia virheitä muualtakin, kuin mihin korjaukset on tarkalleen pyydetty tekemään. Saan tarvittavat korjaukset tehtyä nopeasti, ja ajan muutokset viipymättä Sovellus B:n GitHubiin. Tämän ollessa tehty, palaan jatkamaan Sovellus A:n versiopäivityksen aiheuttamien ongelmien kanssa. Ratkaisu alkaa olla jo lähellä, mutta viimeiset säädöt saavat jäädä seuraavaan työpäivään.

Tiistai 4.2.2025

Päiväkohtainen tavoite: Sovellus A:n ongelmien ratkaisun viimeistelyt

Jatkan heti aamusta Sovellus A:n kirjastopäivitysten ongelmien ratkomisella. Olen saanut nämä lähes selvitettyä. Suuressa roolissa näiden ongelmien ratkaisemisessa on ollut syvempi tuntemus Sovellus A:n toimintalogiikasta. Voin olla työni tulokseen tyytyväinen, sillä se on saavutettu määrätietoisesti testaamalla, rikkomalla ja jälleen korjaamalla Sovellus A:n toimintoja. Tällä tavoin olen saavuttanut syvempää tuntemusta koko sovelluksen toiminnasta. Toteutetut korjaukset ovat henkilökohtaisen työn sekä ammatillisen kasvun tuotosta.

Päivän aikana saan myös uuden tehtävän Kollega A:lta: toimeksiantajayritys pyytää selvittämään mahdollisuutta rakentaa eräs graafinen osuus Sovellus A:sta uudella modulaarisella menetelmällä. Aiempi tapa on vaatinut kovakoodatun ratkaisun jokaiseen eri tilanteeseen, mutta toiminnallisuus tahdotaan nyt saavuttaa rakentamalla grafiikka pienemmistä osista, joista valitaan renderöitäväksi vain tilanteeseen sopivat ominaisuudet. Aloitan tämän työn valmistelut siivoamalla toimeksiantajayritykseltä saadut kuvatiedostot oikeaan muotoon, jossa menee loput työpäivästä.

Keskiviikko 5.2.2025

Päiväkohtainen tavoite: Sovellus A:n toimintalogiikan syvempi tuntemus, React Native komponenttien mukauttaminen vastaamaan epätyypillisiä tilanteita.

Edellisen päivän taustatyön perusteella minulle on selvää, että modulaarisen ratkaisumenetelmän toteuttamiseksi on Sovellus A:n toiminnasta oltava varsin perusteellinen tietämys. Ensin kuitenkin on tehtävä taustalle modulaarinen auttajafunktio: tarvittava grafiikka valitaan parametrinä saatavan objektin ominaisuuksien perusteella. Toteutuksen koodaaminen on jokseenkin loogisesti haastavaa sekä työlästä, mutta saan tämän tehtyä muutamassa tunnissa. Seuraava ongelma

kuitenkin liittyy aiemmalla toteutustavalla käytetyn komponentin rajoitteisiin. Tämä komponentti ei voi ottaa kuin yhden kuvan parametrikseen, joten sen avuksi tarvitaan uusi komponentti, joka sisältää tarvittavat kuvat. Keskustelen asiasta Kollega A:n kanssa, jonka perusteella nousee ajatus vaihtoehtoisesta toteutustavasta. Vaihtoehtoinen toteutustapa toimiessaan tekisi aiemmin kirjoitetusta auttajafunktiosta tarpeettoman. Päätän ensiksi tehdä vaihtoehtoisen toteutuksen käytännöllisyydestä testin, hyvin rajoitetulla toiminnallisuudella. Vaihtoehtoinen tapa osoittautuu päteväksi, mutta vielä on epäselvää, kumpi tapa on lopulta parempi. Päätän jatkaa vaihtoehtoisen toteutuksen rakentamista ensi viikolla, sillä torstain ja perjantain olen vapaalla.

Torstai 6.2 & perjantai 7.2.2025 ei raportointia – raportoija vapaalla

Viikkoanalyysi 2

Viikon tavoitteita on edistetty johdonmukaisesti, tavanomaisesta lyhyemmästä työviikosta huolimatta. Saavutettujen tulosten perustana töissä ovat olleet sekä teknologiaan että sen käyttökohteeseen – erityisesti Sovellus A:han – keskittynyt itseopiskelu ja tutkiskelu. Sovelluksen toimintaperiaatteiden selvittäminen on mahdollistanut johdonmukaisen lähestymistavan ongelmien ratkaisemiseen, koodissa olevien toiminnallisuuksien ja näiden keskinäisten riippuvuuksien ollessa selvillä. CodeAcademyssä julkaistussa blogikirjoituksessa “How to Work With Code Written by Someone Else” (Thorndyke 2021) listataan useita menetelmiä toisen henkilön kirjoittaman koodin työstämiseen. Näistä on löydettävissä yhtäläisyyksiä viikon aikana käytettyjen työskentelymenetelmien kanssa. Ensimmäinen kohta – “It's only code” – keskittyy siihen, että koodi on epäselkeydestään tai hienostuneisuudestaan riippumatta aina mahdollista ymmärtää lopulta. Väite tukee omia havaintojani Sovellus A:n kanssa työskentelystä. Itse koodin ymmärtämiseen olennaisempi kohta on kolmas kohta “Run the tests”. Juuri erilaisia testejä suorittamalla koodin kryptisemmät osuudet ovat tulleet minulle tutuiksi. Käytännön tasolla tämä on tarkoittanut esimerkiksi jonkin tietyn koodin osan rikkomista ja sen vaikutusten havainnoimista sovellusta käytettäessä, sekä console.log ominaisuuden käyttämistä eri paikoissa, jotta on havaittavissa, milloin mikäkin osa ohjelman koodista ajetaan.

Viikon toista päätavoitetta, yhteistyö ohjelmistokehitystiimissä, on myös edistetty. Ulkomaanmatkani vuoksi työt on tällä viikolla suoritettu etänä, ja keskipiste tämän tavoitteen suhteen onkin luonnollisesti ollut etäkommunikaatiossa. Kommunikaatio ei ole ollut pelkästään kuulumisten vaihtoa, vaan etenkin muiden tiimiläisten pitämistä kartalla omista työn alla olevista tehtävistä ja niiden kanssa edistymisestä. On myös ollut aihetta olla yhteydessä kokeneempiin ohjelmoijiin esimerkiksi tilanteissa, joissa on punnittu parasta lähestymistapaa ongelmaan, johon on ilmennyt useampi ratkaisutapa. Linezeron blogissa “Top 10 Tips for Working Remotely” (Shah

2024) listataan nimestään poiketen viisi vihjettä työntekijälle etätyöskentelyn tuottavuuden ylläpitämiseen. Blogissa kehotetaan esimerkiksi tarkoituksenmukaiseen ylikommunikaatioon. Uskon tämän olevan tarkoituksenmukaista etenkin tapauksessa, jossa työntekijä on vielä suhteellisen aikaisessa vaiheessa uraansa. Lähityöskentelyssä ns. hiljainen tieto siirtyy työntekijältä toiselle lähes huomaamatta, mutta etätyöskennellessä tätä ei tapahdu yhtä helposti (Froehle 2023). Uransa alkuvaiheessa olevalle työntekijälle kokeneempien kollegoiden neuvot ovat kuitenkin mittaamaton resurssi, ja yksi kommunikaation tavoitteista on ollut tämän hyödyntäminen. Kommunikaatio on tapahtunut pääsääntöisesti yrityksen Slack-kanavalla, niin yleisillä keskustelukanavilla kuin yksityisviestein. Tarkoituksena on myös ollut pitää muut perillä omasta työtilanteestani, sekä reagoida uusiin nouseviin tehtäviin, jotka voivat olla kiireisempiä kuin nykyisellään työn alla olevat.

3.3 Seurantaviikko 3

Kolmannen seurantaviikon keskeisimmät tavoitteet liittyvät React Nativen syvempään tuntemukseen sekä tämän hyödyntämiseen teknisissä ratkaisuissa, ja rajatummin sisäisten sekä ulkoisten sidosryhmien kanssa kommunikointiin ja yhteistyöhön.

Maanantai 10.2.2025

Päiväkohtainen tavoite: oman React Native-komponentin rakentaminen ja siihen liittyvät tekniikat.

Kolmas seurantaviikko alkaa perinteisesti viikkopalaverilla. Työntekijänä tehtäväni on jälleen pohtia edellisen viikon saavutuksia, ja tehdä niistä tiivistelmä palaverissa jaettavaksi. Kirjoitan nämä ylös, samalla hahmotellen viikon kulkua mielessäni. Olen saanut viikonlopun aikana muutaman idean siitä, kuinka voisin jatkaa edellisen viikon työtäni. En ole tarkoituksenomaisesti häirinnyt viikonloppulepoani miettimällä töitä, nämä ajatukset ovat nousseet päähäni spontaanisti. Olen kirjoittanut mieleeni tulleet huomiot pikaisesti ylös, jättäen niiden kriittisemmän tarkastelun varsinaiselle työajalle. Jaan jo palaverissa kollegoilleni pari ratkaisua, jotka tuntuvat omasta mielestäni potentiaalisimmilta, ja käyn töihin näiden parissa sen päätyttyä. Tavoitteenani on valmiin kirjaston sijasta rakentaa oma komponenttini, joka toteuttaa haluamamme toiminnallisuuden. Työn määrä on suuri, mutta viikonloppuna saamieni ideoiden ansiosta pystyn aloittamaan toteuttamisen määrätietoisesti. Olen hahmotellut komponentin rakenteen yleisellä tasolla, ja tavoitteenani on tehdä tästä raakile työn pohjaksi. Varsinaiset yksityiskohdat toteutan tämän päälle, tehden komponentista asteittain monimutkaisemman sitä mukaa kun saan eri osioita valmiiksi. Saan päivän loppuun mennessä rajoitetusti toimivan version valmiiksi. Olen työpäivääni hyvin tyytyväinen, sillä huomisen aikataulussa on palaveri Asiakas A:n kanssa, ja odotan pääseväni esittelemään tuotostani tässä yhteydessä.

Tiistai 11.2.2025

Päiväkohtainen tavoite: kommunikointi asiakasyrityksen kanssa, ideoiden jatkokehitys tiimityöskentelynä, React Native-komponentin rakentaminen

Aamuni alkaa palaverilla Asiakas A:n sekä kollegoideni kanssa. Käymme läpi Sovellus A:n seuraavan sprintin tavoitteita, ja vastaamme parhaamme mukaan Asiakas A:n näistä esittämiin kysymyksiin. Nämä koskevat lähinnä näkemystämme eri tehtävien vaativuudesta, sekä näiden toteuttamiseen kuluvan ajan arvioinnista. Palautteemme mukaan tehtävät joko hyväksytään suoraan sprintille, jätetään siltä pois, tai siirretään vielä Asiakas A:n omaan arviointiin tarpeellisuudesta. Pääsen myös esittämään oman arvioni rakentamani toiminnallisuuden haasteista sekä ajallisesta resurssitarpeesta. Aiempi työni tähän liittyen on ollut kenties jossain määrin “proof of concept”-tyyppistä, jota olen työstänyt asiakkaan suostumuksella. Esitän oman ratkaisuni toiminnallisuuden toteuttamiseksi, ja päätämme jatkaa tällä menetelmällä pienin muutoksin. Kollegani ehdottaa pientä muutosta mahdollisen tulevaisuuden jatkokehityksen varalle, ja tämä käy myös asiakkaalle. Tarvitsen tätä varten kuitenkin lisämateriaalia asiakkaalta, ja tämän toimitusta odotellessani jatkan aiempaa työtäni komponentin parissa. Logiikka tulee toimimaan muutoksista huolimatta pitkälti samalla tavalla, ja jatkan sen rakentamista vastaamaan tarvitsemiamme tilanteita. Saan tarvitsemani lisämateriaalin vasta myöhään iltapäivästä, joten jätän sen käsittelyn seuraavaan päivään.

Keskiviikko 12.2.2025

Päiväkohtainen tavoite: React Native tyylitiedostojen tehokas käyttö

Olen edistynyt edellisenä päivänä huomattavasti komponentin toimintalogiikan kanssa, joten tänään keskityn komponentin sovittamiseen lisämateriaalin kanssa toimivaksi. Suurimmalta osin asian suhteen ei ole haasteita: kuten arvelin, komponentin logiikkaosuus toimii varsin hyvin tässäkin tapauksessa. Komponentin visuaalisen osan suhteen joudun kuitenkin tekemään asiat hieman monimutkaisemmin, joten pieni React Native tyylittelyn kertaus on paikallaan. Tyylittely toimii pitkälti samalla tavoin kuin CSS, pienin muutoksin. Elementeille määriteltävät ominaisuudet eroavat CSS:stä hitusen, ja käytän apunani internetistä löytyvää lunttilappua tarvitsemiani etsiessäni. Päivän työt koostuvat itseopiskelusta, ratkaisuiden toteuttamisesta ja toimivuuden arvioinnista. Saan päivän loppuun mennessä komponenttini toimimaan melko hyvin, ja huomiselle jää vielä pientä säätöä. Olen kuitenkin huomannut jotain huolestuttavaa: testaillessa sovellus kaatuu toisinaan, kertoen virheeksi muistin loppumisen.

Torstai 13.2.2025

Päiväkohtainen tavoite: Komponentin testaus, React Native muistivuotojen tunnistaminen ja korjaus, sekä React Native koodin optimointi.

Rakentamani komponentti on melkein valmis, tarviten enää hieman säätöä. Ennen kaikkea se tarvitsisi kuitenkin vielä muutaman käyttötilanteen testausta, joka on haastavaa etätyöskentelystäni johtuen. Tarvitsisin esimerkiksi tietynlaisen laitteen sovelluksen testaamiseen, mutta minulla ei ole tätä matkalla mukanani. Päätän siis jättää tältä osin testaamisen seuraavaan viikkoon, kun pääsen taas toimistollemme käymään. Korjaan muutaman havaitsemani virheen komponentissa, mutta keskittymistäni häiritsee huoli komponenttini aiheuttamasta muistivuodosta. Tämän tunnistamiseen epäilen tarvitsevani debuggaustyökalua, joka onneksi löytyy Exposta sisäänrakennettuna. Tämän käytössä on kuitenkin ongelma: työkalu menettää yhteyden sovellukseen ennen kuin ajamani analyysit ehtivät valmistua, ja näiden tulokset menetetään samalla. Asia turhauttaa sekä häiritsee keskittymiskykyäni entisestään. Kysyn Slackin kautta kollegoiltani, mikäli heillä olisi vaihtoehtoista tehtävää, jota voisin tehdä välissä. He ovat kuitenkin ilmeisesti palaverissa, enkä saa heitä kiinni ennen työpäivän päätöstä.

Perjantai 14.2.2025

Päivä menee Suomeen matkustaessa, joten tältä päivältä ei työhön liittyvää raportoitavaa.

Viikkoanalyysi 3

Mennyt viikko oli ammatillisen kasvun kannalta monipuolinen. Työkalujen ja teknologioiden osalta se sisälsi haasteita, turhautumista, oppimista sekä onnistumista. Olennainen osa viikon tehtäviä oli myös kommunikaatio sisäisten sekä ulkoisten sidosryhmien kanssa, liittyen tavoitteeseen yhteistyöstä ohjelmointitiimissä. Sisäisesti kommunikaatio on ollut kollegoiden konsultoimista ja ideoiden vaihtamista työtehtäviin liittyen. Ulkoinen kommunikaatio on pääasiassa ollut keskustelua asiakasyrityksen kanssa tuotteesta, jota heille toimitamme alihankintana. Aiemmin olen ollut lähinnä kuuntelemassa kollegoideni keskustelua asiakkaan kanssa, mutta tällä viikolla olen ensimmäistä kertaa päässyt myös itse antamaan panokseni yhteiseen kokoukseemme. Olen aiemmin vaihtanut yksittäisiä viestejä heidän kanssaan mm. Slackissa, kysyen esimerkiksi tarkennuksia heidän määrittämiin vaatimuksiin. Kokouksessa tehtäväni on ollut esitellä hahmottelemaani ratkaisua heidän määrittämälle toiminnallisuudelle, sekä antaa aika-arvio sen toteuttamiseen kuluvasta ajasta. Epäröin hieman aika-arvion antamisen kanssa, mutta koen päässeeni siinä suhteellisen realistiseen lukuun.

Olen alkanut hahmottamaan viikoittaisen tiimipalaverimme etuja konkreettisemmin: samalla kun kirjoitan aikaansaamiani asioita ylös, hahmotan, kuinka paljon aikaa olen käyttänyt mihinkin.

Muistelen samalla tehtäviin liittyviä ongelmia, ja mietin, kuinka olisin voinut tehokkaammin pureutua niiden aiheuttamiin ongelmiin. On myös mielenkiintoista verrata pöytäkirjaamme kirjaamiani tavoitteita tulevalle viikolle siihen, mitä olen varsinaisesti päätynyt tekemään. Edellisen viikon osalta tämä oli hyvin lähellä ennakoimaani, mutta aiempina viikkoina on usein noussut jotain odottamatonta työlistalle.

“Proof of conceptin” (PoC) tarkoitus on testata idean toteuttamiskelpoisuutta. Sen päämääränä on näyttää, että idea on rakennettavissa, että se toimii ja että se voi täyttää jonkin käyttötarkoituksen. PoC:n avulla on siis testattavissa, onko jokin idea mahdollisesti järkevää toteuttaa suhteessa käytettävissä oleviin resursseihin (Geeks for Geeks 2024). Tämä oli myös oma tarkoitukseni viikon keskeisimmän tehtävän kannalta. Oli ennakoitavissa, että tehtävä on melko työläs, mutta tarkemmasta resurssitarpeesta tai parhaasta toteutustavasta ei ollut vielä varmuutta. Tuotin kaksi vaihtoehtoista menetelmää halutun toiminnallisuuden rakentamiseksi. Sama toiminnallisuus oli toteutettu myös jo ennestään kehitettävään tuotteeseen, mutta menetelmä oli hyvin kömpelö ja jatkokehityksen kannalta haastava. Ensimmäinen menetelmäni rakensi tämän päälle, prosessia yksinkertaistaen. Pian tuli kuitenkin selväksi, että toteutustapa olisi riippuvainen monesta muusta osatekijästä, ja vaatisi täten huomattavasti ajallisia resursseja sen toteuttamiseksi. Ei myöskään ollut varmaa, olisiko ratkaisu lopulta aiempaa menetelmää yksinkertaisempi ja soveltuvampi jatkokehitykselle. Sain kollegaltani vinkin hänen aiemmin lyhyesti testailemasta vaihtoehtoisesta tavasta, ja päätin tutkia tätä toteutustapaa. Työ tuotti tulosta, ja tämä toinen menetelmä osoittautui edellistä paremmaksi: tapa on modulaarinen sekä yksinkertainen, joten sen jatkokehitys ja ylläpito tulisi olemaan huomattavasti toista vaihtoehtoa helpompaa. Sain rakennettua tästä rajoitetusti toimivan komponentin, jonka selostin lyhyesti palaverissamme Asiakas A:n kanssa. Päätimme jatkaa tällä menetelmällä. Koen prosessin olleen tuottoisa ammatillisen kasvuni suhteen. Vaikka sain hieman vihjettä vaihtoehtoiseen toteutustapaan, olen kuitenkin tehnyt varsinaisen rakennustyön sekä menetelmien arvioinnin suhteellisen itsenäisesti. Olen esitellyt menetelmät kollegoideni ja asiakkaan keskuudessa, sekä perustellut valitsemani toteutustavan verraten molempien vahvuuksia ja heikkouksia. Jatkossa uskallan tarttua samankaltaisiin haasteisiin entistä suuremmalla itsevarmuudella.

Viikko on kuitenkin myös sisältänyt stressiä, johtuen ongelmista aiemmin mainitun selvitystyön kanssa. Tarkoituksenani on ollut selvitä prosessista mahdollisimman itsenäisesti, joten menetelmien suunnittelu ja arviointi on vaatinut paljon selvitystä sekä pohdintaa. Välillä olen ollut melko varma jonkin tietyn idean toimivuudesta, mutta huomannut, ettei käytäntö olekaan aivan yhtä yksinkertaista kuin olin odottanut. Tämä on toisinaan aiheuttanut turhautumista, mutta ei kuitenkaan lannistumista. Näitä tilanteita olen käsitellyt enemmän opetuksellisina, tietäen että pystyn jatkossa entistä kokeneempana ottaa useampia asioita suunnittelussa huomioon. Viikko on täten nähdäkseni ollut varsin tuottoisa tavoitteideni suhteen. Oman jaksamiseni kannalta olen

kuitenkin tunnistanut tarpeen opiskella stressinhallintaa, kenties juuri ohjelmistokehityksen tuomien haasteiden näkökulmasta.

3.4 Seurantaviikko 4

Neljännen seurantaviikon päätavoite on React Nativen sekä JavaScriptin syvempi tuntemus ja tämän vieminen käytäntöön työtehtävissä.

Maanantai 17.2.2025

Päiväkohtainen tavoite: Komponenttini viimeistely

Viikoittaisessa palaverissamme käydään kaikkien työtilanteet läpi, sekä edellisellä viikolla työstetyt asiat että alkavan viikon suunnitelmat. Jatkan tämän jälkeen rakentamani komponentin parissa. Se alkaa olla suurilta osin valmis, mutta edellisellä viikolla havaitsemani ongelma muistinkäytön kanssa aiheuttaa huolia. Ensi töiksi kuitenkin korjaan pari huomaamani virhettä toimintalogiikassa. Pyrin tämän jälkeen toistamaan muistiongelmaa, mutta en saa ongelmaa tällä kertaa toistumaan. Tiedän kuitenkin, ettei se voi olla ilman mitään toimenpiteitä ratkaistu, joten ryhdyn tutkimaan eri menetelmiä muistinkäytön optimoimiseksi. Yksi varteenotettava ratkaisu voisi olla niin sanottu komponentin memoaminen. Tyypillisesti React Native päivittää visuaaliset komponenttinsa sovelluksen tilan (”state”) muuttuessa. Kyseessä on ominaisuus, joka yleensä on hyödyllinen – kun olennaiset muuttujat saavat uusia arvoja, käyttäjälle näytettävä tieto päivittyy. Koodi ei kuitenkaan aina pysty automaattisesti tunnistamaan, onko päivitykselle tosiasiallista tarvetta. Joskus muuttujalle asetetaan täysin edellistä vastaava arvo, ja tämä aiheuttaa komponentin päivityksen itse sisällön pysyessä samana. Tilanteessa, jossa näitä komponentteja on useita, voi tämä toiminta etenkin olla epätoivottua. Rakentamastani komponentista näytetään useita eri parametrein luotuja kopioita samanaikaisesti, ja näiden uusiksi renderöinti kuluttaa turhaan resursseja. Päätän laittaa nämä React Nativen memo-metodin sisään, jolloin näin määritellyt komponentit renderöivät itsensä uusiksi ainoastaan tiettyjen ehtojen täyttyessä. Ehdin alustamaan tämän toteuttamisen, mutta loppu jää huomiselle.

Tiistai 18.2.2025

Päiväkohtainen tavoite: React Native suorituskykyoptimoinnin opiskelu ja tämän vieminen käytäntöön komponentillani, omaan työhöni liittyvän vertaispalautteen käsittely

Olen saanut illalla Slackiin viestin kollegaltani – hän oli testannut komponenttiani tuotantoympäristössä, jolloin niistä on moninkertainen määrä kopioita samanaikaisesti käytössä kehitysympäristöön verrattuna. Hän kertoo sovelluksen muuttuvan tällöin toiminnaltaan erittäin hitaaksi. Hetkeksi huolestun – onko ideani ollut täysi susi, ja yli viikon työ mennyt hukkaan? Päätän

kuitenkin vielä kokeilla eilen aloittamani ratkaisun toteutusta. Aluksi memo-metodista ei tunnu olevan juurikaan apua, ja ongelma toistuu jatkuvasti. Kokeilen säätää uudelleenrenderöintisääntöjä useampaan otteeseen, mutta komponentit renderöityvät vieläkin turhaan uusiksi. Vaikka dokumentaatiota löytyy, se ei vastaa aivan täysin omaa käyttötilannettani, joten rupean kokeilemaan eri asioita käyttäen lokitoimintoja monipuolisesti avuksi. Nyt olen varma, että renderöintiehtoni ovat varsin pätevät, mutta vieläkään ne eivät toimi niin hyvin kuin pitäisi. Käännyn taas dokumentaation puoleen ja huomaan vihdoin virheeni – käytän komponenttini sisässä Reactin useState metodia, jolla asetetaan muuttuja ns. tilamuuttujaksi. Tilamuuttujan muuttuva arvo pakottaa React & React Native komponentin päivittymään aina, memo-metodista riippumatta. Muokkaan komponenttiani toimimaan ilman tilamuuttujia, ja vihdoin saavutan läpimurron. Sovelluksen suorituskyky paranee lähes lähtötilannetta vastaavalle tasolle. Korjaukseni on kuitenkin rikkonut joitain osia komponentin toiminnallisuudesta, joten päätän jatkaa näiden korjaamista huomenna.

Keskiviikko 19.2.2025

Päiväkohtainen tavoite: React Nativen syvempi tuntemus ja komponentin toimintalogiikan järkeistäminen.

Komponenttini ulkoasun muuttuminen oli aiemmin hyvin riippuvainen tilamuuttujista, ja tilamuuttujien käytöstä pois siirtyminen on rikkonut tähän liittyvän toiminnallisuuden. Olin alun perinkin käyttänyt tilamuuttujaa lähinnä siksi, että komponentin saama parametri ei jostain syystä saanut oikeaa arvoa ensimmäisen renderöinnin aikana. Tällöin tilamuuttujan päivittyminen pakotti uudelleenrenderöinnin, kun arvo vihdoin asetettiin. Alan tarkastelemaan kuinka välitän parametrit parent-komponentilta tälle omalleni, ja huomaan optimoimisen olevan mahdollista. Järkeistän hieman näitten välittämistä, ja saan palautettua toiminnallisuuden ilman tilamuuttujan käyttöä. Teen myös muuta pientä siivousta, ja saan komponentin suorituskykyä hieman parannettua entisestään.

Torstai 20.2.2025

Päiväkohtainen tavoite: Oman työn ongelmatilanteista keskustelu sekä yhteistyöllinen ongelmanratkonta ja päätöksenteko.

Käytettyäni nyt tällä viikolla uutta komponenttiani tuotantoa vastaavassa ympäristössä, olen huomannut lisää puutteita sen toiminnassa. En aiemmin muita ongelmia ratkoessa ollut kiinnittänyt huomiota siihen, että vaikka komponentti toimii lähes täydellisesti ilman vaikutusta sovelluksen suorituskykyyn, kestää sen alustuksessa ilmeisesti hetki. Käytän tuotantoympäristöä sovelluksessa kehitysversiolla, ja on tyypillistä, että kehitysversion suorituskyky on tuotantosovellusta huonompi. Komponenttini alustuksessa kuitenkin kestää niin kauan, että pelkään sen olevan

loppukäyttäjällekin ilmeistä, etenkin vanhempaa tai halpaa mobiililaitetta käytettäessä. Keskustelen Kollega A:n kanssa huolistani tähän liittyen, ja kerron, etten näe komponenttiani valmiina tuotantosovellukseen vietäväksi. Tutkimme yhdessä ratkaisuja ongelmaan, ja huomaamme, ettei ongelmani ole uniikki – usea muu kehittäjä on jakanut omia ongelmiaan samankaltaista tarkoitusperää toteuttavalla komponentillaan. Ongelma näyttää siis johtuvan ennemminkin React Native-teknologiasta, eikä omasta osaamisestani.

Perjantai 21.2.2025

Päiväkohtainen tavoite: Oman toiminnan suunnittelu, vastoinkäymisen ja sen aiheuttaman turhautumisen käsittely.

Keskityn vaihteeksi muihin työtehtäviin kuin komponenttiini. Olemme saaneet asiakkaalta uusia tehtäviä Sovellus A:han liittyen, ja ryhdyn mielelläni työstämään näitä. Komponenttini ongelmat ovat saaneet aikaan lievää turhautumista, ja olen onnellinen vaihtoehtoisesta tekemisestä. Saan tehtävät hoidettua muutamassa tunnissa. Seuraavaksi keskustelen jälleen Kollega A:n kanssa komponentistani – tulemme yhdessä siihen tulokseen, että sen toteutukseen on löydettävä vaihtoehtoinen tapa. Tästä johtuva pettymys häiritsee hetkellisesti kykyäni keskittyä, enkä saa hetkeen mitään tuottavaa aikaan. Iltapäivällä onneksi kykenen aloittamaan vaihtoehtoisen toteutustavan suunnittelun, ja alan kirjoittamaan suunnitelmaa tätä varten ylös. Ollessani tyytyväinen suunnitelmaani, suljen tietokoneeni valmiina aloittamaan toteuttamisen ensi viikolla.

Viikkoanalyysi 4

Neljäs seurantaviikko on ollut työtehtäviltään melko yksipuolinen. Suurin osa työajasta on keskittynyt päivämerkinnöissä käsitellyn komponentin rakentamiseen. Tämän myötä olen edistynyt etenkin ensimmäisen tavoitteen – Teknologian ja työkalujen käyttö – suhteen.

Komponentin rakentaminen on velvoittanut opiskelemaan monia React Nativeen ja JavaScriptiin liittyviä ominaisuuksia sekä niiden etuja ja ongelmakohtia syvällisesti. Olen kohdannut ennalta-arvaamattomia tilanteita sekä vastoinkäymisiä. Kokemukseni mukaan haastavien tilanteiden itsenäinen ratkaisu luo parhaat edellytykset ammatilliselle kasvulle. Vielä tässä vaiheessa uraani on tietojen haku olennainen osa ongelmien ratkaisemista. Tämänkin suhteen olen kehittynyt: osaan rakentaa hakukonekyselyni paremmin täsmällisempien tulosten löytämiseksi, eikä aikaa tuhlaannu epäolennaisten tulosten läpiselaamiseen. Hakukoneiden taidokasta käyttöä kutsutaan leikkimielisesti ”Google-Fu”-ksi, ja se on olennainen osa ohjelmistokehittäjän henkilökohtaista työkalupakkia (Xu 2021).

Olen myös edistänyt tavoitettani yhteistyötoiminnasta ohjelmistokehitystiimissä. Olen aikaisemmin esimerkiksi hieman arkaillut kysyä liikoja neuvoja kollegoiltani, kenties peläten leimaantumista

heidän silmissään avuttomaksi. Nämä tuntemukset ovat uranvaihdon alkuvaiheessa kuitenkin normaaleja (N. 2023). Tärkeää on, ettei näistä tuntemuksista lamaannu, vaan määrätietoisesti rakentaa omaa ammatillista itsetuntoaan. Olen tämän mukaisesti rohkeammin kysellyt sekä kyseenalaistanut ohjeita kollegoiltani, ja olen huomannut, että vuorovaikutussuhteiden osalta vaikutukset ovat ennemminkin olleet positiivisia. Jokainen keskustelu madaltaa kynnystä tuleville vuorovaikutustilanteille, ja myös kollegani ovat paremmin perillä kyvyistäni. Pelkkä tehtäviin liittyvien ongelmien ääneen ratkominen toisinaan saa minut myös huomaamaan, että tiedänkin ratkaisun ongelmaan jo itse, vaikken tätä itsekseni mietiskellessä vielä hahmottanutkaan.

Yhteisöllisyyden osalta on tavoitteiden edistämisessä vielä petrattavaa. Vuorovaikutukseni tähän mennessä on suurimmalta osalta töissä ollut vahvasti työtehtäviin liittyvää. En ole tätä tietoisesti vältellyt – kenties ennemminkin luonnollisia tilanteita vapaampimuotoiselle keskustelulle ei yksinkertaisesti synny kovinkaan usein. Tämä johtunee ohjelmointityön luonteesta, jota tehdään ensiksi pään sisällä suunnitellen, ja tämän jälkeen päätelaitteella näppäillen. Usein huomaankin pohtivani, viitsinkö kysyä kollegaltani juuri nyt jotain, sen varalta, että saatan mahdollisesti katkaista pitkän ajatusprosessin epäotolliseen aikaan. Tätä olen huomannut omassakin työskentelyssä, ja välillä voi olla vaikeaa päästä takaisin omaan ajatuksenkulkuun käsiksi, jos se katkeaa kriittisessä kohdassa. Tästä huolimatta epämuodollisiakin keskustelutilanteita on syntynyt. Keskustelimme matkastani kollegoideni kanssa heti ensitöiksi viikon alussa, ja siirryimme tästä keskustelemaan mm. elokuvista ja vanhojen mykkäfilmien erikoistehosteista. Toisinaan yhteisiä keskustelunaiheita löydämme myös esimerkiksi musiikista sekä videopeleistä.

Vaikka viikon päätteeksi tulimmekin siihen tulokseen, ettei rakentamaani komponenttia kannata vielä nykyisessä muodossaan viedä tuotantoon, olen itsekin yllättynyt, kuinka vähän asia lopulta on minua vaivannut. Olen tätä rakentaessa opiskellut monipuolisesti React Nativeen ja Javascriptiin liittyviä tekniikoita ja etenkin ”best practice” käytäntöjä suorituskyvyn ja ylläpidettävyyden edistämiseksi. Näissä toimissa kerrytetty käytännön kokemus on parantanut luottamustani omaan ammatilliseen taitooni. Asioiden toteuttamiseen ryhtyminen on sitä helpompaa, mitä paremmin niihin liittyvät tehtävät ovat etukäteen suunniteltu (Messaoudi 2021). Tätä tarkoitusperää varten olin suunnitelmanikin laatinut vuokaavion omaisesti. Suunnitelman eri osa-alueet olivat pilkottu pieniksi tehtäviksi, ja näiden keskinäiset riippuvuudet kirjoitettu ylös selvästi. Erittelin tehtävien järjestyksen, ja mistä osatehtävistä mikäkin tehtävä oli riippuvainen. Määrittelin myös etukäteen, missä muodossa tulee jonkin tietyn osuuden koodista välittää dataa toiselle osuudelle, ja mitä tämän datan tulee sisältää. Tämän kokemuksen perusteella olen päättänyt jatkossa panostaa entistä enemmän toimintani suunnitelmallisuuteen. Pystyin hyvin laaditun jatkosuunnitelmani ansiosta jäämään viikonloppuvapaalle rennoin mielin – olihan selvää, miten ensi viikolla töissä jatkaisin vastoinkäymisestä riippumatta.

3.5 Seurantaviikko 5

Viikkotavoite: Osaamisen monipuolistaminen työkalujen ja teknologioiden käytössä

Maanantai 24.2.2025

Päiväkohtainen tavoite: Työkalujen ja teknologioiden käyttö, etenkin refaktorointi sekä skriptien muodostaminen.

Päivän ensimmäinen puolisko kuluu pääsääntöisesti komponenttini koodia refaktoroidessa. Pyrin tekemään koodistani selkeää ja helpposelkoista. Ajatuksenani on, että mikäli tähän on aiheellista palata joskus tulevaisuudessa, olisi sen hyvä olla helposti ymmärrettävässä muodossa. Moni eri osa koodista, jotka ovat minulle nyt selkeitä, voivat olla esimerkiksi vuoden tauon jälkeen jo huomattavasti vaikeampia ymmärtää tarkoitusperiltään ja keskinäisiltä riippuvuuksiltaan. Samalla kuitenkin toivon löytäväni ratkaisun suorituskykyongelmiin, mutta tämän varaan en voi laskea. Valitettavasti parannuksillani on kuitenkin vain marginaalinen vaikutus suorituskykyyn. Iltapäivällä asiakas välittää meille lisää tehtäviä Sovellus A:han liittyen, ja aloitan näiden toteuttamisen. Kyseessä on pientä käyttöliittymäpäivitystä sovellukselle, eikä näiden rakentamiseen kulu paljoa aikaa. Seuraavaksi aloitan vaihtoehtoisen toteutuksen komponentilleni. Kuten aiemmista päivämerkinnöistä käy ilmi, komponentin käyttöperä on pääosin visuaalinen. Tarkoituksena oli eräiden usein toistuvien käyttöliittymäelementtien luomisen virtaviivaistaminen, ja alun perin rakentamallani komponentilla näitä luotiin dynaamisesti tarpeen mukaan, vaihtuvin parametrein. Tämä ei kuitenkaan ollut suorituskyvyn näkökulmasta optimaalista. Opiskelen uutta käyttötarkoitusta varten JavaScript-skriptien muodostamista loput päivästä.

Tiistai 25.2.2025

Päiväkohtainen tavoite: Skriptin kirjoittaminen valmiiksi työpäivän loppuun mennessä.

Olen tämän päivän etätöissä. Päivän agenda on selvä: tarkoitukseni on saada käyttövalmiiksi skripti, joka generoi tarvitsemamme visuaaliset elementit etukäteen annetuin säännöin jokaiselle eri parametrikombinaatiolle. Näiden esittämiseksi emme luo koko elementtiä sillä hetkellä kuin se tarvitaan, vaan kokoamme näistä kirjaston, joista valitaan oikea kuva sen mukaisesti, millaisia parametrejä milläkin näkymällä on sillä hetkellä, kun elementti näytetään. Näitä eri kombinaatioita on yli 150, ja mikäli tulevaisuudessa niitä tahdottaisiin päivittää, olisi sen käsin tekeminen vaivalloista. Tämän perusteella nämä ehdollisesti generoiva skripti on resurssitehokkain vaihtoehto. Työni on käytännössä ollut tähän asti hyvin front end-painotteista. Tämä skriptin muodostaminen muistuttaa kuitenkin logiikaltaan enemmän back end-koodaamista, ja pidän sen tuomasta vaihtelusta työhöni. Koska keskityn yhteen tarkkaa keskittymistä vaativaan asiaan koko päivän, tuntuu etätyö tähän tarkoitusperään optimaaliselta. Opiskelen internet-lähteistä tekniikoita

haluamani lopputuloksen aikaansaamiseksi, sekä testailen ja hion osaelementtien asettelulogiikkaa, kunnes olen tulokseen tyytyväinen. Päivä venyy hieman ylitöiksi, sillä olen vakaasti päättänyt saavani skriptini tehdyksi tänään. Saan skriptini valmiiksi noin kaksi tuntia tavanomaisen työajan päätyttyä.

Keskiviikko 26.2.2025

Päiväkohtainen tavoite: Skriptin sisällyttäminen ohjelmaan ja sen mukaiset muutokset sen tuotoksesta riippuvaisiin komponentteihin, sekä näiden välitys Sovellus A:n tuotantoversioon.

Visuaaliset elementit generoiva skriptini on valmis, mutta päätän vielä lisätä muutaman asian sen toiminnallisuuksiin. Elementtien aikaisempi toteutustapa on käyttänyt kuvia base64-muodossa, jossa tiedosto muutetaan merkkijonoksi. Kyseiseen käyttötarkoitukseen tässä ei ole varsinaisesti etua, eikä toisaalta haittaakaan. Kiinnostukseni tähän johtuu yksinkertaisesti siitä, että voin säästää aikaa ja vaivaa vain muokkaamalla aiempaa toteutustapaa hieman paremmaksi ja asettamalla sen käyttämään uusia generoimiani elementtejä. Tätä varten skriptin generoimat elementit tulee kuitenkin muuntaa base64-formaattiin. Teen skriptiini toisen funktion, jonka voi ajaa komentoriviltä itse elementtien generoimisen valmistuttua. Tämän perusteella niistä tehdään oma tsx (typescript) tiedosto, joka sisältää kaikkien elementtien base64-muunnokset omana avain-arvoparinaan, asetettuna objektin sisälle. Aikaisemmalla toteutustavalla jokainen base64-kuva tarvitsi oman importin valintalogiikan sisältävän komponentin sisään, parannuksieni ansioista riittää, että importaan vain base64-kuvat sisältävän objektin tälle komponentille. Saan täten asiakkaan pyytämän toiminnallisuuden vihdoin tuotantoon välitettäväksi, ilman vähäisintäkään suorituskykyheikennystä. Näiden visuaalisten elementtien jatkopäivitys esimerkiksi vaihtoehtoisen visuaalisen ilmeen saavuttamiseksi on myös merkittävästi helpompaa tulevaisuudessa.

Torstai 27.2.2025

Päiväkohtainen tavoite: Syventyminen Python-ohjelmointikieleen.

Tarkoitukseni on nyt toistaiseksi keskittyä erääseen omaan tuotteeseemme, johon viittaan tästä lähtien nimellä Sovellus C. Kyseinen sovellus on aiemmasta poiketen toteutettu Python-ohjelmointikielellä React Nativen sijasta. Kyseessä ei myöskään tällä kertaa ole mobiilisovellus, vaan se on tarkoitettu Microsoft Windows-käyttöjärjestelmällä käytettäväksi. Saamieni tehtävien onnistunut toteutus vaatii aluksi opiskelua. Python on minulle ohjelmointikielenä tuttu, mutta en ole käyttänyt sitä muutamaan vuoteen. Huomaan unohtaneeni jopa merkkijonojen käsittelymetodit. Pyrin kuitenkin aloittamaan toteutukseni rakentamisen melko suoraan, etsien tietoa sitä mukaa kuin ongelmia nousee esiin. Kyseisen ohjelman on tarkoitus ottaa yhteys ulkopuoliseen, tietokoneeseen liitettyyn laitteeseen, ja välittää sille komentoja. Koska tietokoneeseen on usein

liitetty useita eri laitteita, on tarkoituksenomaista ensiksi ryhtyä selvittämään yhteydenmuodostusta oikeaan laitteeseen. Rakennan tätä varten valitsijalogiikan, joka tutkii minkälaista dataa mikäkin yhteysportti palauttaa siihen otettaessa yhteyttä. Tämän perusteella oikeaksi tulkittu portti asetetaan ohjelman oletusyhteydeksi. Edistys on hieman hidasta aluksi, mutta nopeutuu sen mukaisesti mitä enemmän saan aikaiseksi.

Perjantai 28.2.2025

Päiväkohtainen tavoite: Python-ohjelmointikielen jatko-opiskelu, graafisen käyttöliittymän (GUI) toteutus Tkinter-kirjastolla.

Olen saanut edellisenä päivänä oikean yhteysportin automaattisen valinnan toteutettua. Tästä seuraavaksi tehtäväni on päivittää sovelluksen graafista ulkoasua käyttäjäystävällisemmäksi. Erotan eri toiminnallisia elementtejä, kuten valikoita ja nappeja, hieman toisistaan sovellusikkunassa, ja ryhmittelen ne käyttötarkoituksiensa mukaisesti. Lisään sovellukseen visuaalisia elementtejä, jotka havainnollistavat käyttäjälle, mitä sovellus milloinkin tekee. Tällöin käyttäjät eivät esimerkiksi virheellisesti tulkitse sovelluksen jumittuneen, kun toiminnallisuus on hetkellisesti estettynä jonkin taustaprosessin valmistuessa. Prosessi on opettavainen: en ole ennen tehnyt GUI:ta pythonilla, joten lähes puolet ajastani kuluu dokumentaation lukemiseen. Käytämme toteutukseen tkinter-kirjastoa, joka soveltuu hyvin visuaalisesti vaatimattoman ja yksinkertaisen GUI:n laatimiseen. En ole valinnut tätä kirjastoa käytettäväksi kuitenkaan itse. Työstämästäni ohjelmasta on ollut jo aiempi, toiminnaltaan melko rajoitettu versio olemassa, ja tarkoitukseni on laajentaa sen toimintaa. Jatkan siis aiemman toteutuksen päälle rakentamista. Minulla ei ole käytettävästä GUI-kirjastosta huomautettavaa, sillä se vaikuttaa olevan yleisesti käytössä. Täten sen dokumentaatio on kattavaa ja vertaiskokemuksia ongelmatilanteista löytyy internetistä hakemalla helposti. Kaiken kaikkiaan olen tyytyväinen viimeisten kahden päivän tehtäviini, sillä koen niiden monipuolistaneen osaamistani ammattiin liittyvissä työkaluissa ja teknologioissa.

Viikkoanalyysi 5

Viikolla tapahtunut ammatillinen kehitys on painottunut teknologioiden ja työkalujen käytön, sekä siihen liittyvän osaamisen, monipuolistamiseen. Skriptini kirjoitukseen pystyin parhaiten tarttumaan etätyön tuomassa eristyksessä muusta työyhteisöstä. Samoin Sovellus C:n parissa työskentely on tarkoittanut fyysistä etäisyyttä kollegoihini. Vaikka olen työstänyt Sovellus C:tä vain toimistolla, sen toimivuuden testaamiseksi olen joutunut siirtämään työpisteeni eri huoneeseen. Laitteisto, johon sovelluksella otetaan yhteys, on eräänlaisessa testipenkissä, joka sijaitsee toimistomme sivuhuoneessa, ja on täten työympäristönä melko eristäytynyt muusta toimistosta.

Alkuviikosta keskityin aiemmin tuottamani koodin refaktorointiin. Kiireessä tehty koodi on usein epäjohdonmukaista ja huonosti jäsenneltyä, kun prioriteettina on saada jokin toiminnallisuus toteutettua esimerkiksi toimitusaikataulussa pysymiseksi. Refaktoroinnin tarkoitus on jälkikäteen jäsennellä, järkeistää ja yhdenmukaistaa aiemmin toteutettua koodia (BMC 2024). Omalta osaltani tämä oli asianmukaista laatimani komponentin koodille: prioriteettina oli teknisen toteutuksen toiminta, ja tähän liittyi useiden eri menetelmien pikaista peräkkäistä kokeilua. Koodi oli tämän vuoksi paikoittain huonosti jäsenneltyä. Vaikkei komponenttini aiempi ratkaisu päätynytkään lopulta Sovellus A:n tuotantoversioon, jää sille erottamani haara githubiimme tulevaisuuden käyttötarkoituksien varalta. Tätä varten tavoitteeni oli, että koodi on myös tulevaisuudessa ymmärrettävää, oli lukijana sitten minä taikka joku kollegani. Samassa opin myös muun muassa parametrien järkevämpää välitystä komponenttien välillä React Nativea käytettäessä, joka tulee varmasti myös jatkossa tarpeeseen.

Opin myös paljon uutta skriptien hyödyntämisestä kehitystyössä. Laatimani skriptin tarkoitus on virtaviivaistaa muuten työlästä prosessia, joka liittyy Sovellus A:n käyttämien visuaalisten elementtien luontiin ja muokkaukseen, toisin sanoen manuaaliseen kuvankäsittelyyn. Toteutin skriptin Sovellus A:n kehitysversioon, ja se toimii täysin kehittäjän työkaluna. Loppukäyttäjillä ei ole tähän pääsyä lainkaan. React Nativen sijasta tämä on rakennettu puhtaalla JavaScriptillä, ja se ajetaan erillisillä komennoilla komentoriviltä, sovelluksen muita komponentteja täydentäen ja apuna käyttäen. Huomattuani tämän käytännöllisyyden sekä ajansäästölliset hyödyt, laajensin skriptin toteuttamaan myös muita käyttötarkoituksia kuin olin osannut etukäteen tavoitteeksi asettaa, ja tiedän sen olevan hyödyllinen työkalu myös kollegoilleni jatkossa. Pystyn tämän ansiosta tulevaisuudessa paremmin tunnistamaan tehtäviä, joissa voidaan saavuttaa resurssihyötyjä skriptin avulla.

Loppuviikosta keskityin vielä Python-ohjelmointikieleen sekä Windows-sovelluksen GUI:n luomiseen. En ollut käyttänyt Pythonia pitkään aikaan, joten jouduin aluksi keskittymään perusasioiden kertaamiseen. Muistin lähinnä Pythonin omintakeiset demarkaatiosäännöt, mutta muuten minun täytyi aluksi hakea verkosta ohjeita melkein kaikkeen mitä pyrin koodilla toteuttamaan. Sovellus C, jonka jatkokehittämiseen tehtäväni keskittyivät, oli melko primitiivisessä vaiheessa. Kollegani olivat luoneet yksinkertaisen käyttöliittymän ja toiminnallisuuden sille noin vuosi sitten, eikä siihen oltu sen jälkeen juuri koskettu. Vaikka koodi oli sinänsä selkeätä, sitä ei ollut juurikaan kommentoitu. Kielen ja etenkin ohjelman käyttämien koodikirjastojen ollessa itselleni tuntemattomia entuudestaan, oli sen eri osuuksien ymmärtäminen minulle alkuun hyvin haastavaa. Tähän auttoi dokumentaation lukeminen sekä pienten muutosten tekeminen koodiin, jolloin sen tulosta tarkastelemalla pystyin pikkuhiljaa hahmottamaan mitä mikäkin sen osa käytännössä teki. Monimutkaisen ja vaikeasti ymmärrettävän ongelman ollessa edessä, on sen pilkkominen pienempiin osa-alueisiin hyödyllistä (Emelianov 2023). Tätä tarkoitusperää varten toteutin myös

itse omaa menetelmääni ohjelman koodin ymmärtämiseksi. Ymmärrykseni koodin toiminnasta kasvoi lumipalloefektin kaltaisesti, ja pian pystyin jo rakentamaan omaa toteutustani aiemman koodin päälle. Tulin huomanneeksi saman kuin jo aiemmilla seurantaviikoilla: aluksi vaikeaselkoiset asiat ratkeavat kyllä, kun niitä selvittää määrätietoisesti.

Kulunut viikko on merkittävästi monipuolistanut osaamistani eri ohjelmointityökalujen ja -teknologioiden käytössä. Pystyn jatkossa tämän ansiosta tarttumaan entistä laajempaan joukkoon työtehtäviä, ja luottamaan paremmin omaan ammattitaitooni. Monipuolisempi kyvykkyys ohjelmoinnissa myös lyö varmuutta uralleni eri teknologioiden tullessa käyttöön sekä aiempien käytöstä poistuessa (SkillReactor 2023), kun koko osaaminen ei ole kiinni vain yhdessä teknologiassa.

3.6 Seurantaviikko 6

Viikon tavoitteet: Teknologian ja työkalujen käyttö, yhteistyö ohjelmistokehitystiimissä

Maanantai 3.3.2025

Päiväkohtainen tavoite: Sovellus C:n testauskäyttöliittymän toteutus, laitteidenvälinen kommunikaatio sarjaportteja käyttäen ja tätä tukevat ohjelmistot.

Uuden viikon työtehtävät jatkuvat Sovellus C:n parissa. Olen saanut edellisen viikon tehtävät sen parissa pitkälti suoritettua, ja tälle viikolle saan Kollega A:lta uusia. Tarkoituksenani on nyt rakentaa testauskäyttöliittymä sovellukseen. Tämän tehtävän tavoitteena on ottaa yhteys ulkoiseen laitteeseen ja antaa sille testikomentoja. Näiden aikaansaaman palautteen perusteella on mahdollista havainnoida, että laite toimii oikein. Tälle halutaan oma osionsa sovelluksen käyttöliittymässä, joka avautuu, kun käyttäjä kytkee testaustilan päälle. Edellisellä viikolla kerryttämäni kokemuksen perusteella saan rakennettua rajoitetun käyttöliittymän testaukselle nopeasti. Yllättäen kuitenkaan en saa testikomennoilla mitään aikaan. Yritän etsiä laitteen rajapintadokumentaatiosta syytä tälle, tuloksetta. Lopulta kysyn kollegaltani, jos hänellä olisi ideoita ongelman ratkaisemiseksi. Selviää, että näitä testikomentoja tulee ajaa eri sarjayhteyden kautta, kuin viime viikolla rakentamalleni toiminnallisuudelle. Pieni informaatiokatkos on tuhlannut muutaman tunnin työaikaani, mutta en lannistu tästä. Lataan tietokoneelleni PuTTY-ohjelman, joka on avoimen lähdekoodin pääte-emulaattori. Pystyn tällä ohjelmalla ottamaan yhteyden suoraan laitteisiin ilman tarvetta muovata Sovellus C:n ohjelmakoodia. Tämä sopii käyttötarkoitukseeni, sillä tahdon aluksi ainoastaan löytää sarjaportin, joka kykenee kommunikoimaan laitteen kanssa. Tunnustelen tietokoneestani löytyviä avoimia sarjaportteja, kunnes saan ajamalleni komennolle vastauksen. Oikean yhteysportin selvittyä saan ajettua testikomennot laitteelle onnistuneesti.

Päivän toimintaa olisi voinut kenties tehostaa, jos olisin tajunnut kysyä yhteysportista kollegaltani jo aiemmin.

Tiistai 4.3.2025

Päiväkohtainen tavoite: Aktiivinen osallistuminen asiakaspalaveriin, Sovellus A:n uuden sprintin aloitus sekä tähän liittyvät tehtävät

Päivä alkaa palaverilla Sovellus A:han liittyen. On jälleen aika aloittaa uusi sprint sen kehittämisessä, ja keskustelemme yhdessä asiakkaan kanssa sprintin tavoitteista. Osallistun kokoukseen kahden kollegani, Kollega A ja Kollega B, kanssa. Annamme asiakkaan nostamille vaatimuksille arviomme niiden toteuttamisen vaativuudesta sekä ajasta, ja keskustelemme lähestymistavoista haluttujen muutosten toteuttamisessa. Ryhdyn toteuttamaan Sovellus A:n uuden sprintin tehtäviä heti palaverimme jälkeen. Huomaan pian jotain huolestuttavaa: kun projektin lähdekoodin kopioi tuoreeltaan GitHubista, ei sovellus suostu aukeamaan emulaattorilla, kun sen vaatimat kirjastot ovat asennettu. Sovelluksen ruutujen välistä siirtymistä hallinnoiva kirjasto ilmoittaa, ettei sitä ole alustettu oikein ohjelmakoodissa. Tarkistan tämän sovelluksen koodissa, ja kaikki näyttää olevan tehty virallisen dokumentaation mukaisesti. Myös kollegani ryhtyvät ongelman ratkaisemiseen. Saamme selville, että jostain syystä kyseisestä kirjastosta asentuu kaksi eri versiota samanaikaisesti. Emme kuitenkaan heti keksi, kuinka korjata ongelma. Jotta pääsemme varsinaisen työn pariin sovelluksen kehittämisessä, asetan manuaalisen eston kirjaston väärän version asentumiselle Yarn-pakettihallintaohjelman resolutions-ominaisuudella. Tällä ominaisuudella voidaan määrittää kirjastoille tietty versio, jota paketinhallinta ei pysty kiertämään. Tyhjennettyäni kirjastot projektikansiosta ja asennettuani ne uusiksi, toimii ohjelma taas. Ehdin tämän jälkeen vielä toteuttaa muutaman pienen tehtävän sovellukseen liittyen, ennen kuin päivä on päätöksessään.

Keskiviikko 5.3.2025

Päiväkohtainen tavoite: React Native komponentin refaktorointi tyypistä toiseen

Jatkan Sovellus A:n kehityksen parissa. Tehtävänäni on muuttaa eräs sovelluksen sivu erillisestä screenistä komponentiksi. Aiemmin tämä screen on ollut puolittain tyhjä, jolloin sen alla ollut screen on jäänyt puoliksi näkyviin sen alle. Tämä on ollut aiemmin haluttu ominaisuus, muttei yhteensopiva asiakkaan haluaman uuden toiminnallisuuden kanssa. Tavoitteena on, että sekä alla olevan screenin sekä päällä olevan screenin toiminnallisuudet ovat käytössä samanaikaisesti. Tästä syystä alan muovaamaan tätä päällä ollutta screeniä BottomSheet-komponentiksi alemman screenin sisään. BottomSheet on komponentti, joka aukeaa joko kokonaan tai osittain aktiivisen näkymän päälle, nimensä mukaisesti ruudun alareunasta laajentuen. Itse visuaalisten elementtien

siirto on melko mutkatonta. Suuri osa toiminnallisuudesta on kuitenkin riippuvaista siitä, että komponentti on oma screen sovelluksen navigaatiopinossa. Sen sijaan että voisimme käyttää näitä toimintoja suoraan tässä komponentissa, joudun siirtämään ne alemman screenin koodiin, josta näitä kutsutaan ns. CallBackilla komponentin sisältä. Tässä huomaan hyödyn aiempien viikkojen ongelmien kanssa painimisesta: parametrien ja kutsujen välittäminen komponentilta toiselle ovat minulla nyt hyvin muistissa, ja pystyn suoraan rakentamaan toiminnallisuutta näiden päälle ilman tarvetta dokumentaation lukemiselle.

Torstai 6.3.2025

Päiväkohtainen tavoite: Ehtolausekkeiden järkeistys ja tarkastuslogiikan optimointi, React Native optimointimenetelmät

Eilen rakentamani muutokset näyttävät toimivan hyvin, joten siirryn Sovellus A:n seuraavaan tehtävään. Eräs käyttäjälle näytettävä tiedote ei vaikuta toimivan niin kuin sen pitäisi. Tiedotteen tulisi aina aueta tietyn ehdon täyttyessä, mutta jostain syystä se aukeaa vain toisinaan, ja välillä ilman ehdon täyttymistä. Tarkastelen koodia, ja koitan etsiä virhettä sen logiikassa. Huomaan, että ehdon täyttymistä seurataan hieman epäloogisella tavalla: ehto ei liity suoraan tilaan, jossa haluamme tiedotteen näkyvän, ja arvelen sen kenties olleen parempi tapa jossain aikaisemmassa sovelluksen versiossa. Sovellus A:n nykymuodon kanssa tämä ei kuitenkaan selvästi enää toimi, joten rakennan uuden toiminnallisuuden tämän ehdon täyttymisen seuraamiseksi. Aiemmin tätä seurattiin monimutkaisen laskennan kautta, ja jonkin osamuuttujan tarkoituksen muututtua, laskutoimitus ei enää saanut tarkoitettua arvoa kaikissa tilanteissa. Teen tästä yksinkertaisemman tarkistuksen: backendista haettavan tiedon ollessa olemassa asetamme boolean-muuttujan arvoksi ”true”, ja sen puuttuessa kokonaan asetamme arvoksi ”false”. Näytämme tiedotteen käyttäjälle, mikäli tämä arvo on “true”. Tehtävä on tällä ratkaistu, joten jatkan tästä seuraavaan. Olen edellistä tehtävää testaillessani huomannut ongelman edellisen tehtäväni toteutuksessa. BottomSheetin visuaaliset elementit alkava välkkymään sen oltua auki noin 10 sekuntia. React-komponentit uudelleenrenderöityvät oletusarvoisesti aina, kun niiden käyttämät parametrit muuttuvat. Oletan tämän olevan syy tähän käytökseen: ohjelma tietää ainoastaan, että muuttujien arvot ovat asetettu uudestaan, muttei tee vertausta siihen, ovatko itse arvot muuttuneet. Turhan renderöinnin välttämiseksi käytän Reactin useCallback metodia. UseCallbackin sisälle sijoitettu komponentti renderöidään uusiksi ainoastaan silloin, kun jokin sen käyttämä parametri tosiasiallisesti muuttuu. Saan tällä turhan välkkymisen estettyä.

Perjantai 7.3.2025

Päiväkohtainen tavoite: React Native JSX-elementtien opiskelu ja käyttötarkoituspohjainen valikointi, käyttäjäkokemuksen hiominen

Olemme huomanneet kehityskohteen Sovellus A:n toiminnassa: joissain näkymissä puhelimen näppäimistö saattaa avautua syötekentän päälle, jolloin käyttäjä ei näe kirjoittamaansa tekstiä. Koodin perusteella käyttäjän pitäisi kyllä saada vähintään vieritettyä ruutunäkymää syötekentän paljastaakseen, mutta tämä toimii vaihtelevasti. Saan tehtäväksi käydä sovellusta läpi ja etsiä kohtia, jossa kyseinen toiminnallisuus on rikki. Kerään ongelmallisista näkymistä listan, ja tämän jälkeen suunnittelen ratkaisun toteuttamista. React Nativen visuaalisten elementtien asetteluun käytetään JSX:ää, joka on syntaksilaajennus JavaScriptille. JSX muistuttaa hyvin pitkälti HTML:ää, käyttäen kuitenkin omia elementtejään. Ongelmat näyttävät johtuvan pitkälti sovelluksen monimutkaisesta JSX koodista. Vaikuttaa siltä, että visuaalisten elementtien koodia on jatkettu ja korjailtu sieltä täältä tarpeen mukaan, monen eri tekijän toimesta, ja lopputulos on sekalainen viidakko eri JSX elementtejä. Löydän internetistä ohjeet KeyboardAvoidingView-elementin käyttöön. Tämä vaikuttaisi sopivan käyttötarkoitukseemme, ja lähden toteuttamaan sen käyttöönottoa komponenteilla, joilla havaitsin näppäimistön asetteluun liittyviä ongelmia. Saan tällä toteutettua tarvitsemamme korjaukset.

Viikkoanalyysi 6

Olen huomannut omassa tekemisessäni ammatillisen itsetunnon kasvun. Pystyn aiempaa määrätietoisemmin tarttumaan annettuihin tehtäviin, kysymättä kollegoilta neuvoa esimerkiksi toteutustapaan liittyen. Tämä korostuu etenkin Sovellus A:ta työstäessä. Olen käyttänyt suurimman osan työajastani juuri Sovellus A:n parissa, joten sen toiminta alkaa olla minulle varsin tuttua. Sovellus koostuu kymmenistä eri komponenteista ja näyttösivuista, enkä aluksi osannut hahmottaa näiden keskinäisiä riippuvuuksia. Arkailin myös asioiden muuttamista. Vaikka jokin koodin osa saattoi vaikuttaa itselleni oudosti toteutetulta, arvelin sille olevan jokin syy. Ollessani nyt paremmin perillä sovelluksen toiminnasta, pystyn kuitenkin jo melko itsenäisesti arvioimaan, milloin minun on aiheellista muuttaa sen toimintaa. Toisinaan kollegoillani on kysyttävää ratkaisuistani, ja tällöin joudun perustelemaan valitsemiani toteutustapoja. Näistä yhdessä keskusteleminen tuntuu olevan oppimisen kannalta erityisen hyödyllistä. Jos en pysty perustelemaan ratkaisuani vakuuttavasti, saan usein arvokasta palautetta ja ehdotuksia paremmista toteutustavoista. Palaute työn jäljestä auttaa ammatillisessa kehityksessä sekä työn laadun takaamisessa (Heath 2024). Joskus myös huomaan erheet omassa suunnittelussani jo yksinkertaisesti siinä, että selitän päätöksiäni sanallisesti auki. Näin pystyn kriittisesti tarkastelemaan omaa toimintaani, ja tunnistamaan kehitettäviä kohteita. Vaihtoehtoisesti ratkaisujeni ollessa vakaasti perustellulla pohjalla koen tyytyväisyyttä omaan työhöni, ja uskallan toteuttaa omia menetelmiäni entistä määrätietoisemmin.

Toisinaan itsekin joudun laatimaan epäoptimaalista koodia, sillä ns. parhaiden käytäntöjen mukaisesti tehty koodi ei yksinkertaisesti toimi tietyssä käyttötilanteessa. Tämä on tyypillisintä silloin, kun kirjoitettavalla koodilla on keskinäinen riippuvuus toiseen osaan koodista. Aina ei ole

mahdollista refaktoroida koodia järkevämmäksi, vaan oma ratkaisu tulee asettaa toimimaan aiemman toteutuksen mukaisesti. Usein näissä tilanteissa tulee mietittyä, että refaktoroinnin voi tehdä sitten kun on paremmin aikaa, mutta harvoin tätä aikaa tuntuu löytyvän. Mahdollisuuksien mukaan olisi kuitenkin hyödyllistä refaktoroida koodia esimerkiksi uusia toiminnallisuuksia lisätessä (Altexsoft 2018), ja toteuttaa uuden toiminnallisuuden koodi tämän mukaisesti. Muuten refaktoroitavan koodin määrä kasvaa, ja refaktoroinnin toteuttaminen on tulevaisuudessa vaivalloisempaa.

Aiemman kokemukseni vähyys näkyi hieman siinä, kuinka ratkaisin väärästä kirjastoversiosta johtuneen ongelman Sovellus A:ssa. En osannut heti epäillä, että toinen kirjasto voisi erikseen ilmoittamatta asentaa eri versioisen kaksoiskappaleen jo käyttämästämme kirjastosta. Tämä johtuu lähinnä siitä, että itse tekemäni projektit ovat olleet pieniluonteisia, eivätkä rakenteeltaan kovinkaan monimutkaisia. Harvoin projektini ovat edes olleet tarpeeksi pitkäikäisiä, jotta niiden käyttämien kirjastojen päivitys olisi ollut aiheellista. Jatkossa pystyn kenties nopeammin debuggaamaan tämän kaltaisia ongelmatilanteita.

Kulunut viikko on myös monipuolistanut lähestymistäni ohjelmointiin. Aina ei ole aiheellista keskittyä vain toiminnallisuuksien tekniseen toteutukseen. Sovelluksen toiminnassa olennaista on myös käyttäjäkokemus, ja tästä näkökulmasta työstin etenkin perjantain näppäimistöongelmaa. Hyvin suunniteltu käyttöliittymä parantaa sovelluksen käytön sujuvuutta, ja täten asiakastyytyväisyyttä (Chhetri 2023). Tarkoitettu toiminnallisuus oli jo tehty ja teknisesti toimiva, muttei kovin käyttäjäystävällinen. Hioin ratkaisuani ja sen tyylittelyä, kunnes sain sen tasolle, jolla en itsekään käyttäjänä saisi mieleeni huomautettavaa sen toiminnasta.

Osallistuin kuluneella viikolla kolmatta kertaa asiakkaan kanssa pidettävään sprinttipalaveriin Sovellus A:n kehitystyöhön liittyen. Ensimmäisellä kerralla olin mukana lähinnä kuuntelemassa, mutta kahdessa viimeisimmässä palaverissa olen ollut myös mukana antamassa mielipiteitäni ja ideoita ratkaisujen suunnittelulle sekä toteuttamiselle. Nämä kokemukset ovat parantaneet ymmärrystäni yhteistyöstä ohjelmistokehitystiimissä. Näkökanta tähän liittyen ei ole rajoittunut ainoastaan oman tiimimme väliseen vuorovaikutukseen, ja keskustelut sovelluksen kehitystyötä tilaavan asiakaan kanssa ovat parantaneet ymmärrystäni kehitykseen liittyvistä liiketoiminnallisista prosesseista. On sekä meidän että asiakkaan intresseissä, että työn laadulliset sekä ajalliset tavoitteet täyttyvät. Laadukas ja aikamääreissä pysyvä työtulos auttaa yhteistyösuhteemme jatkumisessa (Franklin 2025), sekä mahdollistaa työaikamme käyttämisen myös muihin projekteihimme, kuin toimitussopimuksen mukaisen työn toteuttamiseen Sovellus A:lle.

Kaiken kaikkiaan viikko on ollut tuloksellinen ammatillisen kehittymiseni kannalta. Kehitystä on tapahtunut sekä teknologian ja työkalujen käytön suhteen, että yhteistyöllisten tavoitteiden osalta.

3.7 Seurantaviikko 7

Viikkotavoitteet: Työkalujen ja teknologioiden hyödyntäminen käyttäjäkokemuksen optimoinnissa, modulaarisuus ja koodin uudelleenkäytettävyys sekä ylläpidettävyys

Maanantai 10.3.2025

Päiväkohtainen tavoite: React Native komponentin luominen näkymien yhtenäistämiseksi

Kerron viikoittaisessa palaverissamme perjantaina toteuttamastani ratkaisusta näytön asetteluongelmien korjaamiseksi. Kokeneempi kollegani pyytää minua jatkamaan toteutustani modulaariseksi käyttöliittymäkomponentiksi. Tavoitteena on luoda visuaalinen komponentti, jolla voimme yhtenäistää Sovellus A:n eri näyttöjä. Nykyisellään sovelluksen eri näyttöjen JSX-koodi on tapauskohtaisesti luotua. Niiden sisältämien komponenttien tyylittely on toteutettu osin Ennalta määrätyillä ja standardoiduilla tyylitiedostoilla, ja osin tapauskohtaisin määrityksin. Nykyinen toteutustapa on hankalasti ylläpidettävä, ja sovelluksen visuaalisen ilmeen päivittämiseksi tarvitaan muutoksia lukuisiin eri komponentteihin. Uudella komponentilla määritämme standardoidut, modulaariset raamit käytettäväksi useilla eri näytöillä, ja ainoastaan sisemmät elementit luodaan tapauskohtaisesti vastaten näytön käyttötarkoitusta. Header- ja Footer-osiot sen sijaan ovat kiinteitä osia tätä uutta komponenttia, ja niiden toiminta määritetään komponentille välitettävillä propseilla. Saan tästä maanantain aikana luoduksi alkeellisen version.

Tiistai 11.3.2025

Päiväkohtainen tavoite: Komponentin modularisointi, yhteistyöllisen toiminnan edistäminen

Käyn eri näyttöjämme läpi, ja selvitän, mitä toiminnallisuuksia minkäkin näytön Header- ja Footer-osiossa on toteutettuna. Koska nämä osiot ovat kiinteästi uudessa komponentissa, minun täytyy luoda niiden tarvitsemat propsit komponentin TypeScript-määritelmään. Tätä jatkaen asetan ehdollista muotoilua komponentille, jota määrittää komponentin saamat propsit. Ehdollista muotoilua tarvitsee esimerkiksi Modalit – nämä ovat toisen näytön sisällä aukeavia koko ruudun näkymiä, jotka eivät kuitenkaan ole omia näyttöjään. Sovelluksen isäntänäkymä on rootissa määritetty SafeAreaView:n sisään, mutta Modalit kiertävät tämän. SafeAreaView:n on tarkoitus ottaa huomioon etenkin iOS laitteiden yläpalkit taikka etukamerat, jotka usein ovat upotettu keskelle näytön yläreunaa. Tästä syystä tarvitsemme Modaleille lisätäytettä yläreunaan. Asetan komponentin tyylittelyn paddingiin suuremman arvon, kun sille välitetään prop isModal arvolla true. Komponentin käytön helpottamiseksi aloitan myös JSDoc-ohjeen työstön. JSDocilla voin luoda komponentilleni ohjetekstin, jonka ohjelmoija näkee pitäessään hiirtä komponentin JSX-elementin päällä koodieditorissa. Tämä auttaa jatkossa sekä kollegoitani että itseäni komponentin käytössä.

Keskiviikko 12.3.2025

Päiväkohtainen tavoite: JSDoc opiskelu, käyttäjäkokemuksen parantaminen teknologisin ratkaisuin, Python-ohjelmointikielen syvempi tuntemus

Olen saanut näyttökomponenttini valmiiksi, ja asettanut sen käyttöön kaikille näkymille, joille näen sen aiheelliseksi. Seuraavaksi tavoitteena on parantaa sen käytön helpottamiseksi tekemääni JSDocia. Laajan määrän eri näyttöjä ja niiden vaatimien toiminnallisuuksien vuoksi mahdollisten propsien määrä on kasvanut melko suureksi. JSDocissa listaan eri mahdolliset propit, niiden käyttötarkoituksen, muodon missä mikäkin prop tulee välittää sekä tiedon onko kyseessä pakollinen prop. Näin nämä tiedot ovat helposti saatavilla ohjelmoidessa. Tämän jälkeen jatkan Sovellus C:n työstämistä. Jotkin sovelluksen toiminnallisuudet vaativat eri tietoyhteysportin laitteeseen, kuin toiset. Tätä varten luon valikon, josta käyttäjä voi valita milloinkin tarvitsemansa yhteysportin. Huomaan toisinaan turhautuvani Pythonilla ohjelmointiin. Tiedoston pituuden kasvaessa Pythonin tapa erotella koodin eri loogiset osuudet toisistaan alkaa olla hieman vaikealukuista. En ole kuitenkaan Pythonissa tarpeeksi kokenut, että uskaltaisin jakaa koodia eri tiedostoihin. Eri osuudet koodissa tarvitsevat pääsyn samoihin muuttujiin, enkä ole varma, kuinka näitä välitetään luotettavasti tiedostojen välillä Pythonilla. Tunnistan tässä kuitenkin kehityskohteen omaan osaamiseeni.

Torstai 13.3.2025

Päiväkohtainen tavoite: yhteistyöllinen toiminta ohjelmistokehityksessä, käyttäjäkokemuksen parantaminen teknologisin keinoin, Python-kirjastojen hyödyntäminen

Työstän päivän aluksi Sovellus A:ta. Asiakas on pyytänyt meitä toteuttamaan ratkaisun, jolla loppukäyttäjille välitetään sovelluksen sisältämistä tuotteista lisätieto. Tämän mahdollistamiseksi minun täytyy muokata tapaa, jolla haemme tuotteen tiedot backendistä. Kollegani tekee muutokset backend-puolelle, ja itse toteutan näiden tietojen käsittelyn frontendissä. Tähän sisältyy halutun tiedon lisäys tuotteen TypeScript-määritelmään, sekä tiedon näyttäminen tuotekomponentissa. Saan kollegaltani esimerkkilistan backendistä välitettävälle tiedolle toteutukseni rakentamista helpottamaan. Saamme yhteistyöllä asiakkaan pyytämän ratkaisun toteutettua noin tunnissa. Jatkan tämän jälkeen Sovellus C:n työstöä. Olen opiskellut Pythonin Threading kirjaston käyttöä, ja päätän käyttää sitä hyödykseni sovelluksen toiminnassa. Threading kirjastolla voimme jakaa sovelluksessa käytettäviä toimintoja eri säikeisiin prosessoitavaksi. Näin voimme ajaa eri toiminnallisuuksia yhtäaikaisesti, ilman resurssiristiriitoja. Aiemmin joidenkin toiminnallisuuksien ajo jäädytti GUI:n visuaaliset elementit siksi aikaa, kunnes funktion koodi oli käsitelty. Näin voin esimerkiksi näyttää visualisaatiota toiminnon etenemisestä GUI:ssa sillä aikaa, kun sitä suoritetaan. Käyttäjäystävällisyyden lisäämiseksi luon myös skannausalgoritmin, jonka ajamalla

sovellus asettaa automaattisesti tahdotun toiminnallisuuden vaatiman yhteysportin käyttöön. Tämän kanssa ilmenee kuitenkin ongelmia – jotkin yhteysportit pysyvät ohjelman toiselle toiminnallisuudelle varattuina, jos niitä on jo käytetty aiemmin ajon aikana.

Perjantai 14.3.2025

Päiväkohtainen tavoite: Komponentin käyttöasteen laajennus, koodin toteutustavan priorisointi ongelmatilanteessa

Käyn läpi Sovellus A:n eri näkymiä löytääkseni kohtia, jotka voisivat vielä hyötyä uuden komponenttini käyttöönotosta. Eri näyttöjen suuren määrän johdosta en ole muistanut ottaa näitä kaikkia huomioon aiemmin. Laajennan komponentin käyttöä eri näytöissä löydöksieni mukaisesti. Jatkan tämän jälkeen Sovellus C:n rakentamista. Yhteysporttien varattuna olemiseen liittyvä ongelma osoittautuu haastavaksi, enkä löydä tämän korjaamiseksi dokumentaatioista ohjetta. Ongelman ratkaisemiseksi on kuitenkin kiertotie. Olen huomannut, että portit vapautuvat yhden epäonnistuneen yhteydenoton jälkeen. Asetan porttien skannaukseen tarkoitetun funktioni kutsumaan jokaista porttia ylimääräisen kerran – näin seuraava yhteydenotto onnistuu aina. En kuitenkaan ole kovinkaan ylpeä tästä ratkaisusta, ja toivon keksiväni tälle oikeaoppisemman tavan myöhemmin.

Viikkoanalyysi 7

Viikolla on edistytty työkaluja ja teknologiaa sekä yhteistyötä koskevien tavoitteiden osalta. Työkalujen ja teknologian osalta tämä on sisältänyt eri kirjastojen opiskelua sekä ylläpidettävyyden ja käyttäjäkokemuksen parantelua. Yhteistyön osalta osaamista on edistetty työskentelemällä saman tehtävän parissa vastuunjaoin, ja helpottamalla kollegoiden työntekoa dokumentaatiota laatimalla.

Sovellus A:n osalta kehitystyö on keskittynyt uuden komponentin luomiseen. Komponentti on niin sanottu wrapper, jonka tarkoitus on yhtenäistää sovelluksen visuaalista ulkoasua ja käytettävyyttä, sekä helpottaa ulkoasun ylläpitämistä ja jatkokehittämistä. Komponentin laatiminen oli monimutkaisempaa, kuin osasin aluksi odottaa. Sen vaatiman korkean modulaarisuuden johdosta oli tarpeen luoda käsittelylogiikka suurelle määrälle propseja, eli muista näkymistä komponentille välitettäviä parametrejä. Propsien määrä kasvoi sitä mukaa, kun tunnistin niille tarvetta eri näyttönäkymiä läpikäydessä. Näillä propseilla säädetään esimerkiksi footer-osion sisältöä, näkyvyyttä ja varjostusta, sekä header- ja footer-osioiden sisältämien painikkeiden kutsumia funktioita. Propsien suuri määrä mahdollistaa komponentin käytön useassa eri näkymässä, tarpeettomien toiminallisuuksien ja elementtien ollessa kytkettävissä pois päältä tarpeen mukaisesti. Tämä tuo ylläpidettävyyteen helpotusta, kun visuaalista ulkoasua muutettaessa riittää

usein ainoastaan wrapper-komponentin määritystiedoston muokkaaminen. Näin muutokset voidaan ajaa kymmeniin eri näyttöihin yhden tiedoston säädöllä. Wrapper-komponentin käytön etuja ovatkin mm. abstraktio (monimutkaisten osioiden ”siivoaminen” – kehittäjän tarvitsee vain huolehtia oikeiden propsien välittämisestä), uudelleenkäytettävyys, ylläpidettävyys sekä virheensieto (Bhimani 2024). Käyttöliittymän yhtenäistäminen myös helpottaa sovelluksen käyttöä loppukäyttäjille, kun toiminnalliset elementit sekä sivujen asettelut ovat samankaltaisesti toteutettu näkymästä riippumatta.

Laadin myös dokumentaatiota luomani wrapper-komponentin käytön tukemiseksi. Hyvin laadittu dokumentaatio on yhteistyöllisen sovelluskehityksen kulmakivi. Dokumentaatio selkeyttää monimutkaisten rakenteiden toimintaa, vähentäen väärinymmärryksiä usean henkilön työstäessä samaa koodiperustaa (Iankovych 2023). Aiemmin olen tehnyt tätä kommentoimalla koodiani, mutta komponenttini laajan käyttöasteen koodissamme johdosta tahdoin tehdä dokumentaatiosta kollegoilleni mahdollisimman helposti löydettävää ja luettavaa. JSDoc on pääasiassa Java-/TypeScriptin dokumentointikieli, jolla voidaan luoda helposti saavutettavaa ohjeistusta eri toiminnallisuuksien ja funktioiden käyttöön. Useimmilla koodieditoreilla on JSDocille luontainen tuki, jolloin dokumentaation näkemiseen riittää esimerkiksi hiiren vienti sitä koskevan komponentin tai funktion päälle koodissa. Koodieditori näyttää tällöin JSDocilla määritellyn dokumentaation erillisessä pienessä ikkunassa, joka avautuu tarkasteltavan komponentin/funktion nimen päälle koodieditorissa (Ramatis 2024). Opiskelin JSDoc-dokumentaation asettelua ja tyylittelyä, ja parantelin tämän mukaisesti toteuttamaani dokumentaatiota iteratiivisesti helppolukuisuuden ja selkeyden varmistamiseksi. En ollut aiemmin kommentoinut koodiani JSDocilla, ja sillä luodun dokumentaation helppo saavutettavuus teki minuun vaikutuksen. Aion jatkossa hyödyntää JSDocia laajemmin koodissani, jotta toteuttamieni komponenttien käyttö ja ylläpitäminen on koko tiimimme kesken mahdollisimman helppoa.

Uutta on opittu myös Pythonin käytössä. Threading-kirjaston opiskelu ja käytännön toteutuksella hyödyntäminen on parantanut Sovellus C:n käytettävyyttä. Eri toimintojen ajamisen ollessa riippumatonta GUI:n tulostamiseen tarkoitetusta säikeestä, olen pystynyt tarjoamaan käyttäjälle parempaa visualisaatiota ja tiedon välitystä sen suorittamista toiminnoista. Esimerkiksi yhteysportteja skannatessa GUI:sta on nähtävissä, mikä portti on milloinkin tunnustelun kohteena. Tätä viestintää tehostetaan indikatiivisin värein, mustan ollessa neutraalin tekstin väri, vihreän ollessa tekstin väri onnistuneelle tulokselle, ja punaisen virhetilanteessa. Käyttöä ei kuitenkaan ole helpotettu ainoastaan visuaalisin menetelmin. Olen myös luonut funktion, jolla sovellus tulkitsee yhteysportista saamansa palautteen perusteella, mihin toimintoon kukin portti soveltuu. Tämä tehdään tarkastelemalla sarjayhteyden välityksellä ajetuille komennoille saatavaa palautetta. Tässäkin on ollut minulle paljon uutta opiskeltavaa, ja Sovellus C:n työstäminen on jälleen osoittautunut hyödylliseksi teknologisen osaamiseni monipuolistamisessa.

Yhteenvetona viikko on ollut onnistunut ammatillisen kasvun näkökulmasta. Jatkuva oppiminen on keskeinen vaatimus sovelluskehittäjänä menestymiseen (Gerges 2023), ja omassa työssäni tätä tarvetta sanelee käytännön tuomat haasteet. Jo aiempina viikkoina olen tunnistanut, kuinka kertyvä oppiminen mahdollistaa entistä nopeamman uuden tiedon omaksumisen jatkossa. Toisinaan uusien asioiden opiskelu tuntuu kuitenkin korostavan ennemmin sitä, kuinka paljon opittavaa vielä löytyy.

3.8 Seurantaviikko 8

Viikkotavoitteet: yhteistyöllinen ongelmanratkaisu, bugien selvitys ja korjaus, teknisten ratkaisujen punnitseminen sekä kollegoiden että asiakkaan kesken

Maanantai 17.3.2025

Päiväkohtainen tavoite: Bugien selvitys, toistaminen ja ratkaisun ideointi.

Työviikko alkaa jälleen viikoittaisella aamupalaverillamme, jossa käymme edellisellä viikolla suoritetut tehtävämme läpi ja ennakoimme tulevan viikon työtaakkaa. Tämän jälkeen käyn töihin Sovellus A:n parissa. Sovellus A:n kehityssprintti on päättymässä huomenna, ja tavoitteemme on saada sprintin tehtävät viimeisteltyä, jotta saamme sovelluksen uuden version tuotantoon. Listamme viimeiset tehtävät ovat kaksi bugia. Nämä ovat ilmeisesti vaivanneet sovellusta jo pitkään, mutta niiden hankalan toistettavuuden vuoksi näitä on ollut haastavaa selvittää. Käyttäjät eivät ole voineet seikkaperäisesti eritellä toimintaansa sovelluksessa bugien ilmaannuttua, joten ainoa vaihtoehtomme on summittaisesti yrittää käydä läpi erilaisia tilanteita, joilla bugit voisivat ilmentyä. Pitkähkön yrittämisen jälkeen huomaan todennäköisen syyn toiselle bugille – sovelluksen siirtyessä taka-alalle, eräät sovelluksen taustaprosessit katkeavat. Taka-alalle siirtymisen aiheuttaa tässä tapauksessa käyttöjärjestelmän oma toiminnallisuus, jota käytämme sovelluksen toiminnan tukena. Sovellus käyttää samaa koodia sekä iOS että Android käyttöjärjestelmille, ja tämä bugi toistuu ainoastaan Android-laitteilla. Etsittyäni ongelmasta tietoa verkkohaulla, selviää että kyseessä on tarkoituksellinen suunnitteluratkaisu Android-järjestelmässä, eikä sen korjaaminen tule olemaan helppoa. Ideoimme muutaman ratkaisun kollega A:n kanssa ongelman ratkaisemiseksi, mutta jätämme varsinaisen päätöksen huomiseen.

Tiistai 18.3.2025

Päiväkohtainen tavoite: Eri teknisten ratkaisujen punnitseminen, valitun ratkaisun käytännön toteutus

Olemme ideoineet kolme vaihtoehtoista ratkaisua eilen selvitetyn bugin ratkaisemiseen. Näistä on hankala valita parasta, sillä jokainen ratkaisu tulisi jokseenkin heikentämään Sovellus A:n

käyttäjäkokemusta. Ensimmäinen ratkaisu olisi sallia ongelman aiheuttava toiminnallisuus vasta kun taustaprosessit ovat valmiit, mutta tämä hidastaisi sovelluksen käyttöä. Toinen ratkaisu olisi tarkistaa jälkikäteen, että prosessit ovat onnistuneesti suoritettu taustalla, mutta tämä lisäisi mahdollisuutta uusille bugeille eikä olisi siksi ideaali ratkaisu. Kolmas ratkaisumme olisi taustaprosessit katkaisevan toiminnallisuuden korjaaminen kolmannen osapuolen kirjastolla, ja tätä pidämme järkevimpänä. Asiakkaamme on kuitenkin ilmaissut toiveensa välttää tätä ratkaisua, sillä heidän mielestään natiivitoiminnallisuuden toteutus on käyttäjäystävällisempi. Sovelluskehittäjänä ongelma on turhauttava: paras ratkaisumme on jo etukäteen hylätty asiakkaan puolelta. Valitsemme näistä ratkaisuksi prosessien tuloksen tarkistuksen ja tarvittaessa uudelleenajamisen. Lisään tarvittavat tarkistukset sekä uudelleenajokomennot koodiin, ja tällä saamme bugin korjattua.

Keskiviikko 19.3.2025

Päiväkohtainen tavoite: bugin korjauksen käytännön toteutus

Olimme jo luulleet bugin olevan korjattu, mutta olen saanut mieleeni vielä toisenkin harvemmin epäonnistuvan prosessin, jonka sama bugi saattaa rikkoa. Ryhdyn tämän korjaukseen, mutta huomaan, että sillä on lieviä ristiriitoja eilen toteutettujen muutosten kanssa. Kulutan suurimman osan päivästä kirjoittaessani ratkaisua, joka sopii molempiin tarkoituksiin. Juuri kun olen saanut korjauksen valmiiksi, kertoo kollegani keskustelleensa asiakkaan kanssa. Hän on saanut neuvoteltua asiakkaan hyväksymään kolmannen osapuolen kirjaston käytön, ja korjaus on sittenkin päätetty tehdä sillä menetelmällä. Vaikka toista päivää työstämäni ratkaisu osoittautuukin täten turhaksi, olen saanut tästä arvokasta kokemusta. Samankaltainen ongelma Android-käyttöjärjestelmän ominaisuuksista johtuen voi hyvin toistua jatkossakin, ja nyt minulle on selvää, mikä sen todennäköisesti aiheuttaa.

Torstai 20.3.2025

Päiväkohtainen tavoite: Bugin tunnistaminen ja korjaus, React Native event-kuuntelijat ja dynaaminen tyylitys

Olen löytänyt uuden bugin sovelluksestamme. Tämä liittyy edellisellä viikolla rakentamaani komponenttiin, jonka toiminnassa havaitsen epäyhtenäisyyttä laitteesta riippuen. Uusimmilla Android-versioilla komponenttini toimii odotetusti, mutta vanhemmalla käyttöjärjestelmällä taikka pienellä näytöllä eräs sen toiminto ei enää toimi odotetusti. Komponentin tyylittely menee sekaisin käyttäjän avatessa näppäimistön. Parhaaksi ratkaisuksi arvioin tyylien dynaamisen määrittelyn, ja tätä rakentaakseni ryhdyn etsimään verkosta vertaiskokemuksia sekä aiheeseen liittyvää dokumentaatiota. Löydän React Nativen omista kirjastoista Keyboard-komponentin, jolla voimme seurata näppäimistön avaamista ja sulkemista. Toteutan tämän mukaiset muutokset

komponenttiini, mutta se ei siltikään toimi aivan odotetusti. Tyyliä dynaamisesti määrittävä tilamuuttuja saa arvonsa oikein vasta, kun näppäimistö on kertaalleen suljettu ja avattu jälleen uusiksi. Yritän muuttaa rakentamaani logiikkaa, tuloksetta. Viimein huomaan ongelman: olen laiskuuttani jättänyt tilamuuttujan TypeScript-määritelmän tekemättä, ajatellen sen olevan tarpeetonta. Lisättyäni oikean tietotyypin muuttujalle, saa se oikean, tarkoitetun arvonsa jo ensimmäisellä näppäimistön avaamisella. En ole aivan varma miksi muuttujan tyypittämisellä on näin iso merkitys, mutta kokemus on kuitenkin opettanut minua välttämään turhia oikopolkuja koodissani.

Perjantai 21.3.2025

Päiväkohtainen tavoite: sarjaporttikommunikaation optimointi

Sovellus A:n sprintti on valmis, mutta jätämme uuden version julkaisemisen seuraavaan viikkoon. Haluamme välttää mahdolliset viikonlopun aikana ilmenevät ongelmatilanteet, sillä emme pystyisi reagoimaan niihin tällöin tehokkaasti. Jatkan Sovellus C:n työstämistä. Esittelen aiempia aikaansaannoksiani kollega A:lle. Saan häneltä huomion sovelluksen toiminnallisuudesta, jonka voisi tehdä paremmin. Ajaessani sarjaporttien kautta komentoja yhdistetylle laitteelle, olen joutunut pysäyttämään laitteelta luettavan palautteen vastaanottamisen uuden komennon ajamista varten. Tämä on ollut siksi, että sarjaportin ollessa käytössä, ei portti hyväksy uutta yhteydenottoa. Kollegani on kuitenkin ratkaissut samankaltaisen ongelman jo toisella tuotteellamme, ja saan häneltä linkin tämän toteutuksen GitHubiin. Koodi ei ole suoraan käytettävissä Sovellus C:n toteutukseen, mutta saan tästä kuitenkin hyvän mallin omalle ratkaisulleni. Uuden yhteydenoton sijasta odotamme sopivaa taukoa laitteen palauttamassa syötteessä, ja vasta silloin välitämme uuden komennon yhdistetylle laitteelle. Näin voimme jatkaa laitepalautteen lukemista katkeamatta. Toteutan sovellukseen myös muutaman uuden testausominaisuuden.

Viikkoanalyysi 8

Viikon alku on painottunut käyttäjäpalautteena saatujen bugien ratkomiseen. Aiempi kokemukseni bugien korjaamisessa on painottunut joko itseni tai kollegani huomaamiin ongelmiin. Käyttäjien raportoimissa bugeissa tunnistin suurimmaksi haasteeksi itse bugien toistamisen. Sovelluskehittäjinä kollegoideni kanssa pystymme kenties intuitiivisemmin tunnistamaan bugin toistamiseen vaikuttavat tekijät, mutta käyttäjät eivät välttämättä osaa huomioida samoja asioita. Tästä syystä itse havainnoimiemme bugien toistaminen on helpompaa, sillä osaamme seikkaperäisesti selittää toisillemme, mitä vaikuttavia tekijöitä ohjelmiston käytössä on ollut bugin ilmaantuessa. Käyttäjiltä saatu palaute voi tästä poiketen olla hyvinkin yksioikoista, esimerkiksi pelkkä maininta bugin olemassaolosta. Pystymme tunnistamaan joitakin yhdistäviä tekijöitä käyttäjälokista, esimerkiksi sovellusversion taikka käyttöjärjestelmän, mutta usein silti tarkemmat

yksityiskohdat jäävät pimentoon. Yksityiskohtainen palaute bugin aiheutumiseen vaikuttavista tekijöistä on suuri apu sen selvittämisessä. Jos sovelluskehittäjä ei pysty toistamaan bugia, on sen ratkaisu hankalaa. Vaikka siihen saataisiinkin tiedostamatta tehtyä korjaus, ei kehittäjä voisi tätä varmasti tietää vertailupisteen puuttuessa (Bird 2012).

Käyttäjien raportoimien bugien selvittäminen alkoi pitkälti erilaisia käyttäjäskenaarioita läpikäymällä. Yritin keksiä erilaisia tilanteita sekä eri luonteille tyypillisiä tapoja käyttää sovellusta. Näitä systemaattisesti läpikäymällä sainkin lopulta toistettua meille raportoidut bugit. Prosessia voisi kenties sujuvoittaa helpottamalla käyttäjien bugiraportointiprosessia, esimerkiksi luomalla sovellukseen virtaviivaistettu, oleellisia tarkentavia kysymyksiä esittävä raportointiportaali.

Viikon kokemukset ovat myös osoittaneet, että teknisesti paras ratkaisu ei aina käy yhteen työn tavoitteiden kanssa. Bugin korjaukseen liittyvät töissä olimme aluksi ratkaisuiden suhteen rajoittuneita, sillä asiakas ei pitänyt parhaaksi näkemämme ratkaisun tuottamasta käyttäjäkokemuksesta. Vaikka kollegani saikin asiakkaan lopulta vakuutettua parhaaksi näkemämme menetelmän eduista, olisi myös vaihtoehtoinen ratkaisumme toiminut. Toteutukseen lopulta valikoitunut ratkaisu on kuitenkin tulevaisuuden toimintavarmuuden ja ylläpidettävyyden kannalta paras.

Viikon työtehtävät opettivat minulle myös huolellisen työn etuja. Rakentamani komponentin näppäimistöongelmien ratkaisu olisi ollut nopeammin valmis, mikäli olisin vaivautunut noudattamaan TypeScriptin muuttujien tyypittämissääntöjä. Vaikka tästä aiheutunut ajallinen menetys ei ollutkaan mittava, on tähän syytä kiinnittää huomiota jatkossa kehitysprosessin virtaviivaisuuden takaamiseksi.

Etenkin bugien ratkaisu on edistänyt ymmärrystä käyttöjärjestelmien eroista johtuvista haasteista mobiilisovelluskehityksessä. React Nativen suuri etu on sen mahdollistama yhden koodikannan käyttö useammalle alustalle, mutta siltikin joitain alustaperusteisia eroja tulee ottaa huomioon sovelluskehityksessä. Tästä konkreettisena esimerkkinä on ollut järjestelmien eriävä taustaprosessien hallinta: iOSilla sovelluksemme käynnistämät prosessit ajettiin onnistuneesti loppuun sovelluksen ollessa taka-alalla, kun taas Android ei kyennyt käsittelemään näitä taustalla. Ohjelmistokehittäjänä pystyn tämän kokemuksen pohjalta ennakoimaan tilanteita, joissa samankaltaiset ongelmat ovat todennäköisiä, ja suunnittelemaan toteutukseni nämä asiat huomioon ottaen.

Kokonaisuudessa viikolla saavutettu edistyminen työtehtävissä on kuitenkin tuntunut pieneltä. En koe saaneeni toteutettua yhtä suurta määrää tehtäviä kuin kenties joillain toisilla viikoilla. Toisaalta tämä on saanut minut pohtimaan, voiko työllistä menestystä mitata ainoastaan konkreettisin keinoin: GitHubiin vietyjen committien tai sarakkeesta toiseen siirrettyjen kanban-korttien määrällä.

Vaikka esimerkiksi bugin korjaamisessa suuri osa ajasta kului ratkaisuun, joka ei päätynyt lopulliseen toteutukseen, se velvoitti minut työntekijänä perehtymään syvällisesti koodin toimintaan sekä backendin ja frontendin keskinäisiin riippuvuuksiin. Etenkin urani alkuvaiheessa opittavaa on paljon, eikä kaikkiin tilanteisiin voikaan varautua. Tässä tapauksessa merkittävää onkin se, kuinka johdonmukaisesti kehitän omaa toimintaani näiden kokemusten perusteella.

4 Pohdinta

Lähtötilanteessa asettamani tavoitteet ammatilliselle kehitykselle olivat teknologian ja työkalujen käyttö, yhteistyö ohjelmistokehitystiimissä, sekä työhyvinvointi ja yhteisöllisyys. Käyn tässä pohdintaosiossa läpi keskeisimpiä työssäni kohtaamiani ja oppimiani asioita, jotka ovat edesauttaneet näiden tavoitteiden saavuttamista.

Yksi alkuvaiheessa tunnistamistani haasteista oli yhteistyöllisen ohjelmistokehityksen käytännön kokemuksen puute. Tämän osalta kehitystä onkin tapahtunut seurantajaksolla melko orgaanisesti. Aiemmissa, muiden alojen työtehtävissä opitut ja omaksutut yhteistyötä tukevat keinot olivat jo käytössä, varsinaisen oppimisen keskittyessä erityisesti ohjelmistokehityksen erikoispiirteisiin asian suhteen. Näitä erikoispiirteitä ovat esimerkiksi versionhallintaan ja tehtävien sekä vastuiden jakoon liittyvät asiat.

Versionhallinnalliset haasteet ilmenivät yhteistyöllisesti tuotettavan koodin työstössä. Useamman tekijän työskennellessä samalla projektilla, samaa koodikantaa käyttäen, on käytettävä menetelmiä ristiriitojen välttämiseksi. Näitä ristiriitoja luovat esimerkiksi päällekkäin työstettävä koodi, taikka keskenään yhteensopimaton toimintalogiikka. Versionhallinnan työkaluna olemmekin käyttäneet pääsääntöisesti GitHubia. Olen aikaisemmin käyttänyt GitHubia omien projektien sekä koulutöiden apuna, ja tämän ansiosta perusasiat olivat jo raportin lähtötilanteessa hyvin hallussa. Työtehtäväni velvoittivat minut kuitenkin syventymään entistä tarkemmin sen tarjoamiin ominaisuuksiin.

Yhteistyöllisessä työnteossa rutiinia on ollut oman koodin ajaminen GitHubin yhteiseen koodirepositorioon, ja vastavuoroisesti omalla työlaitteella olevan koodin synkronointi repositorion kanssa. Näin työntekijöiden paikalliset, työstettävät kopiot projektin koodikannasta pysyvät ajan tasalla muiden työntekijöiden tekemien muutosten kanssa. Ajan tasalla oleva paikallinen koodikanta on auttanut välttämään tilanteita, joissa eri tekijät tuottavat keskenään yhteensopimatonta koodia, tiedostamatta kollegoiden tekemiä muutoksia projektin muihin osa-alueisiin. Tämä on edellyttänyt projektien eri osallisilta myös oma-aloitteisuutta kollegoiden tekemien muutoksien tarkastelussa, jossa GitHubin muutoshistoriaominaisuus onkin auttanut suuresti. Muutoshistoriasta on nähtävissä erivärisin korostuksin sekä lisätyt että poistetut koodirivit. Muutoshistoria koostuu niin sanotuista commiteista, jotka ovat yksittäisen käyttäjän tallentamia muutoskokonaisuuksia repositoriossa. GitHubin projektirepositoriossa näille listataan commitin tekijä, luonnin ajankohta sekä tiedostot joihin commitilla on tehty muutoksia. Commitin luonut käyttäjä myös antaa sille lyhyen otsikon, johon hyvän tavan mukaisesti tulisi tiivistää sen sisältämien muutoksien tarkoitus. Hyvien käytäntöjen noudattamisen olen lukenut yhdeksi osa-alueeksi myös teknologian ja työkalujen käytön tavoitteessa, ja pyrkinyt näitä noudattamaan koko seurantajakson ajan.

GitHubin versionhallinnallisessa käytössä minulle hyödyllinen opittava asia on ollut etenkin haarojen käyttö. Toisinaan on ollut tarpeellista erottaa jonkin työprosessin toteutus omaan haaraan, jolloin käyttäjän sen hetkisestä haarasta projektin koodissa luodaan kopio. Näin ohjelmoijana olen voinut rakentaa uutta toiminnallisuutta häiritsemättä kollegoideni työprosessia. Syitä tämän eduille on monia. Yhdeksi esimerkiksi voidaan esimerkiksi nostaa koodin testaus: toiminnallisuuksien rakentaminen on usein iteratiivinen prosessi, joissa lopullinen ratkaisutapa löytyy vasta usean eri kokeilun jälkeen. Tämä oli erityisen tarpeellista esimerkiksi viikoilla 3 ja 4 rakentaessani Sovellus A:n uutta visuaalista komponenttia – komponentin kokeellisen luonteen ja laajan käyttöasteen vuoksi, oli suuri osa sovelluksen muusta toiminnasta puutteellista komponentin ollessa keskeneräinen. Erottamalla tähän liittyvä kehitystyö omaksi haaraksi kollegani pystyivät jatkamaan kehitystyötä muiden vaatimusten osalta, ja samalla myös tahtoessaan tarkastella ja testata erillistä työhaaraani. Näin sainkin kollegaltani arvokasta palautetta komponenttini toiminnasta, valitettavasti kuitenkin päätyen päätökseemme hyllyttää komponenttini jatkokehitys toistaiseksi. Tämäkin on olennainen osa yhteistyötä ohjelmistokehitystiimissä – muuten olisin todennäköisesti jatkanut tämän komponentin rakentamista pidempään, päätyen samaan lopputulokseen.

Olennainen yhteistyöllinen haaste liittyi myös tehtävien sekä vastuiden jakoon. Resurssien tehokkaan käytön vuoksi on ollut aiheellista välttää tilanteita, joissa useampi työntekijä työstää samaa asiaa itsenäisesti. Tässä tärkeintä on ennen kaikkea toimiva kommunikaatio, jota voidaan tehostaa erilaisin teknologisin menetelmin. Olemme käyttäneet töissämme verkkopalvelussa toimivaa Kanban-tyylistä taulua eri projektien tehtävien jakoon sekä edistymisen seurantaan. Tässä palvelussa eri tehtävät jaetaan niin sanotuille Kanban-korteille, joiden edistystä seurataan siirtelemällä niitä eri sarakkeisiin. Näitä sarakkeita voivat olla esimerkiksi: Odottaa – Työn alla – Testattavana – Valmis. Näin eri tehtävien edistyminen on ollut ohjelmointitiimin kesken nopeasti visuaalisesti havainnoitavissa. Tähän liittyen olenkin huomannut itselleni jatkokehitettävän kohteen. Kollegani muistuttivat minua säännöllisesti tekemään uusia Kanban-kortteja huomaamistani kehityksen kohteista tai korjattavista ongelmista, mutta tämän sijaan – kenties tottumuksesta yksin työskentelyyn – huomasin usein kirjoittavani näitä lähinnä henkilökohtaisiin muistilistoihini.

Kenties seurantajakson suurin ammatillinen kasvu on kuitenkin tapahtunut työkalujen ja teknologian käytön tavoitteessa. Työni on ollut pääsääntöisesti ohjelmointia, ja uusien ohjelmointimenetelmien opiskelu on ollut tämän ohessa väistämätöntä. Seurantajakson alkupuolella tehtäväni keskittyivät Sovellus A:n käyttämien kirjastojen päivitykseen. Koska päivitimme suuren määrän kirjastoja samanaikaisesti, johti tämä ristiriitoihin sovelluksen sen hetkisen koodiperustan kanssa. Kirjastojen päivittäminen on kuitenkin tärkeä osa sovelluksen elinkaarta. Päivitysten päämääränä on korjata bugeja, paikata tietoturva-aukkoja, parantaa suorituskykyä sekä ratkaista kirjastojen välisiä yhteensopivuusongelmia (Lolugu 2023).

Seurantajakson alun tilanteessa näistä viimeinen oli etenkin keskiössä – vaikkakin paikoin päinvastaisesti. Dominoefektin kaltaisesti, yhtä kirjastoa päivittäessä, toinen samasta kirjastosta riippuvainen kirjasto saattoi lakata toimimasta. Tämän ratkaisemiseksi jouduin lopulta päivittämään useita kirjastoja vuoron perään, sitä mukaa kun havaitsin uusia ongelmia. Prosessi oli opettavainen, enkä ollut aiemmin omissa projekteissani joutunut samaan tilanteeseen. Työstämämme sovellus on näitä huomattavasti monimutkaisempi, sisältäen suuren määrän eri kirjastoja, ja tämän päivityksestä kertynyt kokemus on valmistanut minut varautumaan näihin tilanteisiin tulevaisuudessa. Etenkin jos kyseessä on suuria versiopäivityksiä, on järkevämpää pyrkiä päivittämään vain rajattu määrä kirjastoja kerrallaan (Pina 2022).

Kirjastoihin liittyvien ongelmien selvittelyssä kerrytetty käytännön kokemus on opettanut minua kirjastojen käytössä ohjelmakoodin toteuttamiseen, sekä eri kirjastojen tarpeellisuuden arvioinnissa. Vaikka kirjastot helpottavatkin työtä, voi niiden turha käyttö monimutkaistaa ohjelman toimintaa. Mitä enemmän kirjastoja ohjelmassa käytetään, sitä todennäköisempiä myös edellisessä kappaleessa käsitellyt hallinnointiongelmat ovat. Jonkin asian ollessa käytetyssä kirjastossa rikki, on myös sitä käyttävä projekti riippuvainen kirjastoa ylläpitävän tahon kehitystyöstä tämän ratkaisemiseksi.

Työkalujen ja teknologian hyödyntämisen parissa saavutettu kehitys ei ole rajoittunut pelkästään syvempään tietämykseen kirjastojen päivityksestä ja ylläpidosta. Työssäni keskeisin tehtävä on ollut koodin tuottaminen, ja tämän tarkoitusperän toteuttamiseksi on koodikieliin erikoistuva osaaminen, ja etenkin siinä kehittyminen, ollut olennaista. Tiettyjen koodikielen erityispiirteisiin ja etuihin perehtymällä olen voinut käyttää niitä mahdollisimman tehokkaasti hyödyksi tuotettavassa koodissa. Kyvykkyys koodikielten hallinnassa mahdollistaa tehokkaan vuorovaikutuksen laitteiston kanssa, ja tämän kautta organisaation tarpeiden mukaisesti räätälöityjen ratkaistujen tuottamisen (American Public University 2024). Opinnäytetyöni seurantajakson aikaisessa työssä käytettyjen ohjelmointikielten keskiössä ovat olleet JavaScript/TypeScript sekä Python. Edeltä mainitun osalta kokemus on keskittynyt etenkin näiden käyttöön React Native kirjastoa hyödyntäen. Olen oppinut monia hyödyllisiä tekniikoita, jotka ovat tehostaneet React Nativen käyttöäni. Näistä työtehtävieni kannalta keskeisiä esimerkkejä ovat optimointimetodien hyödyntäminen, TypeScript tyypittäminen sekä propsien välitys.

Python oli minulle ohjelmointikielenä tuttu jo ennen seurantajaksoa. En kuitenkaan ollut käyttänyt tätä teknologiaa pitkään aikaan, ennen kuin sain sitä hyödyntäviä työtehtäviä. Viikolta 5 alkaen työtehtäväni keskittyivät toisinaan myös Sovellus C:n kehitykseen, joka on toteutettu Pythonilla. Seurantajakson jälkimmäisellä puoliskolla opiskelin kattavasti Pythoniin liittyvää dokumentaatiota, jotta pystyin hyödyntämään sitä tehokkaasti tavoiteltujen toiminnallisuuksien toteuttamiseksi. Keskeisimpiä opiskeltuja taitoja olivat etenkin Tkinter-kirjaston käyttö graafisen käyttöliittymän

luomiseksi, funktioiden säikeittäinen ajaminen, sekä laitteen kanssa kommunikaatio sarjayhteyden kautta. Graafista käyttöliittymää luodessa oli väistämättä tarpeen miettiä sen eri osien asettelua, ja tämän vaikutusta käytettävyyteen. Koodin eri toiminnallisuuksia säikeistämällä taas mahdollistetaan näiden samanaikainen suorittaminen (Iroegbu 2023). Näin sain esimerkiksi mahdolliseksi käyttöliittymän visualisoinnin dynaamisen päivityksen Sovellus C:n skannatessa yhteysportteja ulkoiseen laitteeseen. Täten pystyn välittämään käyttäjälle visuaalista palautetta skannausprosessin edetessä tulostamalla informatiivista tekstiä käyttöliittymään skannauksen ollessa vielä kesken, jolloin käyttäjälle on selvää, mitä ohjelma tekee taustalla. Samoin pystyin itsekin paremmin havainnoimaan sovelluksen toimintaa sitä kehittäessä, tehostaen työskentelyäni.

Optimointimetodien keskeisin tarkoitus on ohjelman suorituskyvyn parantaminen (Murtaza 2023). Suorituskyvyn parannukset saavutetaan metodien mahdollistamalla tehokkaammalla laskenta- ja muistiresurssien käytöllä. Tyypillisimmin näitä metodeja käytetään funktioille, jotka saavat usein käsiteltävikseen identtisiä parametrejä. Samoin parametrein näiltä voidaan odottaa samoja tuloksia. Funktioiden laskennan tulokset tallennetaan välimuistiin, jolloin turhaa laskentaa ja muistinkäyttöä voidaan välttää ajamalla nämä funktiot ainoastaan, kun niiden parametreinä saamat arvot poikkeavat niille aiemmin välitetyistä. Työssäni olin tätä hyödyntänyt etenkin luodessani omaa visuaalista komponenttiani Sovellus A:lle viikoilla 3 ja 4. Merkittävä suorituskykyä alentanut tekijä oli komponenttini laaja-alainen käyttö, usean kopion siitä ollessa näkyvillä samanaikaisesti. Komponenttini sisältämän koodin sisällyttäminen memoamismetodiin toi merkittävää parannusta sovelluksen suorituskykyyn, sen kyetessä arvioimaan onko näiden uusiksi laskennalle tosiasiallista tarvetta. Tälle opitulle taidolle tuli jälleen käyttöä viikolla 6, ja aiemman kokemukseni ansiosta pystyin nopeasti tunnistamaan tilanteeseen sopivan optimointimenetelmän ja toteuttamaan sen koodissamme.

Seurantajaksolla tapahtunut kehitys propsien välityksen ymmärtämisessä on ollut erityisen hyödyllistä viikoilla 7 & 8 työstämäni näkymiä yhtenäistävän komponentin rakentamisessa. Olin muiden töiden ohessa oppinut välittämään näitä tehokkaammin varsinkin TypeScript-tyypitystä hyödyntäen. Uutta komponenttia rakentaessa olen osannut määrittää TypeScriptillä sen toiminnalle tarpeelliset parametrit, sekä missä muodossa komponentin tulee nämä saada. Useimmat koodieditorit huomauttavat tämän perusteella kehittäjälle, mikäli esimerkiksi jokin prop on jäänyt välittämättä isäntäkomponentilta. Tämä on ollut työssäni tarpeellista esimerkiksi tilanteessa, jossa useampi isäntänäkymä käyttää samaa tekemääni komponenttia, mutta eri tarpein. Näitä oppeja hyödyntämällä onnistuin toteuttamaan modulaarisen, eri näkymät yhdistävän komponentin, jota käytämme laajasti Sovellus A:n koodiperustassa. Oletan tässä kertyneen kokemuksen olevan jatkossakin keskeinen osa tarvitsemaani osaamista ohjelmistokehittäjän työssä.

Työhyvinvoinnin ja yhteisöllisyyden tavoitteen seuranta osoittautui ongelmalliseksi seuranta-aikana. Vaikka koen näihin liittyvien tekijöiden tuntemuksen sekä hyvinvointia ylläpitävien toimenpiteiden suorittamisen tärkeänä osana ammatillista osaamista työssä kuin työssä, jäi tässä tapahtuneen kasvun seuranta muista tavoitteista jälkeen päiväraportoinnissani. Työtä suorittaessa muut tavoitteet ovat tuntuneet merkityksellisimmiltä, tämän tavoitteen kustannuksella. Tämän kokemuksen perusteella on selvää, että jatkossa minun tulee keskittyä enemmän myös kyseiseen teeman huomioimiseen oman kehitykseni kannalta työkyvyn ja jaksamisen ylläpitämiseksi.

Päiväkirjaraportointi on kokonaisuudessaan ollut hyödyllinen työkalu oman ammatillisen kasvun seuraamisessa. Analysoimalla toimintatapoja sekä kirjaamalla opittuja asioita ja käytettyjä menetelmiä, on ollut mahdollista seurata konkreettisia kehityksen mittareita. Näiden analyyttisellä seurannalla on myös ollut mahdollista tunnistaa puutteita ja vahvuuksia omissa toimintatavoissani, ja tämän perusteella kehittää toimintaani ja määrittää ammatillisen kehitykseni suuntaa. Olen kyennyt merkintöjeni perusteella myös paremmin huomaamaan aikaansaannokseni, sekä oman panokseni merkittävyyden työprojektiemme kannalta. Pystyn myös tulevaisuudessa käyttämään tässä opinnäytetyössä esittelemiäni tuloksia vertailupisteenä omalle jatkokehitykselleni.

Lähteet

Agile Alliance s.a. What is Agile? Agile Alliance verkkosivu. Luettavissa: https://www.agilealliance.org/agile101/. Luettu: 7.4.2025.

Agile Alliance s.a. Iteration. Agile Alliance verkkosivun asiasanasto. Luettavissa: https://www.agilealliance.org/glossary/iteration/. Luettu: 7.4.2025.

AirFocus s.a. What is a front end (in a website)? AirFocus asiasanasto. Luettavissa: https://airfocus.com/glossary/what-is-a-front-end/. Luettu: 7.4.2025.

Altexsoft Editorial Team, 27.9.2018. Code Refactoring Best Practices: When (and When Not) to Do It. Altexsoft blogi. Luettavissa https://www.altexsoft.com/blog/code-refactoring-best-practices-when-and-when-not-to-do-it/. Luettu: 11.3.2025.

American Public University, 31.1.2024. Information Technology Coding Skills and Their Importance. American Public University blogi. Luettavissa: https://www.apu.apus.edu/area-of-study/information-technology/resources/information-technology-coding-skills-and-their-importance/. Luettu: 15.4.2025.

Bhimani, K. 17.1.2024. Designing High-Performance UIs with React Component Wrapper. DhiWise blogi. Luettavissa https://www.dhiwise.com/post/designing-high-performance-uis-with-react-component-wrapper. Luettu: 18.3.2025.

Bird, J. 8.8.2012. Fixing Bugs – if you can’t reproduce them, you can’t fix them. JavaCodeGeeks blogi. Luettavissa: https://www.javacodegeeks.com/2012/08/fixing-bugs-if-you-cant-reproduce-them.html. Luettu: 25.3.2025.

BMC, 16.12.2024. What is Code Refactoring? How Refactoring Resolves Technical Debt. BMC Blogs, blogi. Luettavissa https://www.bmc.com/blogs/code-refactoring-explained/. Luettu: 4.3.2025.

Chhetri, R. 2.11.2023. Importance of UI- and UX design in Software Development. WeAre blogi. Luettavissa https://www.weare.fi/en/importance-of-ui-and-ux-design-in-software-development/. Luettu: 11.3.2025.

Deshpande, C. 29.8.2024. TypeScript vs. JavaScript: Which One is Better? Artikkeli, SimpliLearn opetusmateriaali. Luettavissa: https://www.simplilearn.com/tutorials/typescript-tutorial/typescript-vs-javascript. Luettu: 18.4.2025.

Expo. s.a. Introduction. Expo dokumentaatio. Luettavissa: https://docs.expo.dev/get-started/introduction/. Luettu: 7.4.2025.

SkillReactor Editorial Team, 29.12.2023. The Crucial Role of Skill Diversification in Tech Careers. SkillReactor blogikirjoitus. Luettavissa https://www.skillreactor.io/blog/the-crucial-role-of-skill-diversification-in-tech-careers/. Luettu: 4.3.2025.

Emelianov, D. 18.11.2023. Simplifying Complex Tasks: A Comprehensive Guide. Trimbox blogikirjoitus. Luettavissa https://www.trimbox.io/blog/simplifying-complex-tasks. Luettu: 4.3.2025.

Franklin, A. 12.2.2025. What is customer satisfaction? Definition + importance. Zendesk blogi. Luettavissa https://www.zendesk.com/blog/3-steps-achieving-customer-satisfaction-loyalty/#. Luettu: 11.3.2025.

Froehle, C. 12.6.2023. Why 100% Remote Work Doesn't…Work. Medium.com artikkeli. Luettavissa: https://medium.com/%40CRA1G/why-100-remote-work-doesnt-work-e6cb643a60ce. Luettu: 5.4.2025.

Geeks for Geeks 30.7.2024. What is Proof of Concept (POC) in Software Development? Luettavissa: https://www.geeksforgeeks.org/what-is-proof-of-concept-poc-in-software-development/ Luettu: 18.2.2025.

Gerges, M. 5.6.2023. Continuous Learning Strategies for Successful Developers. Kodeco verkko-opetusmateriaali. Luettavissa: https://www.kodeco.com/39765245-continuous-learning-strategies-for-successful-developers. Luettu: 18.3.2025.

Haapalinna, J. 6.5.2018. Tyyppijärjestelmät Web-ohjelmoinnissa. Kandidaatintyö. Tampereen teknillinen yliopisto, Tieto- ja sähkötekniikan kandidaatin tutkinto-ohjelma. Luettavissa: https://trepo.tuni.fi/bitstream/handle/123456789/26476/haapalinna.pdf. Luettu: 18.4.2025.

Heath, A. 6.8.2024. Effective Feedback Strategies at Work. WeThrive blogi. Luettavissa https://wethrive.net/performance-management-resources/effective-feedback-strategies-at-work/. Luettu: 11.3.2025.

Iankovych, N. 4.6.2023. Why Software Documentation Is Important. Steps to Create It in 2023. Medium.com blogi. Luettavissa: https://medium.com/@o.kornienko/why-software-documentation-is-important-steps-to-create-it-in-2023-820dcd2fd6fa. Luettu: 18.3.2025.

Iroegbu, S. 7.11.2023. Python Multithreading: Benefits, Use Cases, and Comparison. Pieces for Developers blogi. Luettavissa: https://pieces.app/blog/python-multithreading-benefits-use-cases-and-comparison. Luettu: 18.4.2025.

Kinsta, 17.11.2023. What Is GitHub? A Beginner's Introduction to GitHub. Kinsta tietopankki. Luettavissa: https://kinsta.com/knowledgebase/what-is-github/. Luettu: 7.4.2025.

Lolugu, K. 25.9.2023. Navigating the Importance of Package Updates. Medium.com blogi. Luettavissa: https://medium.com/design-bootcamp/navigating-the-importance-of-package-updates-53d230e3599b. Luettu: 10.4.2025.

Mastorio, T. 1.6.2018. How to escape tutorial purgatory as a new developer — or at any time in your career. FreeCodeCamp blogikokoelma. Luettavissa: https://www.freecodecamp.org/news/how-to-escape-tutorial-purgatory-as-a-new-developer-or-at-any-time-in-your-career-e3a4b2384a40/. Luettu: 4.2.2025.

Meltzer, R. 3.1.2023. What is a Programming Library? A Beginner's Guide. CareerFoundry blogi. Luettavissa: https://careerfoundry.com/en/blog/web-development/programming-library-guide/#what-is-a-programming-library. Luettu: 7.4.2025.

Messaoudi, F. 25.2.2021. The importance of planning before coding. Medium. Luettavissa https://fakhrymessaoudi.medium.com/the-importance-of-planning-before-coding-a75c6d1dfcbb. Luettu: 25.2.2025.

Mozilla s.a. What is JavaScript? Mozilla Developer Network verkkokurssi. Luettavissa: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript. Luettu: 18.4.2025.

Murtaza, A. 8.1.2023. React Performance Optimization: useMemo vs useCallback. Dev.to blogi. Luettavissa: https://dev.to/ahmedgmurtaza/react-performance-optimization-usememo-vs-usecallback-2p2a. Luettu: 15.4.2025.

N., C. 29.3.2023. Stepping Out of Your Comfort Zone: Why Feeling Incompetent is Normal During a Career Change. LinkedIn blogi. Luettavissa https://www.linkedin.com/pulse/stepping-out-your-comfort-zone-why-feeling-normal-navarro-mba-pmp. Luettu: 25.2.2025.

Pina, N. 5.7.2022. How to Update NPM Dependencies. FreeCodeCamp uutiset. Luettavissa: https://www.freecodecamp.org/news/how-to-update-npm-dependencies. Luettu: 10.4.2025.

Ramatis, C. 23.1.2024. What is JSDoc and why you may not need typescript for your next project? Dev.to blogi. Luettavissa https://dev.to/cherryramatis/what-is-jsdoc-and-why-you-may-not-need-typescript-for-your-next-project-54n1. Luettu: 18.3.2025.

React. s.a. React - The library for web and native user interfaces. React projektisivu. Luettavissa: https://react.dev/. Luettu: 7.4.2025.

React Native. s.a. React Native - Learn once, write anywhere. React Native projektisivu. Luettavissa: https://reactnative.dev/. Luettu: 7.4.2025.

Shah, K. 13.9.2024. Top 10 Tips for Working Remotely. Linezero blogi. Luettavissa https://www.linezero.com/blog/tips-for-working-remotely. Luettu: 10.2.2025.

Singh, S. 25.10.2023. Ammatillinen kasvu ja epävarmuuden hallinta. Merikratos blogi. Luettavissa https://merikratos.fi/blogi-ammatillinen-kasvu-ja-epavarmuuden-hallinta/. Luettu: 25.2.2025.

Thorndyke, K. 23.6.2021. How to Work With Code Written by Someone Else. CodeAcademy blogi. Luettavissa https://www.codecademy.com/resources/blog/how-to-work-with-code-written-by-someone-else/. Luettu: 10.2.2025.

Xu, T. 25.10.2021. What Is Google-Fu? Do You Have It? Builtin blogi. Luettavissa https://builtin.com/software-engineering-perspectives/google-fu. Luettu: 25.2.2025.