Haaga-Helia
ammattikorkeakoulu Oy

Työni sähköisen allekirjoituspalvelun kehittäjänä

Tomi Toiviainen

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2017

Haaga-Helia ammattikorkeakoulu Oy

Tiivistelmä

Päiväys

Tekijä(t)
Tomi Toiviainen

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön otsikko
Työni sähköisen allekirjoituspalvelun kehittäjänä

Sivu- ja liite-sivumäärä
45 + 1

Opinnäytetyön otsikko englanniksi
My worklife in electronic signature development

Opinnäytetyössä seurataan 12 viikon ajan työelämää sähköisen allekirjoituspalvelun kehittäjänä. Jokaiselle päivälle on asetettu ensin tavoite ja päivän päätteeksi on kerrottu mitä päivän aikana on saanut tehtyä. Viikon lopuksi viikko on purettu tarkemmin auki viikkoanalyysissä.

Signom Oy:ssä koostuu kymmenestä työntekijästä. Työkuvaani kuuluu asiakkaiden palveluiden ja yrityksen sisäisten työkalujen suunnittelu, toteutus ja testaus.

Opinnäytetyössä on kuvattu työympäristö, käytettävät työkalut, työtehtävien vaadittu osaaminen ja sidosryhmät. Seurantajakson päätyttyä pohditaan kirjoittajan kehitystä mikä on tapahtunut ajanjakson aikana.

Päiväkirjamaisen opinnäytetyön tarkoitus on seurata kirjoittajan kehittymistä työtehtävissään raportoinnin ajan.

Asiasanat
Ohjelmistokehitys, projektinhallinta, ohjelmointi

Sisällys

1 Johdanto

Opinnäytetyö on tehty päiväkirjamallilla, missä tulen kertomaan full stack ohjelmistokehittäjän arjesta 12 viikon ajan. Kerron jokaisen työpäivän alusta sen, mitä minulla on ohjelmassa ja lopuksi kirjoitan yhteenvedon päivästä. Yritän asettaa itselleni aina tiettyjä tavoitteita ja lopuksi pohdin, miten suoriuduin näistä tavoitteista. Viikkoanalyyseissa pureudutaan tarkemmin koko kuluneen viikon tehtäviin. Opinnäytetyön ajanjakso on 16.10.2017 – 5.1.2018.

Signom Oy on vuonna 2010 toimintansa aloittanut suomalainen yritys. Signom kehittää laajoja sähköisen asioinnin ratkaisuja, joiden ytimessä on sähköinen allekirjoitus. Signomin ydinosaaminen muodostuu teknologian, juridiikan ja helppojen käyttöliittymien yhdistämisestä. Signom tarjoaa yrityksille mm. sähköistä allekirjoituspalvelua, web-lomakkeita, luottotietojen tarkistusta ja maksusuorituspalveluita. Tuotteen voidaan jakaa melkein kahteen kategoriaan: yksityishenkilöiden ja PK-yritysten peruspalvelu sekä yritysasiakkaiden juuri heille räätälöidyt toteutukset.

Yritysasiakkaille tuotetaan yleensä lomakkeita heidän omalla ilmeellään, minne loppukäyttäjät ohjataan heidän omalta sivulta. Loppukäyttäjät täyttävät lomakkeen josta lopuksi generoidaan PDF-sopimus, niiden tietojen perusteella millä käyttäjä täytti lomakkeen. Kun lomake on täytetty ja sopimus luotu, kaikki sopimuksen allekirjoittavat osapuolet saavat sähköpostitse kutsun, missä pyydetään siirtymään Signomin sivulle ja allekirjoittamaan sopimus. Kun kaikki osapuolet ovat allekirjoittaneet sopimuksen, sopimus sulkeutuu ja siitä ilmoitetaan jokaiselle osapuolelle.

Peruspalvelussa asiakkaat saavat ladata sivuille omia sopimuksiaan ja kutsua niihin allekirjoittajia.

Tiimi on melko pieni, kymmenen ihmisen kokoinen tällä hetkellä. Tiimi koostuu tällä hetkellä seuraavista henkilöistä:

- 1 toimitusjohtaja/juristi
- 1 juristi
- 2 projektipäällikköä
- 2 senior developeria
- 1 tietoturvavastaava
- 3 junior developeria

Sain Signomilta työharjoittelupaikan tammikuussa 2017 ja 6 kuukautta kestäneen harjoitusjakson jälkeen allekirjoitin vakituisen työsopimuksen. Olin kolmen muun kanssa yrityksen ensimmäiset harjoittelijat. Koska järjestelmä on massiivinen, lähestulkoon kokonaan monoliitti –arkkitehtuurimallilla kirjoitettu ohjelmisto, ensimmäiset viikot menivät talon sisäisten perusasioiden opettelemiseen.

Keskeisimmät vaaditut osaamiset ovat:
• Ohjelmointikielet: Java, JavaScript, Python, Bash
• Tietokannat: MySQL
• Template Enginet: Thymeleaf, JSP, Velocity
• Arkkitehtuuri: Spring Framework, Google Cloud
• Versionhallinta: Git

2 Lähtötilanteen kuvaus

2.1 Oman nykyisen työn analyysi

Työtehtäviini kuuluu:

- Dokumentointi
    - Projektin sisäinen dokumentti
        - Toimintasuunnitelma mikä kirjoitetaan puhtaaksi ennen kuin varsinainen koodaus aloitetaan. Tarkoitus on jäsenellä kaikki projektin eri vaiheet niin tarkkaan kuin mahdollista, jotta itse tekemisvaiheessa ei tarvitse ihmetellä mitä tehdä seuraavaksi.
    - Prosessikuvaus
        - Graafinen kuvaus asiakkaan haluamasta prosessista. Esim. UML-kaavio. Tästä näkee nopeasti yleiskuvauksen siitä mistä projektissa on kyse.
    - Testausohjeet asiakkaalle
        - Pomminvarma step-by-step kuvaus palvelusta. Kerrotaan selkokielellä, miten palvelu toimii ja miten sitä käytetään. Tämä käydään yleensä asiakkaan kanssa läpi yhdessä, kun softa on asennettu testiympäristöön.
- Ohjelmointi
    - Front-end
        - HTML5, JavaScript, JSP, Thymeleaf, CSS.
    - Back-end
        - Java

        - Spring Boot
    - Testaus
        - Automaatiotestit kirjoitetaan Javalla käyttäen Selenium ohjelmistokehystä.
- Tietokanta
    - MySQL
        - Käytetään talon sisäistä graafista käyttöliittymää, mikä pyörii lokaalisti selaimessa, joten aivan täydellistä tietämystä MySQL:stä ei vaadita. Perusteet pitää kuitenkin olla hallussa.
- Asiakastuki
    - Käytämme talon sisäistä tikettijärjestelmää, minne asiakkailta ja loppukäyttäjiltä tulee lähes päivittäin tukipyyntöjä. Tukipyynnöt käsittelevät mm. asiakkaiden neuvomista, vikakorjauksia, selvityspyyntöjä, lisäpalveluiden käyttöönottoja ja lomakemuutoksia.

Työtehtäväni normaalissa asiakasprojektissa:

1. Saan projektipäälliköltä tai vanhemmalta kehittäjältä projektin speksit.
2. Dokumentoin projektisuunnitelman.
3. Vanhempi kehittäjä katselmoi suunnitelman, hyväksyy sen tai tekee korjauksia.
4. Aloitan ohjelmoinnin ja täytän dokumentaatiota samalla.
5. Kirjoitan yksikkötestit ja varmistan että ohjelma toimii niin kuin pitää.
6. Asennan ohjelman testiympäristöön missä asiakas pääsee testaamaan palvelua.
7. Asiakas antaa palautetta ja tehdään mahdollisia korjauksia sen mukaan.
8. Asiakas hyväksyy projektin ja se asennetaan tuotantoympäristöön.

Kehitysprojekti alkaa kun, asiakkaalta saadaan heidän vaatimukset siitä, mitä ohjelman pitää tehdä. Vaatimuksien perusteella kirjoitetaan käyttäjätarinat, minkä perusteella voidaan alkaa kirjoittamaan ylös eri tehtäviä, mitä projekti tulee vaatimaan. Kun projektisuunnitelma on valmiina, projektipäällikkö käynnistää ensimmäisen sprintin ja jakaa kehitystiimille tehtäviä eli taskeja. Kehitystiimin jäsenet arvioivat, kuinka paljon aikaa he tulevat käyttämään jokaiseen eri tehtävään. Näin saadaan karkea arvio siitä, kuinka paljon sprintin ensimmäinen aihe tulee viemään aikaa. Sprinttejä on yleensä monia riippuen projektin koosta ja kuin kaikki sprintit on tehty, projekti on valmis. Sprinttien välissä – ennen kuin ohjelma on kokonaan valmis – ohjelma usein asennetaan testiympäristöön, jotta asiakas pääsee testaamaan ohjelmaa ja voi antaa siitä palautetta. Kun asiakas on testannut ja hyväksynyt ohjelman, se asennetaan tuotantopalvelimelle.

Tuntuu että minulla tällä hetkellä hyvä perusosaaminen aiheissa joita tarvitsen tällä hetkellä ammatissani. Ongelma on kuitenkin se, että vaikka osaan perusjutut, en ole minkään aiheen erikoisosaaja. Pelkkään minulle tuttuun koodiin kuten Javaan liittyvät ongelmat osaan yleensä ratkaista pikaisella googletuksella. Suurin ongelma on Signomin omat palvelut ja niiden ymmärtäminen. En ole ikinä aikaisemmin työskennellyt näin massiivisen Java –ohjelman parissa, enkä vieläkään tiedä miten kaikki sen osat toimivat. Varsinkin kun kaikki koodi on jonkun toisen 5 vuotta sitten kirjoittamaa, on sen uudelleenkirjoittaminen välillä erittäin haastavaa. Käyttäisin tässä termiä ”OOP Hell”, eli ”Olio-ohjelmointi helvetti”. Tässä ei myöskään auta yhtään se seikka, että koko backend koodista ei löydy juurikaan mitään yrityksen sisäistä dokumentointia.

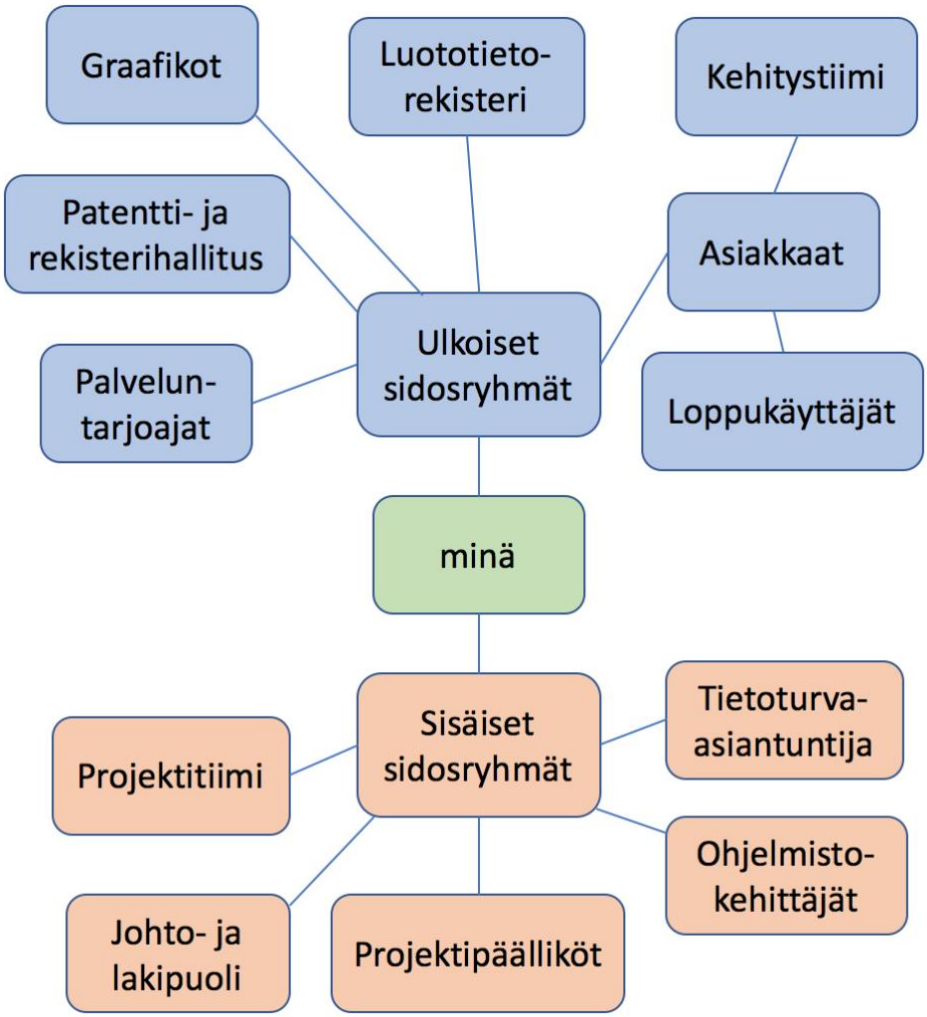
Koska tuotamme allekirjoituspalveluita, projekteissa käytetään myös paljon virallista lakijargonia, mikä on minulle täyttä hepreaa. Nämä onneksi selkenevät päivä päivältä enemmän.

2.2 Sidosryhmät työpaikalla

Isoin ulkoinen sidosryhmä ovat asiakkaat ja heidän loppukäyttäjät. Muita suuria ryhmiä ovat palvelun tarjoajat keiltä esimerkiksi vuokraamme servereitä ja Patentti- ja rekisterihallitus, mitä ilman meidän tarjoamat palvelut eivät olisi mahdollisia. Graafiset suunnittelijat auttavat välillä sivun ulkoasun kanssa. Asiakkailla on myös usein oma kehitystiimi, keiden kanssa teemme yhteistyötä, kun suunnittelemme heidän palveluaan. Asiakkaiden loppukäyttäjät riippuvat heidän toimialastaan. Rakennusalan asiakkaiden loppukäyttäjät voivat olla urakoitsijoita, mutta suurin osa loppukäyttäjistä on yksityisasiakkaita.

Sisäisiin sidosryhmiin kuuluu projektitiimi, johto- ja lakipuoli, projektipäälliköt, tietoturvavastaava ja ohjelmistokehittäjät.

Kuva 1: Sidosryhmäkaavio



2.3 Vuorovaikutustaidot työpaikalla

Eniten olen tekemisissä muiden ohjelmistokehittäjien ja projektitiimin kanssa. Projektitiimissä on yleensä vain yksi tai kaksi ohjelmistokehittäjää ja projektipäällikkö. Projektipäällikkö toimii aina ensikädessä suoraan asiakkaan kanssa, minkä jälkeen hän jakaa tehtävät kehittäjille. Johto- ja lakipuolen kanssa kommunikoin harvemmin. Projektipäällikkö konsultoi yleensä laki- ja johtopuolta esimerkiksi tarjousten luonnissa ja muissa lakia käsittelevissä asioissa. Tietoturvavastaavalta kyselen melkein päivittäin asioista, koska itseäni kiinnostaa tietoturva ja haluan pysyä ajan tasalla tietoturva-asioissa. Tietoturvavastaava on myös aina paikalla projektin loppukatselmoinnissa missä hän auditoi, että kaikki tehty parhaita tietoturvakäytäntöjä noudattaen.

Itse kommunikoin asiakkaiden kanssa harvemmin. Asiakaspalveluvuorossa olen kommunikoinut asiakkaiden kanssa sähköpostilla ja muutamassa projektissa olen ollut Skype-puhelussa asiakkaan kehitystiimin kanssa. Tukipyynnöissä asiakkaalta pitää usein

kysyä lisää tietoa ongelmasta, koska kaikki asiakkaat eivät ensimmäisessä viestissään osaa selittää ongelmaa tarpeeksi selkeästi.

3 Päiväkirjaraportointi

3.1 Seurantaviikko 1

Maanantai 16.10.2017

Alku

Olen koko edellisen viikon työstänyt keskikokoista asiakasprojektia. Asiakas halusi kokonaan uuden palvelun missä heidän urakoitsijat voivat tehdä valtakirjahakemuksia.

Urakoitsijan järjestelmävastaava klikkaa asiakkaan sivulta linkkiä mikä ohjaa hänet meidän palveluun. Hakemuksen voi tehdä vain, jos urakoitsijan sähköposti ja y-tunnus ovat sallittujen listalla. Tämän jälkeen urakoitsija täyttää vaaditut tiedot siitä lähtee ilmoitus sähköpostitse asiakkaan hakemuskäsittelijälle. Hakemuskäsittelijä tarkistaa tiedot oikeiksi, valitsee että joko toimitusjohtaja allekirjoittaa sopimuksen yksin, tai kaksi sopimuskäsittelijää allekirjoittaa sopimuksen kaksin. Kun sopimus on allekirjoitettu kaikkien toimesta saa urakoitsija sähköpostin siitä, että sopimus on luotu.

Tänään on Sprint2:n viimeinen päivä ja minun pitää kirjoittaa enää muutama automaatiotesti valmiiksi, viimeistellä dokumentaatio ja asentaa palvelu testiympäristöön.

Loppu

Kaikki sujui hyvin lukuun ottamatta muutamaa pientä ongelmaa, mitkä ilmenivät, kun ajoin automaatiotestejä. Softa itsessään toimi niin kuin pitikin, mutta testit eivät menneet läpi koska yleensä testit kirjoitetaan niin, että uudet testikäyttäjät luodaan satunnaisilla arvoilla missä nimet ja henkilötunnukset vaihtelevat. Tässä projektissa asiakkaalla onkin ns. staattiset käyttäjät eli toimitusjohtaja on aina ”Timo Toimitusjohtaja” ja sama juttu sopimuskäsittelijöillä. Aina kun uusi käyttäjä rekisteröityy palveluun, tietokantaan menee ”vahvistetut sähköpostit” –tauluun asiakkaan sähköposti. Kun se on kerran mennyt sinne ei uutta asiakasta tarvitse enää rekisteröidä uudestaan. Jouduin korjaamaan testit niin, että staattiset henkilöt luotiin vain kerran testin alussa. Muuten testit olisivat epäonnistuneet, jos ne olisi ajettu nykyisessä tilassa muissa ympäristöissä.

Testit korjattua suoritin testiasennuksen, eli migratoin tietokantamuutokset testiympäristöön, yhdistin git-branchin missä olin työskennellyt testi-branchiin ja käänsin

testiympäristön uusilla koodeilla. Ajoin vielä automaatiotestit testiympäristöä vasten ja kaikki toimi niin kuin pitikin. Opin miten testit tulee kirjoittaa, jos pitää testata palvelua missä käyttäjät ovat vakiot.

Tiistai 17.10.2017

Alku

Tänään minun piti saada speksit uuteen projektiin, mutta molemmilla senior devillä oli kiireisempää tekemistä eikä kumpikaan ehtinyt tehdä sitä. Jos olisin saanut speksit niin tavoite olisi ollut ymmärtää paremmin meidän tunnistautumispalvelua, koska en ole vielä ikinä joutunut sen kanssa

Loppu

Tavoite oli aloittaa uusi projekti mutta kun en saanut speksejä, aloin omatoimisesti parantamaan aikaisemman projektin dokumentaatiota. Piirsin hienon UML-kaavion valtakirjahakemusprosessin Flow –rakenteesta. Flow on yrityksen sisäinen termi mikä hoitaa softan logiikan back-end puolella. Se on eräänlainen ”state machine” mikä on kirjoitettu ”fluent interface” -tyylillä, missä metodit ketjutetaan toisiinsa. UML-kuvaus flow prosessista on hyvä olla selkeä, koska jos (ja kun) projektia jatkokehitetään, saa sitä silloin työstävä ohjelmoija nopeasti selkeän kuvan siitä, miten flow prosessi toimii.

Huomasin myös sattumalta, kun googletin ”signom”, että hakutuloksissa ei näkynyt meidän landing pagea ollenkaan, vaan ensimmäinen hakutulos vei suoraan meidän asiakkaille tarkoitetulle sivulle. Ilmoitin ongelmasta slackissa, ja päätimme että hakukoneoptimointia tulisi tehdä joskus tulevaisuudessa, että saisimme ongelman korjattua.

Keskiviikko 18.10.2017

Alku

Saan tänään uuden projektin speksit. Tavoite on tutustua paremmin meidän palveluun, mikä hoitaa tunnistautumisen, ja se on ilmeisesti eriytetty meidän muista palveluista eli se on ns. microservice. Huomasin myös, että Jenkins integraatiotesteissä oli epäonnistunut yksi minun kirjoittama testi. Jenkins on automatisoitu serveri (continuous integration) missä testit pyörivät päivittäin eri ympäristöissä.

Loppu

Korjasin Jenkins testin. Siinä oli outo vika, missä robotti tarkisti, että sähköpostit lähtevät oikeaan osoitteeseen, mutta välillä robotti yritti avata sivua 2 vaikka sitä ei ollut olemassa. Vaikka en saanut testiä epäonnistumaan, kun ajoin sen lokaalisti, kirjoitin pienet korjaukset, tein uuden Jenkins buildin mikä ajoi pelkän epäonnistuneen testin (täysi integraatiotesti vie tällä hetkellä meillä noin 8 tuntia) ja testit menivät läpi.

Torstai 19.10.2017

Alku

Tänään asiakas tulee meidän toimistolle käymään läpi heidän uutta palvelua, minkä sprint2:n sain valmiiksi maanantain. Käymme yhdessä läpi testiversiota ja selitämme miten se toimii. Asiakaspalaverin jälkeen on tarkoitus korjata yhden toisen asiakkaan palvelun tyylit, koska heille tulee uusi ulkoasu. Tavoite tänään on saada kokemusta asiakkaan kanssa toimimisesta ja harjoitella CSS tyylejä.

Loppu

Palaveri asiakkaan kanssa meni sujuvasti. Osasin jopa vastata kaikkiin teknisiin kysymyksiin koskien uutta palvelua, vaikka projektipäällikkö hoitikin suurimman osan puhumisesta. Aluksi hieman jännitti riittääkö oma tietämys, mutta kaikki meni hyvin!

Lähdin iltapäivällä kotiin, koska minulla oli noussut kuume.

Perjantai 20.10.2017

Alku/ Loppu

Olin sairaslomalla.

Viikkoanalyysi

Viikko meni varsin sujuvasti ennen kuin tulin kipeäksi. Huomasin että minulla on suuria tietoaukkoja siitä, kuinka meidän palvelu toimii, koska en ollut aikaisemmin käynyt kaikkia prosesseja manuaalisesti läpi. Tämä johtuu siitä, että kun tulin töihin, opettelimme ensin kuinka automaatiotestit kirjoitetaan, enkä tietenkään sen jälkeen ole jaksanut manuaalisesti testata mitään, koska se vie enemmän aikaa. Kävin manuaalisesti läpi viikon aikana monia perusjuttuja, tämän jälkeen testien kirjoittaminenkin oli helpompaa, koska tiesi tasan tarkkaan mitä ne tekivät.

Nyt kun asiakas oli speksannut, että palvelussa pitää olla vain kolme tiettyä henkilöä, ketkä saavat tehdä toimintoja, ilmeni testauksessa jälkeenpäin ongelmia.

Ongelmat johtuivat siitä että, kun käyttäjä rekisteröityy palveluun sähköpostilla, sähköposti menee tietokannan tauluun, missä on kaikki verifioidut sähköpostiosoitteet. Yhdelle sähköpostille voi siis rekisteröidä vain yhden uniikin käyttäjän. Testit piti kirjoittaa siten, että käyttäjät luodaan vain kerran silloin, kun testiluokka käynnistyy. Tietokannasta yhden käyttäjän kokonaan poistaminen osoittautui myös haastavaksi työksi, koska monet eri käyttäjätaulut referoivat toisiinsa, eikä meillä ollut siihen valmiiksi kirjoitettua SQL-lausetta. En myöskään halunnut vielä ns. nollata lokaalikantaa missä testikäyttäjät olivat, koska olin ehtinyt tehdä sinne jo lukuisia muita muutoksia. Hyväksyin tappioni ja palvelua testatessa loin jatkuvasti uusia käyttäjiä nimettynä tyyliin: Teppo Testaaja, Teppo2 Testaaja, Teppo3 Testaaja jne. Jos olisin tajunnut tämän aikaisemmin, olisin tehnyt kaikki tarpeelliset kantamuutokset ennen käyttäjien lisäämistä. Näin testaaminen olisi ollut paljon helpompaa ja nopeampaa, koska olisin vain nollannut lokaalikannan joka kerta, kun testit piti ajaa uudestaan.

Automaatiotestit tulisi olla aina sellaiset, että ne pyörivät läpi ongelmitta ohjelmiston ympäristöstä huolimatta. Testiympäristössä meillä pyörii siis jatkuva integraatio, joka rakentaa kannan aina uudestaan sen omilla asetuksilla ennen jokaista buildia. Kun sain ohjelman valmiiksi testiympäristöä varten, jouduin muuttamaan tietokannasta jokaisen käyttäjän taas ns. normaalimuotoon (nimessä ei numeroa) ja kirjoittamaan testiluokan uusiksi niin, että se menisi läpi, kun käyttäjät luodaan taas joka kerta uudestaan.

Opin tällä viikolla enemmän meidän omasta tietokannasta ja siitä, miten testit tulisi kirjoittaa, kun ei voi jokaisen testiajon yhteydessä luoda aina uusia satunnaisia käyttäjiä.

3.2 Seurantaviikko 2

Maanantai 23.10.2017

Alku/ Loppu
Olin sairaslomalla.

Tiistai 24.10.2017

Alku/ Loppu
Olin sairaslomalla.

Keskiviikko 25.10.2017

Alku

Sain tehtäväksi sairasloman jälkeen käsitellä tukipyyntöjä. Asiakas1 haluaa uuden PDF:n ja uuden tekstin yhteen lomakkeeseen. Asiakas2:lle pitää korjata vika, missä asiakas pääsee allekirjoittamaan asiakirjan ennen kuin heidän asiakas on käynyt sen allekirjoittamassa. Asiakas3 pitää korjata vahvistusteksti, kun on allekirjoittanut. Asiakas4 haluaa, että lomakkeen päivämäärän valinnassa jokaisen kuun viimeinen päivä on korostettu. Asiakas5 haluaa uuden logon. Asiakas6 haluaa, että voivat pääkäyttäjänä antaa toisille muitakin rooleja kuin sopimuskäsittelijä. Tavoitteena tehdä mahdollisimman paljon korjauksia ja opetella Gimpin käyttöä.

Loppu

Korjasin lomakkeen tekstit ja sain projektipäälliköltä uuden PDF:n, missä uudet tagit ja lähetin sen SCP:llä (Secure copy, komentoriviohjelma, jolla tiedostoja kopioidaan SSH-protokollan yli) testiserverille. Ennen tätä, tarkistin, että uudet kentät tägittyivät oikein. Tämän jälkeen ajoin testikoneella uuden buildin ja testasin vielä uuden PDF:n manuaalisesti testiympäristössä. Asiakas2:n vika oli helppo korjata, piti vain vaihtaa yhden allekirjoitusosapuolen ”action stage”, eli numero missä järjestyksessä sopimuksen saa allekirjoittaa. Uuden logon tekeminen Gimpillä meni hienosti, vaikka omat kuvankäsittelytaidot ovat huonot. Piti onneksi vain pienentää kahta logoa ja liimata ne yhdeksi logoksi, niin se ei tuottanut suurempia ongelmia.

Torstai 26.10.2017

Alku

Jatkan tehtäviä Asiakas4:sta, elikkä kalenteri pitää toteuttaa jQueryn Datepicker widgetillä niin, että seuraavasta kuukaudesta eteenpäin jokaisen kuukauden viimeinen päivä on korostettu. Projektipäällikkö oli saanut isolta asiakkaalta lisää korjauksia aikaisempaan projektiin minkä toteutin, joten käymme ne hänen kanssaan läpi puoleltapäivin palaverissa. Tavoitteena opetella jQueryn Datepickeriä.

Loppu

Sain Datepickerin toimimaan halutulla tavalla. Apuna tähän käytin jQueryn loistavaa dokumentaatiota (http://api.jqueryui.com/datepicker/). Kuun viimeisen päivän sain kätevästi kun lisäsin Date-objektin kuukauteen +1 ja päivään -1. Kuume nousi taas iltapäivällä joten lähdin kotiin huilimaan.

Perjantai 27.10.2017

Alku/ Loppu
Olin sairaslomalla.

Viikkoanalyysi

Olin suurimman osan kipeänä tästä viikosta ja kun olin töissä en ollut täysin terveenä. Siitä huolimatta sain mielestäni paljon aikaan ja tein ensimmäistä kertaa sekalaisia pieniä korjauksia isojen projektien sijaan.

Suurin osa pienistä korjauksista tulevat meillä tikettijärjestelmän kautta, mitä aina jompikumpi projektipäällikkö päivystää. Järjestelmään tulee tikettejä asiakkailta ja ne koostuvat vikailmoituksista, yleisistä kysymyksistä ja muista sekalaisista tukipyynnöistä. Näitä korjauksia ei yleensä laskuteta tuntiperusteisesti, mutta jos muutokset ovat tarpeeksi laajat, projektipäällikkö arvioi siihen käytettävän työmäärän ja välittää asiakkaalle tarjouksen muutoksista. Jos tarjous hyväksytään, projektipäällikkö taskittaa projektin ja se annetaan eteenpäin ohjelmoijille. Tämän viikon hommat olivat kaikki sen verran pieniä, että niistä ei ketään asiakasta laskutettu.

Huomasin että kun on paljon pieniä juttuja korjattavana eri asiakkaiden portaaleissa, kannattaa pitää muistivihkoa (itse suosin TextWrangler-nimistä ohjelmaa), minne kirjaa kaikki muutokset mitä on tehnyt. Varsinkin tietokantamuutokset on hyvä kirjata aina ylös, koska eri asiakkaiden muutokset tehdään omiin Git-brancheihin, ja jokaisella branchilla on oma versionsa tietokannasta. Koodimuutoksia ei ole niin tärkeä kirjata ylös, koska niihin tehdyt muutokset näkyvät aina Gitissä. Käytämme SourceTree nimistä graafista Git-käyttöliittymää, mistä on helppo seurata jokaista branchia ja sieltä näkee nopeasti kaikki muutokset mitä on tehnyt kyseisen repositoryn tiedostoihin.

3.3 Seurantaviikko 3

Maanantai 30.10

Alku/Loppu
Olin sairaslomalla.

Tiistai 31.10

Alku/Loppu
Olin sairaslomalla.

Keskiviikko 1.11

Alku

Jatkan ison asiakkaan projektia, tavoitteena saada paljon asioita tehtyä koska olen ollut kipeänä ja sprintin palautus asiakkaalle on suunniteltu torstaiksi. Jos en kuitenkaan saa kaikkea tehdyksi, projektipäällikkö sanoi, että ok jos palautetaankin vasta perjantaina. Tavoitteena tehdä niin nopeasti hommia kuin mahdollista.

Loppu
Asiakas halusi muuttaa viestiä, mikä näytetään käyttäjälle, kun hän on esitäyttänyt lomakkeen. Tämänlaista ominaisuutta meidän palvelussa ei normaalisti ole, joten joudun tekemään sen itse. Kirjoitin meidän Flow -javakoodeihin uutta instruction luokkaa, mikä tallentaa sivulle kustomoidun infoviestin. Uusi metodi withInfoMessage ottaa parametrinä string-arvon, asettaa sen infoMessage attribuutin arvoksi ja palauttaa itsensä. Kun esitäytetyn lomakkeen tiedot tallennetaan, ohjelma tarkistaa onko infoMessage-attribuutti null. Jos ei ole null eli ohjelmoija haluaa muokatun infoviestin, se laitetaan päälle, jos on null, niin infoviestiksi laitetaan sen default arvo. Kun muokataan koodia mitä kaikki palvelut käyttävät, pitää olla erityisen tarkka, että muutokset eivät vahingossa riko muita prosesseja.

Torstai 2.11

Alku
Jatkan ison asiakkaan pyytämiä korjauksia. Asiakas on erittäin tarkka infoteksteistä mitä palvelun tulee ilmoittaa asiakkaalle, kun he tekevät tiettyjä toimenpiteitä. Tämä vaatii peruspalvelun koodien muokkaamista. Tavoitteena tänään saada niin paljon hommaa tehtyä kuin mahdollista.

Loppu
Ajoin ensin automaatiotestillä koko palvelun läpi, että sain kiinni, mikä teksti käyttäjälle näkyy, kun hän allekirjoittaa sopimuksen. Tämä prosessi on erilainen kuin muut, koska vaikka sopimus vaatii kaksi allekirjoitusta asiakkaan puolesta, sopimusta ei tässä tapauksessa allekirjoiteta jaetulla oikeudella vaan molemmilla osapuolilla on

yrityssidonnainen allekirjoitusoikeus. Löysin Java-luokan, mikä asettaa infotekstit, kun asiakas allekirjoittaa sopimuksen ja kirjoitin sinne uuden if-putken, mikä muuttaa tekstiä jos kyseessä juuri tämä asiakas ja sopimuksen allekirjoittaja on sellainen käyttäjä, mille on toimitusjohtajan puolesta myönnetty allekirjoitusoikeus. Tässä tekstissä allekirjoittajalle kerrotaan, että sopimus on allekirjoitettu, mutta vaatii vielä toisen allekirjoituksen. Jos allekirjoittaja on itse toimitusjohtaja joka allekirjoittaa aina yksin, infotekstiksi tulee viesti missä kerrotaan suoraan, että sopimus on allekirjoitettu ja suljettu.

Perjantai 3.11

Alku

Sain tehtäväksi kirjoittaa meidän palveluun uuden sisäisen ominaisuuden, joka tarkistaa, että asiakkaan selain tukee uutta TLS-standardia. Transport Layer Security (TLS), aiemmin tunnettu nimellä Secure Sockets Layer (SSL), on salausprotokolla, jolla voidaan suojata Internet-sovellusten tietoliikenne IP-verkkojen yli. Turvallisuussyistä johtuen palveluamme ei voi enää käyttää 1.1.2018 eteenpäin vanhemmilla selaimilla jotka eivät tätä tue. Jos asiakkaan selain ei tue TLS 1.2 versiota, tulee sivun ilmoittaa käyttäjälle, että hänen tulee päivittää selaimensa 1.1.2018 mennessä.

Loppu

Ehdin tänään saada vain speksit TLS-tarkistukseen, jonka jälkeen meillä alkoi kaksi tuntia kestävä palaveri koskien uutta Google Cloud -arkkitehtuuria, mikä meillä on nyt pikkuhiljaa pienissä paloissa otettu käyttöön. Palaverin päätteeksi aloimme juhlimaan, koska pikkujoulut.

Viikkoanalyysi

Viikon aikana ilmeni ongelmia yrityksen sisäisessä projektinhallinnassa. Siirryimme hiljattain käyttämään Outlookin kalenteria resurssivarauksiin, jotta kaikki näkevät nopeasti siitä mitä hommaa jokaisella on millekin päivälle. Käytimme tähän aikaisemmin Salesforcen Taskray –appia, mutta luovuimme siitä ja nykyään käytössä Atlassianin JIRA, missä projekteja hallinnoidaan ketterällä kehitystavan mallilla. Kaikkien mielestä tämä ei kuitenkaan ole nyt paras ratkaisu, ja osa haluaisi takaisin Taskforcen, missä on laajemmat analyyttiset työkalut työnseurantaa varten. Ohjelmoijan näkökulmasta en haluaisi aina jokaista pientä tehtävää taskittaa erikseen, koska siihen menee paljon turhaa aikaa, kun pitää miettiä jokainen pikkujuttu erikseen ja arvioida paljon siihen menisi aikaa. Varsinkin alkuvaiheessa tämä oli haasteellista koska, kun tekee asioita ensimmäistä kertaa, on mahdotonta arvioida kuinka, paljon siihen menee aikaa.

Yksi ongelmista joka, tuli ilmi on se, että kun meillä on vain kaksi senior ohjelmistokehittäjää jotka ovat kahdestaan rakentanut koko massiivisen ohjelman ja loput ohjelmoijat ovat vielä junior kehittäjä -tasolla, työnteko keskeytyy usein kokonaan, kun juniorille tulee eteen ongelma mitä hän ei osaa ratkaista. Päätimme siis, että aina daily-palaverin yhteydessä junior kehittäjät voivat varata senioreilta maksimissaan noin 30 minuuttia kyselyaikaa epäselviin asioihin liittyen. Tämä sovittiin sen takia, että senior kehittäjät eivät pysty työskentelemään tehokkaasti, jos joku on repimässä hihasta jatkuvasti 15 minuutin välein.

Olen pannut merkillä, että vaikka pystyn jo ilman apua perustamaan uuden asiakkaan peruspalvelun ja tekemään siihen kaikki tarvittavat lomakkeet, sähköpostipohjat ja tietokanta konfiguraatiot, minun pitäisi opiskella lisää Javaa, jotta ymmärtäisin meidän softan ”ison kuvan” paremmin. Osaan Javaa sen verran hyvin, että osaan muokata jo valmiiksi kirjoitettuja luokkia niin, että ne toimivat omassa projektissani, mutta jos sellainen pitäisi kirjoittaa alusta asti ilman mitään mallia, en usko, että pystyisin siihen. Pitäisi joku päivä vaan töitten jälkeen käydä tarkemmin lähdekoodeja läpi, ja koittaa samalla tavalla kirjoittaa oma testiohjelma mutta pienemmässä mittakaavassa, niin saisi varmasti asioista paremmin kiinni. Työaikana kun pitää aina tehdä pelkästään töitä mitä silloin annettu, niin itse koodin opiskelulle harvemmin jää aikaa.

Java on onneksi jo niin vanha kieli, että sille on kehitetty toimintamalleja mitä tulisi noudattaa, koska ne on todettu parhaiksi tavoiksi tehdä asioita. Nämä ns. best practices eli parhaat toimintamallit tulisi jokaisen Java-ohjelmoijan osata.

Yksi näistä on Fluent API –malli, missä selkeästi nimetyt metodit ketjutetaan. API metodit eivät siis ole normaaleja gettereitä ja settereitä, vaan ne palauttavat toisen domain olion tai itsensä, ja niille voi antaa filter parametrejä (Mitra 2016). Meidän Flow –luokka on kirjoitettu juuri tällä tavalla. Toinen malli mitä siinä hyödynnetään, on ns. state machine –malli. Ensin kirjoitetaan wrapper –luokka mikä pitää sisällään listan eri tiloista. Tämä luokka tietää aina missä tilassa ollaan ja se määrittää mistä tiloista voi siirtyä toisiin tiloihin (Shvets 2017). Flow –luokassa käytämme näitä molempia toimintamalleja yhdessä.

Vaikka en aluksi tiennyt kummastakaan toimintamallista, osasin silti vanhoja koodeja lukemalla kirjoittamaan sinne uudet logiikat, sillä koodi oli erittäin helppolukuista juuri näiden parhaiden toimintamallien takia. Pelkistetyssä sopimuksenluontiprosessissa ensimmäinen tila olisi ”näytä sivu”, missä ajetaan instruction –metodit ”lataa sivu” ja ”aseta perusarvot”. Kun nämä on ajettu ilman virheitä, transition –metodi vaihtaa tilan ”odota

käyttäjän inputtia” –tilaan, missä ollaan niin pitkään, kunnes käyttäjä lähettää lomakkeelle POST-kutsun. Tästä tilasta siirrytään ”validointi” –tilaan, missä tarkistus –metodi tarkistaa, että kaikki syötetyt tiedot ovat virheettömät. Sama prosessi toistuu niin pitkään, kunnes sopimus on luotu. Toimintamalli on loistava, koska sinne on helppo kirjoittaa uusia tiloja, kun niitä tarvitsee ja jokaiselle tilalle voi kirjoittaa uusia instruction –metodeja, mitkä tekevät juuri sitä mitä asiakas on pyytänyt.

3.4 Seurantaviikko 4

Maanantai 6.11

Alku

Tänään pitäisi saada TLS-tarkistus toimimaan, mikä vaatii varmasti JavaScriptiä. Itse koodin saa varmasti GitHubista valmiina jonku toisen kirjoittamana, joten tavoitteena on etsiä toimiva koodi ja opetella kuinka se implementoidaan meidän järjestelmään. Ensin pitää saada selville järkevin toteutustapa ja käydä senior kehittäjän kanssa se läpi niin että, se hyväksytään. Loppupäivästä päivystän asiakaspalveluvuorossa, eli käyn läpi satunnaisia asiakkaiden tukipyyntöjä.

Loppu

Toteutin TLS-tarkistuksen googlettamalla ensin pienen JavaScript pätkän, mikä palauttaa selaimen version, jonka lisäsin omalle sivulleen. Sen jälkeen lisäsin sen script-tageihin itse palvelun JSP-sivulle mikä näyttää ilmoitukset, ja tein uuden divin, missä on teksti, joka kertoo asiakkaalle, että selain tulisi päivittää. Div on normaalisti aina piilossa, mutta siitä poistetaan nyt ”no-display” attribuutti, jos selaimen TLS-versio ei ole 1.2.

Loppupäivän selvitin asiakkaan tukipyyntöä, missä hän ei saanut allekirjoitettua dokumenttia, vaikka hänellä pitäisi olla siihen oikeudet. Hain tuotantokannasta ensin sen sopimuksen tiedot, mitä asiakas oli yrittänyt allekirjoittaa. Kannan tiedot olivat kunnossa ja ajoin lokaalisti testit vielä läpi eikä niissäkään löytynyt ongelmia. Seuraavaksi hain tuotannon logeja asiakkaan ip:llä ja sieltä huomasin, että asiakas oli kirjautunut väärään palveluun, missä hänellä ei ollut oikeuksia allekirjoittaa sopimusta. Kuittasin tiketin suljetuksi ja neuvoin asiakasta kirjautumaan oikeaan palveluun.

Tiistai 7.11

Alku

Asiakkaalla on palvelu, mikä lähettää rajapintakutsulla meille tiedot ja niiden tietojen perusteella tehdään sopimus. Asiakas saa kutsun jälkeen sähköpostin, mistä löytyy linkki allekirjoitussivulle. Tässä kohtaa asiakkaan pitää tunnistautua vahvasti palveluun, mikä on y-tunnussidonnainen. Ohjelma tarkistaa, että käyttäjällä on yrityksessään asiakirjojen allekirjoitusoikeus, ja että y-tunnus joka annettiin API-kutsussa vastaa käyttäjän y-tunnusta.

Asiakkaalle pitää luoda kolme uutta testikäyttäjää debug-tunnistukseen (nopeuttaa testaamista asiakkaille, heidän ei tarvitse yksitellen täyttää jokaista käyttäjän tietoa vaan teemme heille valmiit presetit).

Loppu

Tein kolme uutta debug presettiä. Presetit ovat tekstitiedostoja mihin syötetään dataa kuten, presetin nimi mikä näkyy dropdown valikossa, käyttäjän hetu, etu- ja sukunimi, oikeudet yrityksiin ja roolit kyseisissä yrityksissä. Näillä preseteillä siis simuloidaan PRH:n vahvaa tunnistautumista, mikä hoituu normaalisti tuotannossa pankkitunnukset syöttämällä. Tässä vaiheessa huomasin, että en tiedä juurikaan mitään yrityksen työntekijöiden virallisista rooleista yrityksessä. Opin uusia sanoja kuten prokuraattori.

Keskiviikko 8.11

Alku

Olin luonut edellisenä päivänä asiakkaalle uudet presetit debug tunnistautumista varten. Tänään sain kuitenkin kuulla, että uudet käyttäjätunnukset eivät toimineet. Ryhdyn selvittämään mistä tämä johtuu. Päivän tavoitteena ymmärtää paremmin debug-härveliä ja meidän omaa tietokantaa.

Loppu

Pitkän lähes koko päivän kestäneen väännön jälkeen selvisi, että vika johtui siitä, kun käyttäjä uuden yrityksen y-tunnuksella, oli kannassa ja yhdessä taulussa samalla tunnistusnumerolla tämä sama yritys. Näin ei kuitenkaan ikinä pitäisi tapahtua ja vika johtuu jostain vanhoista koodeista, mitkä pitäisi korjata. Tauluissa on kolme eri arvoa, minkä mukaan tunnistus tapahtuu. Joko PRH:ssa (Patentti- ja rekisterihallitus), testissä tai lokaalisti. Tässä tapauksessa tunnistus otti yhteyttä rekisterihallitukseen, vaikka näin ei pitäisi käydä. Opin uusia asioita siitä, miten meidän tunnistus tapahtuu ja kuinka välttää samankaltaisia ongelmia jatkossa.

Torstai 9.11

Alku

Tänään alkaa neljäs sprintti ison asiakkaan projektissa. Kävimme aamulla projektipäällikön kanssa läpi asiakkaan palautteet aiemmasta testiversiosta ja minulla on nyt selkeä kuva, mitä minun pitää tehdä. Tavoitteena ymmärtää tulevat muutokset ja dokumentoida ne niin, että minulla on selkeä kuva siitä mitä minun tulee tehdä ja missä järjestyksessä teen hommat.

Loppu

Tulikin iltapäivällä kiireinen asia, mikä tulisi hoitaa suoraan tuotantoon ja se annettiin minulle. Eräs asiakas haluaa tiedostosiirrot päälle ja heillä on kaksi erilaista sopimustyyppiä. Sopimus1 menee heille kansioon "signom/sopimus1/blabla.zip" ja sopimus2 kansioon "signom/sopimus2". Yritin selvittää tätä koko loppupäivän, mutta en kuitenkaan saanut sitä toimimaan. Tiedostot siirtyivät palvelimelle, mutta eivät oikeaan kansioon. Luovutin asian kanssa, koska en saanut kiinni siitä mistä ongelma johtui, enkä saanut yhteyttä vanhempiin kehittäjiin.

Korjasin vian ison asiakkaan palvelussa, missä esitäytetty lomake ohjautui väärään lomakkeeseen, jos asiakas oli syöttänyt tiedot väärin ja palasi lomakkeelle ”Takaisin” –napilla. Flow –luokkaan oli jäänyt väärä state machine tila, koska tätä error–tilaa ei oltu päivitetty uusiin lomakkeisiin. Tein neljä uutta error–tilaa mitkä ohjaavat nyt takaisin oikeisiin lomakkeisiin.

Perjantai 10.11

Alku

Jatkan tiedostonsiirto-ongelman parissa. Tavoitteena ymmärtää paremmin, miten meidän tiedostonsiirto toimii. Teen mahdollisesti ison asiakkaan korjauksia, jos saan tiedostonsiirron korjattua ja aikaa jää muuhunkin.

Loppu

Ensimmäinen vika, minkä löysin oli kirjoitusvirhe enää muuttujan nimessä mikä kirjoitettu vuosia sitten. Luulen että kirjoitusvirhettä ei ollut ikinä korjattu sen takia että se oli jo käytössä niin monessa eri projektissa, että sen muuttaminen olisi rikkonut kaiken. Testaaminen on ollut erittäin hidasta, koska en saanut lokaaliympäristössä tiedostonsiirtoa toimimaan, joten jouduin ajamaan kaikki testit Jenkinsin kautta. Tiedostonsiirrot ovat konfiguroitu meillä niin, että lokaaliympäristöä testatessa ne eivät normaalisti ole päällä, jotta tiedostopalvelinta ei spämmittäisi tukkoon.

Ongelma johtui kahdesta seikasta. Ensinnäkin tiedot serialisoitiin väärin kannassa itse sopimukseen, kun ne olisi pitänyt serialisoida suoraan PDF:n kautta, koska tiedostonimet muodostetaan niistä tiedoista mitkä ovat injektoitu PDF:ään. Toinen seikka oli se, että kun tiedostonsiirto-järjestelmä teki testiympäristöön uudet kansiot, kansioille ei automaattisesti annettu kaikki tarvittavia oikeuksia. Kirjauduin manuaalisesti serverille ja annoin kansiolle vapaammat oikeudet.

Viikkoanalyysi

Loppuviikosta syntyi pientä draamaa, koska projektipäällikkö oli antanut minulle tehtäväksi selvittää, minkä takia tiedostonsiirto ei toiminut, enkä saanut sitä selvitettyä ilman vanhemman kehittäjän apua. Vaikka olin saanut selkeät ohjeet, miten ongelma pitäisi selvittää ja tein kaiken ohjeiden mukaan, ongelmia tuli jatkuvasti vastaan enkä osannut selvittää mistä ongelma johtui. Kumpikaan senior kehittäjistä ei myöskään ollut kunnolla paikalla näiden kahden päivän aikana, joten kun tarvitsin neuvoa asiasta, heitä oli vaikea saada kiinni. Projektipäällikkö oli luvannut asiakkaalle, että ongelma korjattaisiin jo torstaina, näin ei kuitenkaan käynyt ja hän joutui kertomaan asiakkaalle, että ongelman korjaus viivästyy perjantaihin.

Perjantaina saimme lähes kokopäiväisen väännön jälkeen yhdessä senior kehittäjän kanssa homman korjattua, mutta projektipäällikkö oli jo ehtinyt ilmoittaa asiakkaalle, että homma venyisi ensi viikon maanantaihin. Panikoin itsekin siinä hieman, koska tuntui aluksi, että vika oli minun, koska en ollut saanut ongelmaa korjattua. Sitten muistin, että senior kehittäjäkään ei tiennyt mistä ongelma johtui, joten en pitänyt itseäni tähän kokonaan syyllisenä.

Ymmärrän kyllä, että välikätenä oleminen on välillä turhauttavaa, kun asiakkaalle pitää välittää pelkästään huonoja uutisia, eikä itse tiedä teknisistä seikoista mitään ja sitten pitäisi selittää heille, miksi asiaa ei saatu kuntoon. Tällaisia tilanteita tulisi välttää, koska ne ovat ikäviä niin projektipäälliköille kuin itse ohjelmoijille.

Yksi tapa olisi parantaa firman sisäistä kommunikaatiota, eli tunnistetaan ajoissa ongelman vaikeusaste ja näin saadaan keskitettyä siihen enemmän aikaa senior kehittäjiltä, eikä anneta projektipäällikölle tyhjiä lupauksia, jos ei olla varmoja, että ongelmat saadaan ratkottua siihen annetussa ajassa. En itsekään pitänyt siitä, että minulle annettiin tehtävät mitkä sain tehtyä, mutta ongelma ei silti ratkennut ja sitten pyörittelin peukaloita, koska senior kehittäjillä ei ollut aikaa auttaa asian kanssa.

Toinen asia mikä on vaivannut jo melkein koko sen ajan, kun olen ollut täällä töissä, on se, että tietokanta on valtava ja todella epäselvä, koska siitä ei ole valmista dokumentaatiota missään. Debugaaminen on välillä tosi hidasta, koska ongelman selvittämiseksi joudun sukeltamaan Java-olio-helvettiin ja kaivamaan sitä kautta tietokantataulut mitä kyseisessä prosessissa käytetään. Jos kannasta olisi esimerkiksi piirretty relaatiokaavio jossain, kaikki tekeminen nopeutuisi huomattavasti. Vaikka tietokantataulut ovat nimetty selkeästi, välillä ei ole hajuakaan minkä takia kyseinen taulu on olemassa. Olen maininnut asiasta, mutta se on niin aikaa vievä tehtävä, että sitä ei ole kukaan vielä saanut toteutettua.

Meillä siis tosiaan on tietokannan hallintaan kirjoitettu graafinen käyttöliittymä, mikä helpottaa paljon kannan kanssa työskentelyä, mutta se ei ole täydellinen koska se ei aina pysy perässä itse kannan kanssa, kun siihen tehdään muutoksia. Onneksi kun käyttöliittymän kautta tehdään kantaan muutoksia, ohjelma kirjaa ylös tehdyt muutokset mitkä näkyvät käyttöliittymässä suoraan MySQL -syntaksina. Nyt kun sain tämän tietooni, on asiat mennyt taas helpommin, koska voin hakea sitä kautta taulujen nimiä tekemällä ns. turhan muutoksen kantaan graafisen käyttöliittymän kautta ja saan näin sitä kautta selville, mitä taulua siinä käytetään.

3.5 Seurantaviikko 5

Maanantai 13.11

Alku

Aloitan uuden sprintin ison asiakkaan projektin kanssa. Suurin osa korjauksista liittyy lomakkeisiin ja sähköposteihin, elikkä muutetaan tekstejä, tyylejä ja kenttien paikkoja. Tänään tavoite saada vaan mahdollisimman paljon noita pikkukorjauksia tehtyä ja sen jälkeen kloonaan palvelun niin, että se toimii muissakin lomakkeissa. Kloonaaminen kannattaa tehdä tietenkin vasta sitten kun kaikki korjaukset on tehty, jotta korjaukset kloonaantuvat myös.

Loppu

Toteutin ison asiakkaan haluamia korjauksia heidän palveluunsa. Yrityksen sisäisen sopimuksen lomakkeelta piti poistaa muutama kenttä, mihin kirjattiin asiakkaan yrityksen nimi ja y-tunnus, koska sisäisessä prosessissa ne ovat aina vakiot. Piilotin kentät lomakkeelta ja laitoin niihin kannasta default-arvot. Etusivulta piti piilottaa linkkejä mitä asiakas ei tule ikinä käyttämään kuten: kirjaudu, rekisteröidy ja kirjaudu asiakkaana. Tämä

johtuu siitä, että yrityksen asiakkaat ohjautuvat palveluun heidän omien sivujen kautta, eivätkä asiakkaan Signom -sivulta, koska sitä tulevat käyttämään ainoastaan asiakkaan omat siihen tehtävään valtuutetut henkilöt.

Sähköposteista oli löytynyt vika, missä asiakkaalle generoituu sähköpostin sisältö kaksi kertaa, jos he hylkäävät sopimuksen. Tämä vika löytyi ja korjattiin nopeasti koska sähköpostien Velocity template engine -pohjaan oli jäänyt yksi looppi liikaa.

Tiistai 14.11

Alku

Sain suurimman osan pienistä korjauksista jo tehtyä, joten aloitan tänään kloonausprosessin. Kloonaan tietokantataulut, Flow Java-luokat, JSP-sivut ja kirjoitan testiluokat uusille lomakkeille.

Loppu

Aloitin kloonaamalla tietokantataulut, mikä on melko yksiselitteistä hommaa. Graafisessa tietokannanhallintatyökalussa on valmiina kloonaus-nappi, mikä ajaa insert-lausekkeella kantaan samat taulut ja arvot, mutta eri nimellä. Toistin prosessin neljä kertaa muuttamatta lomakkeen peruskenttiä, koska asiakas ei ole vielä toimittanut PDF-tiedostoja, mistä sitten myöhemmin näemme, miten lomakkeet tulevat eroavan toisistaan.

Flow-luokkaan lisäsin neljä uutta tilaa jokaiselle uudelle lomakkeelle. Logiikka tässä on lähes identtinen, ainoastaan sopimuksessa käytettävä PDF ja sopimuksen nimi muutetaan. Asiakas siirtyy omien sivujen kautta ensin helper-lomakkeelle, missä valitaan sopimuksen tyyppi. Flow-luokka ohjaa käyttäjän hänen valintansa mukaan sen jälkeen oikealle lomakkeelle.

JSP-sivujen kloonaus oli myös helppoa, koska kaikki vanhat kentät jäivät samanlaisiksi juurikin siitä syystä, että emme vielä tiedä miten ne tulevat muuttumaan. Muutin ainoastaan lomakkeiden otsikkojen nimet.

Testeissä kopioin jo valmiiksi kirjoitetun testin mikä ajoi prosessin läpi kahdella allekirjoittajalla. Loin jokaiselle lomakkeelle valmiiksi uuden sivu-olion, koska vaikka tiedän että lomakkeen kenttien arvot pysyvät vielä samoina, tulevat ne muuttumaan sitten kun saamme uudet PDF-tiedostot. Tämä helpottaa testien kirjoittamista myöhemmissä vaiheissa.

Keskiviikko 15.11

Alku

Jatkan kloonausprosessia. Lomakkeelle tarvitaan JavaScriptillä kirjoitettu koodipätkä, mikä täyttää yrityksen nimen automaattisesti y-tunnuksen perusteella. Tavoitteena opiskella lisää JavaScriptiä koska en ole siinä niin kokenut.

Loppu

Sain automaattisen täytön toimimaan mutta ongelmia tuli skandien kanssa. Koska käytämme melko vanhaa JSP-tekniikkaa, kaikki JavaScriptit syötetään sivuille JSP-sivuina eikä js-loppuisia tiedostoja siis laiteta koskaan suoraan JSP-tiedostoon. Ongelma olisi helposti ratkaistavissa, jos laittaisin koodit <script src=""> tägien sisään "charset='UTF-8"' -attribuutin kanssa, mutta JSP:n kanssa pitää keksiä nyt toinen tyyli. JSP-ongelma ratkesi pienen googlettelun jälkeen, lisäsin JSP-sivulle "contentType" attribuutin Java scriptlettien sisään. Scriptlettien käytöstä en ollut innoissani, koska se on niin vanhaa koodia, ettei sitä tulisi enää käyttää ollenkaan (StackOverflow 2010).

JavaScript autosyöttöhärvelin sain toimimaan ja opin samalla, miten JavaScriptillä kirjoitetaan key/value-array, mikä Javassa löytyy valmiiksi kirjoitettuna sen kirjastoissa.

Torstai 16.11

Alku

Viimeistelen tänään projektin ja suoritan päivän päätteeksi testiasennuksen, jotta projektipäällikkö pääsee testaamaan palvelua manuaalisesti niin, että voidaan todeta sen olevan kaiken puolin toimiva kokonaisuus huomista palautusta varten. Asiakas haluaa, että vain tietyt heidän urakoitsijat voivat solmia kyseisen sopimuksen, joten he toimittivat nyt pitkän Excel -tiedoston kaikista noin 70:stä urakoitsijasta joille halutaan tämä oikeus. Tavoitteena kerrata hieman MySQL-lauseita koska en jaksa yksi kerrallaan lisätä jokaista urakoitsijaa testikantaan.

Loppu

Kopion Excel –tiedostosta jokaisen urakoitsijan sähköpostiosoitteen ja y-tunnuksen tekstieditoriin. Siistin tiedoston niin, että jokainen osoite ja y-tunnus ovat omalla rivillään erotettuna pilkulla. Flow-luokkaan olen jo aikaisemmin kirjoittanut instruction-luokan, mikä tarkistaa onko urakoitsijalla oikeudet solmia sopimuksia. Luokassa ohjelma hakee kannasta taulun, minne nämä oikeudelliset urakoitsijat on lisätty. Kaikki taulun urakoitsijat loopataan läpi niin, että sähköpostiosoite ja y-tunnus jaetaan Javan String-luokan split-

metodilla mille annetaan parametriksi pilkku. Jos lomakkeella annettu sähköpostiosoite ja y-tunnus täsmäävät kannan arvojen kanssa, käyttäjällä on oikeus solmia sopimus.

Tämän jälkeen kirjoitin MySQL insert-lausekkeen valmiiksi ja ajoin sen testiympäristöön 70 kertaa muuttamalla aina nimen ja valuen arvot. Tähän olisi tietenkin voinut automatisoida jollain hienolla Python-scriptillä, mutta en jaksanut alkaa sitä tekemään, koska siinä olisi mennyt varmasti enemmän aikaa sillä en ole kirjoittanut Pythonilla mitään pitkään aikaan.

Perjantai 17.11

Alku

Ison asiakkaan viimeinen lomaketoteutus pitää olla yhtiön sisäinen ja se tehdään eri tavalla kuin aikaisemmat Flow-toteutukset. Sisäinen valtakirjan teko toimii niin, että yrityksen työntekijä jolla on oikeudet tehdä uusia valtakirjoja, kirjautuu palveluun sisään vahvalla tunnistuksella ja täyttää lomakkeen sitä kautta. Tavoite tänään opetella miten tämä prosessi tapahtuu koska en ole aikaisemmin sitä tehnyt.

Loppu

Yrityksen sisäinen valtakirja on todellakin siis meidän perusprosesseja, aivan ensimmäisiä palveluita joita tarjosimme yritysasiakkaille, kun firma perustettiin. Omalle kohdalle näitä ei vielä jostain kumman syystä ole kuitenkaan osunut. Prosessi oli melko helppo toteuttaa ja sain sen vaivatta toimimaan. Suurin ero oli siinä, että sisäisessä prosessissa sopimus pohjan tiedot haetaan kannasta, ennen kuin itse sopimus tehdään. Aikaisemmissa projekteissa kannassa määritetyt pohjan tiedot tulevat käyttöön vasta kun sopimus luodaan. Loppuilta meni yksikkötestien kirjoituksessa, missä meni hieman kauemmin koska en voinut vain kopioida vanhoja testejä suoraan, sillä prosessi kulki hieman eri kautta kuin aikaisemmissa toteutuksissa.

Viikkoanalyysi

Opin viikon aikana hieman lisää MySQL-juttuja, JavaScriptiä ja yrityksen sisäisen sopimuksen luontiprosessin.

Yrityksen sisäisessä prosessissa asiakas siis rekisteröityy palveluun hänen yrityksensä sähköpostiosoitteella. Sähköpostiin tulee linkki mitä painamalla asiakas ohjataan vahvaan tunnistautumiseen, missä hän todentaa, että hänellä on oikeus luoda asiakirjoja kyseisen yrityksen nimissä. Kun hän on kirjautuneena palveluun, hän painaa ”Uusi asiakirja” –

nappia, lataa sivulle valmiiksi kirjoitetun sopimuksen PDF –muodossa ja valitsee allekirjoitettavan sopimuksen osapuolet. Kun sopimus on luotu kaikki sen osapuolet saavat allekirjoituskutsun sähköpostiinsa. Kaikki aikaisemmat projektit mitä olen tehnyt ovat siis monimutkaisempia versioita tästä peruspalvelusta. Sain sisäisen valtakirjaprosessin tehtyä ilman kummempia ongelmia ja ihmettelin jälkeenpäin, miksei tämä ollut yksi ensimmäisistä asioista mitä meille täällä opetettiin.

Scriptlettejä ei tosiaan saisi enää käyttää ollenkaan, mutta koska käytin sitä vain ns. puukkona yhden pienen ongelman ratkaisemiseen, en ole siitä huolissani. Scriptlettejä ei normaalisti pitäisi ikinä käyttää JSP-sivuilla, koska JSP-sivun tehtävä on tiedon esittäminen ja vastaanottaminen, eikä sen tulisi käsitellä itse business-tason logiikkaa. Huonoja puolia scriptleteissä on monia. Suurimpia haittapuolia ovat esim. Uudelleenkäytettävyyteen, olio-ominaisuuksien puuttumiseen, kunnossapitoon ja testaukseen liittyvät ongelmat. Scriptlettejä ei voi käyttää uudelleen koska ne ovat aina sivukohtaisia, eikä olio-ohjelmoinnin hyviä puolia kuten periytymistä voida käyttää. Debugaillessa jos sivulle tulee virhe näet vain tyhjän sivun. Testaus on vaikeaa koska scriptlettejä ei voi yksikkötestata järkevästi. Kaiken tämän lisäksi, jos ohjelman business-tason logiikkaan tehdään muutoksia, tulisi kaikki scriptletit korjata yksi kerrallaan jokaiselta eri sivulta. Javassa JSP-sivuja käyttäessä scriptlettien käyttö korvataan taglibeillä (JSTL) ja EL:llä (Expression Language). Nykyään nämäkin tekniikat ovat jo muinaisia ja ne korvataan Template Engineillä kuten Thymeleaf tai Freemarker, mitä käytetään yhdessä Javan Spring -frameworkin kanssa.

3.6 Seurantaviikko 6

Maanantai 20.11

Alku

Aloitin päivän korjaamalla erään vian, missä asiakkaan nimi ei tulostunut oikein kun, hän oli allekirjoittanut sopimuksen. Ohjelma syöttää normaalisti allekirjoittajan nimeksi hänen etu- ja sukunimensä, mutta asiakas ei kyseisessä palvelussa syöttänyt allekirjoittajan nimeä ollenkaan koska allekirjoitus oli y-tunnus sidonnainen. Korjasin vian kirjoittamalla pienen if-ehdon, missä nimeksi laitetaan tässä tapauksessa yrityksen nimi. Viankorjauksen jälkeen sain tehtäväksi korjata toisen asiakkaan lomaketta, mikä on jo tuotannossa. Sain asiakkaalta Word-tiedoston missä oli kaikki heidän haluamansa muutokset ja ne näyttävät olevan suurimmaksi osaksi tyylikorjauksia. Tavoitteena tänään kerrata CSS-tyylejä.

Loppu

Sain suurimman osan korjauksista tehtyä päivän päätteeksi. Huomasin että kuukausi sitten heille oli jo tehty tyylimuutoksia ja huomasin nyt, että suurin osa tehtävistä oli vain näitten tyylien korjailua. Ihmetytti hieman koska tekemäni muutokset tuntuivat vain poistavan aikaisemmat "korjaukset".

Selvisi jälkeenpäin, kun olin tehnyt kaikki korjaukset Googlen Chrome-selaimella, että Firefoxilla kenttien tekstit eivät tule kokonaan näkyviin. Tämä selittää aikaisemmat tyylikorjaukset missä kaikki kentät olivat isompia kuin normaalisti.

Tiistai 21.11

Alku

Jatkan korjausten tekemistä ja keskityn tänään enemmän JQueryyn. Asiakkaan lomakkeella on kymmeniä dropdown-valikoita ja checkboxeja, jotka pitää checkata/uncheckata riippuen näistä valinnoista. Tavoitteena tänään kerrata JQuerya ja JavaScriptiä.

Loppu

Korjasin tyylit Firefoxille laittamalla pelkästään ”padding-left” attribuutin CSS-tyyleihin, mikä jotenkin kummasti korjasi ongelman missä tekstistä näkyi vain yläosa tekstikentissä.

Keskiviikko 22.11

Alku

Olen tänään asiakaspalveluvuorossa eli käyn läpi asiakkaiden tukipyyntöjä. Tavoitteena opetella ongelmien selvitystä, käyttäen kaikkia meille annettuja työkaluja.

Loppu

Ensimmäisessä tukipyynnössä asiakas ihmetteli, miksei pystynyt yrityksen nimissä myöntämään oikeuksia toiselle työntekijälle. Tein haun kantaan ja huomasin, että kyseisen yrityksen y-tunnusta ei löytynyt sallittavien listalta. Lisäsin y-tunnuksen listaan ja kuittasin ongelman selvitetyksi. Toisessa tukipyynnössä asiakas oli luonut heidän rajapintansa kautta sopimuksen, missä vain itse sopimuksen tekijä oli allekirjoittavana osapuolena, mutta ei päässyt tätä allekirjoittamaan. Ongelman selvittelyssä meni oma aikansa, mutta se selvisi kuitenkin lopuksi. Kuittasin asiakkaalle, että ongelma löydetty ja mistä se johtui.

Torstai 23.11

Alku

Aloitan päivän jatkamalla tukipyyntöjä. Asiakas haluaa tekstimuutoksia ja neljä uutta sopimuspohjaa mihin he ovat toimittaneet uudet PDF-tiedostot.

Loppu

Poistin kolme vanhaa sopimuspohjaa ja laitoin uudet neljä tilalle. Migratoin uudet pohjat testiympäristöön uusien PDF-tiedostojen kanssa. Kokeilin että kaikki toimii niin kuin pitääkin testiympäristössä ja kuittasin homman tehdyksi. Sain sen jälkeen projektipäälliköltä uuden tehtävän kirjoittaa eräälle asiakkaalle peruspalvelun käyttöohjeet englanniksi. Kirjoitan ohjeet Wordilla mihin lisään ruutukaappauksia peruspalvelun pystyttämisen eri vaiheista.

Perjantai 24.11

Alku

Jatkan ohjeiden kirjoittamista. Tavoitteena saada vain ohjeet tehtyä loppuun. Samalla hyvä kerrata ja käydä yksityiskohtaisesti läpi meidän peruspalvelua sekä sen englanniksi kääntämistä.

Loppu

Kirjoitin ohjeet Wordillä, missä selitin selkeästi jokaisen peruspalvelun vaiheen kuvankaappausten kera. Tähän sisältyy mm. Rekisteröityminen palveluun, asemavaltuuden vastaanottaminen, yrityksen asiakirjat näkymä, omat asiakirjat näkymä, asiakirjan luominen ja lähettäminen allekirjoitettavaksi, allekirjoitusjärjestyksen asettaminen ja asiakirjan allekirjoittaminen hylkääminen.

Viikkoanalyysi

En ollut aikaisemmin käynyt läpi jokaista peruspalvelun asiaa näin yksityiskohtaisesti, joten opin samalla uusia asioita itsekin. Joidenkin lakitermien käännöksissä meni aikaa, koska en tiedä suomen kielelläkään mitä ne kaikki tarkoittavat. Tuli mieleen koulussa käyty kurssi mikä käsitteli lakia, eikä itseäni silloin kiinnostanut kurssin asiat juuri lainkaan. Ajattelin, että jos minusta tulee ohjelmoija niin enhän minä näillä tiedoilla mitään tee. Olisi pitänyt olla paremmin hereillä tunneilla, sillä osa täällä vastaantulevista ongelmista ei liity oikeastaan koodiin, vaan sopimusoikeuksiin.

Keskiviikkona oli varsin kiperä ongelma missä asiakas ei päässyt allekirjoittamaan itse luotua sopimustaan. Yritin ensin selvittää, mitä kautta sopimus on luotu. Kyseessä oli niin vanha projekti, että siitä ei löytynyt edes dokumentaatiota. Tein kantahakuja ja huomasin, että kyseinen henkilö on yrittänyt luoda sopimusta sellaisen yrityksen tiedoilla, mitä ei meidän palvelussa ole rekisteröity kantaan ollenkaan. Tein kaupparekisteriin haun kyseisen henkilön henkilötunnuksella ja sain selville, että hänellä ei ole nimenkirjoitusoikeutta kyseiseen yritykseen. Kaupparekisterin tiedot eivät siis olleet ajan tasalla, eikä vika johtunut meidän palvelusta. Tämä oli taas sellainen ongelma mitä en mitenkään olisi osannut selvittää ilman vanhemman työntekijän apua. Lakia en kyllä ala opiskella, eikä varmasti tarvitsekaan. Luulen että nämä asiat selkenevät ajan myötä.

Kun kehitystiimissä on monta eri kehittäjää, on erittäin suositeltavaa – jos ei jopa pakollista – käyttää versionhallintatyökaluja. Versionhallinnalla varmistetaan se, että yhdenkään kehittäjän työ ei mene hukkaan, kun jos vaikka kaksi henkilöä muokkaa samaan aikaan samaa tiedostoa. Git on eräs suosituimmista versionhallintatyökaluista ja sen käyttö on usein tärkeä osa kehitysprosessia. Git pitää kirjaa kaikista projektin muutoksista. Jos jokin uusi pala koodia sattuisi vaikka rikkomaan koko ohjelman, Gitin avulla voidaan palata ajassa taakse päin tilanteeseen, missä ohjelma vielä toimi. (Cygnis Media Editor 2015)

Yksi hyvä käytänne versionhallinnassa on pitää yksi pääkehityshaara, joka haarautetaan omiin pienempiin ominaisuus-haaroihin. Omassa pienessä haarassa ohjelmaan voi tehdä huoletta muutoksia, mitkä eivät riko pääkehityshaaraa. Kun ominaisuus on valmis ja testattu toimivaksi, päähaaraan tulleet mahdolliset muutokset voidaan yhdistää ominaisuus-haaraan. Jos kaikki toimii vieläkin niin kuin pitää, ominaisuus-haaran voi lopuksi yhdistää takaisin pääkehityshaaraan. Tällä tavalla minimoidaan virheiden esiintyminen kehityshaarassa.

Pääkehityshaaraa pidetään jatkuvasti silmällä automaatiotestityökaluilla. Automaatiotestauksella tarkoitetaan palvelinta, mikä ajaa projektin testejä kellon ympäri läpi. Näin jos jokin uusi ominaisuus esimerkiksi rikkoo osan ohjelmasta, näkee sen heti automaatiotesterin lokeista. Ennen tuotantoasennusta pääkehityshaara kannattaa jäädyttää omaksi haarakseen, minne ei enää viedä uusia muutoksia. Jäädytetyn haaran testit ajetaan läpi ja varmistetaan, että ongelmia ei löydy. Kun kaikki testit menevät läpi, haaran voi turvallisesti asentaa tuotantoon. Tällä prosessilla yritetään minimoida tuotantoon asti joutuvat viat.

Aiemmista syistä johtuen, yksikkötestien kirjoitus on tärkeä osa kehitysprosessia. Ennen kuin uuden ominaisuuden ohjelmointi alkaa, on hyvä dokumentoida ylös, minkälaiset testit sille tulee kirjoittaa. Testeillä pitää miettiä järkevät vaatimukset, joissa otetaan huomioon kaikki mahdolliset loppukäyttäjän syöttämät muuttujat. Kolodiy (2014) kertoo artikkelissaan hyvän yksikkötestin pääkohdat ja ne ovat: helposti kirjoitettavuus, luettavuus, luotettavuus ja nopeus. Testit tulevat olla helposti kirjoitettavissa, koska niitä kirjoitetaan niin paljon. Helposti luettavaa koodia on helpompi ylläpitää ja se helpottaa huomattavasti kun etsitään vikoja. Hitaita testejä ajaminen vie aikaa ja saattaa vähentää testausta kokonaisesti, koska kehittäjä ei jaksa ajaa hitaita testejä. Vasta kun testit menevät kaikki läpi, on ominaisuus valmis yhdistettäväksi kehitys- tai testihaaraan.

3.7 Seurantaviikko 7

Maanantai 27.11

Alku

Teen asiakkaan lomakkeeseen korjauksia. Liitetiedostojen nimet täytyy muuttaa, numerokentät validoitava niin, että voi syöttää vain numeroita ja desimaalit erotetaan pilkulla eikä pisteellä. Tavoitteena saada kaikki korjaukset tehtyä ja asentaa muutokset testiympäristöön.

Loppu

Tein ensin tarvittavat muutokset tietokantaan tiedostojen nimien osalta. Tämän jälkeen kirjoitin ensin validaatiosäännöt puhtaaksi ja kokeilin niitä omassa Java-ohjelmassa, ennen kuin syötin ne tietokantaan. Loppupäivän tein vielä satunnaisia teksti- ja lomakemuutoksia. Lopuksi puskin koodit sisään ja migratoin kamat testiympäristöön. Kuittasin vielä asiakkaalle, että korjaukset ovat nyt tehty ja että he pääsevät nyt testaamaan uutta lomaketta testiympäristössä.

Tiistai 28.11

Alku

Olen tänään asiakaspalveluvuorossa. Sain ensin tehtäväksi selvittää miksi ison asiakkaan PDF-tiedostoihin ei tulostu allekirjoitussivua. Toisen asiakkaan tukipyynnössä pyydetään selvittämään miksi tiedostonsiirrot eivät ole menneet läpi tuotannossa. Tavoitteena parantaa ongelmanselvitystaitoja ja oppia lisää meidän palvelun sisäisistä toiminnoista.

Loppu

Selvitin ensin tiedostonsiirto-ongelmaa hakemalla kannasta tiedot taulusta mistä näkee milloin tiedostot ovat lähteneet ja ovatko ne menneet perille asti. Kaikki näyttää meidän päässä toimivan niin kuin pitääkin, tarkistin vielä tuotannon logeista, että kaikki ok ja kuittasin asiakkaalle, että vika on heidän päässään. Loppupäivän selvitin PDF ongelmaa.

Keskiviikko 29.11

Alku

Jatkan asiakaspalvelussa ongelmien selvittämistä. Asiakas ei pääse luomaan sopimusta, vaikka hänellä on prokuraattorin oikeudet. Selvitetään mistä tämä johtuu. Tavoitteena tänään parantaa ongelmanselvitystaitoja.

Loppu

Selvittelin lähes koko päivän aikaisemmin mainittua ongelmaa. Kun sain sen selvitettyä jälkeen aloitin toisen tukipyynnön läpikäymisen vanhemman kehittäjän kanssa.

Asiakkaalla on palvelussa virhe, jos he täyttävät virheellisen sähköpostiosoitteen. Normaalisti kun sähköposti bouncaa, siitä lähtee sopimuksen luojalle viesti, että osoite on virheellinen ja että hänen täytyisi kirjautua palveluun ja korjata tämä osoite. Kenelläkään sopimuksen osapuolista ei kuitenkaan tällä hetkellä ole oikeuksia muokata toisten osapuolien osoitteita. Sain tehtäväksi kirjoittaa uuden Java -testiluokan millä pääsee toistamaan virheen, minkä jälkeen muokataan kanta-asetukset niin, että sopimuksen luoja voi muokata toisten sähköpostiosoitteita. Tässä pitää kuitenkin vielä tarkistaa, että hän ei voi muokata kaikkia yrityksen sähköpostiosoitteita.

Torstai 30.11

Alku

Jatkan edellisen päivän bounced-sähköposti-tapausta, ja teen päivän mittaan tulevia asiakaspalvelu tikettejä. Jos tukipyyntöjä ei tule ja aikaa jää muuhun, saan projektipäälliköltä lisää korjauksia ison asiakkaan projektiin. Tavoitteena parantaa taas ongelmanselvitystaitoja.

Loppu

Sain bounced-sähköposti ongelman ratkaistua, minkä jälkeen aloitin uuden tukipyynnön selvittämisen, missä asiakas ihmettelee, että kun palvelussa vaihtaa allekirjoittavan

osapuolen sähköpostiosoitteen, niin hänelle lähtee vanhanmallinen sähköposti, missä on default-sähköpostipohja. Käytämme sähköposteihin Apachen Velocity -template engineä.

Sain pienen debugaus-session jälkeen kiinni ongelman, mikä johtui siitä, että sähköpostiohjelma lähettää viestit erilaisella event-tyypillä, jos sähköposti vaihdetaan. Kun tälle eventille ei oltu tehty muokattua pohjaa, se käytti automaattisesti default-pohjaa. Korjasin ongelman lisäämällä tauluun konfiguraatiot niin, että sähköpostinvaihto -eventistä lähtee sama pohja, kuin sopimuksenluonti -eventistä.

Perjantai 1.12

Alku

Jatkan vielä edellisen päivän tukipyyntöä. Asiakkaalla oli monta erilaista sopimustyyppiä, joten kirjoitin uuden testiluokan, mikä varmistaa, että sähköpostit lähtevät oikein. Migrasimme testiserverin Google Cloudiin eilen illalla, joten testiasennus pitää hoitaa tänään uudella tyylillä. Asiakkaan kanssa on Skype-palaveri iltapäivällä missä projektipäällikkö hoitaa puhumisen ja olen itse mukana antamassa teknistä tukea. Tavoitteena tänään opetella uuden testiympäristön käyttö.

Loppu

Valmistelin iltapäivällä testiasennusta. Yhdistin muutaman haaran Gitistä testi-haaraan mistä testiympäristö rakennetaan. Pullasin myös uuden migraattori-työkalun sen uudesta omasta repositorysta ja muokkasin siihen uuden testiserverin asetukset. Tässä vaiheessa oli jo hieman kiire, koska tunnin päästä pitäisi käydä asiakkaan kanssa Skype-palaverissa uusia korjauksia läpi. Sain testi-haaran rakennettua, ja tietokanta-migraatiot menivät myös sisään, mutta kun otin SSH-yhteyden uudelle serverille ja ajoin sitä kautta Googlen pilveen serverin päivityskomentoja, tuli Permission Denied -virheitä. Otin yhteyden meidän serverivastaavaan ja ilmeni, että sieltä oli unohdettu antaa ajettaville tiedostoille ajo-oikeuksia. Virheet korjattiin ja sain viimein uuden testiasennuksen tehtyä. Kävimme asiakkaan kanssa Skype-palaverissa palvelun muutokset läpi. He pitivät uudesta ulkoasusta ja kaikki toimi niin kuin pitikin.

Viikkoanalyysi

Tiistaina asiakkaan PDF ongelma selvisi, kun tarkastelin heidän allekirjoittajien kanta-asetuksia. Heillä oli validoijan oikeudet, mitkä ylikirjoittivat normaalin allekirjoittajan säännöt. Korjasin konfiguraatiot, mitkä taas vaikuttivat sisäänkirjautumiseen, sillä vahvaa tunnistautumista ei enää tarvittu. Tämä tietenkin rikkoi kaikki automaatiotestit ja ne piti

kirjoittaa uudestaan. Löysin myös kannasta taulun mikä määrittää asiakkaan evästeasetukset ja ajoin siihen insert-lausekkeen mikä disabloi sen niin, että asiakkaan (testirobotin tässä tapauksessa) ei tarvitse tunnistautua vahvasti aina sen jälkeen, kun on kirjautunut sisään sähköpostilla ja salasanalla. Tämä helpotti testaamista huomattavasti.

Keskiviikon ongelma selvisi taas kantahakujen kautta. Huomasin että tuotannossa käyttäjällä on kolme päällekkäistä oikeutta, mistä vain yksi on varmennettu. Tarkastelin konepellin alta koodeja ja päättelin, että ohjelma ei osaa hakea monia päällekkäisiä oikeuksia ja täten löytää ensimmäiset arvot kannasta, mitkä eivät ole todennettu. Tämä oli kuitenkin ihan ok, kun varmistin asiaa senior kehittäjältä. Tein vielä lokaalikantaan debugtunnukset yrityksen kyseiselle y-tunnukselle ja yritin toistaa ongelmaa lokaalisti, mutta kaikki toimi niin kuin pitikin. Laitoin asiakkaalle sähköpostilla pyynnön, missä kysyin kuvankaappausta sivusta, kun hän on kirjautunut sisään. Tämän jälkeen asiakkaan toinen työntekijä luuli tietävänsä mistä ongelmassa on kyse ja neuvoi toista käyttäjää väärillä ohjeilla. Lähetin saman tien viestiä takaisin missä ilmaisin, että käyttäjä ei tosiaan ole pääkäyttäjä roolissa, vaikka toinen asiakas näin väitti, ja että prokuristin roolilla pitäisi pystyä luomaan uusia sopimuksia. Perään laitoin vielä erittäin selkeät ohjeet siitä, miten ja mistä tämä prosessi tehdään.

Torstaina korjasin bounced-sähköpostiviesti ongelman. Meillä oli valmiiksi kirjoitettu testaus metodi, mikä merkitsi sähköpostin bouncediksi, mutta se lisäsi vain dataa kannan tauluun, mikä hoiti infoviestejä, missä kerrottaan asiakkaalle, että sähköposti on bouncannut. Java-luokka mikä määrittelee käyttäjälle sallittuja toimintoja, haki bounced-sähköpostit eri taulusta. Kirjoitin uuden metodin, mikä päivittää sähköpostin bounced -tilaan oikeaan tauluun, ja laitoin sen ajamaan aina ensimmäisenä, kun testatessa halutaan merkata sähköposti bouncediksi. Bounced-sähköpostien lista käytiin alun perin läpi suoraan JSP-sivulla, mikä ei ole järkevää, vaan tarkistus pitäisi tehdä samassa Java-luokassa, missä muut oikeudet määritetään.

Korjasin Java-luokan niin, että bounced-sähköpostien hallinta sallitaan vain ja ainoastaan, kun käyttäjällä on oikeudet hallita toisten käyttäjien sähköpostiosoitteita. Tämä sen takia, että asiakkaalle tulisi ilmoittaa bounced-viesteistä vain silloin, kun hän voi itse tehdä asialle jotain, eli korjata virheellisen sähköpostin niin, että sopimus saadaan allekirjoitettua loppuun.

3.8 Seurantaviikko 8

LOMALLA KOKO VIIKON

3.9 Seurantaviikko 9

Maanantai 11.12

Alku

Jatkan korjausten tekemistä isolle asiakkaalle. Asiakkaan urakoitsijat esitäyttävät lomakkeen, minkä jälkeen asiakas saa sähköpostiin kutsun tulla tarkastamaan lomakkeen tiedot ja valitsemaan allekirjoittavat osapuolet. Tälle esitäytetylle lomakkeelle pitää laittaa IP-rajoite, mikä estää muiden kuin heidän työntekijöiden pääsyn lomakkeelle. Etusivulta pitää myös piilottaa muutama teksti ja tiedostonsiirtoja varten pitää määritellä datatransformaatiot. Tiedostojen nimeen tulee injektoida aikaleima ja urakoitsijan nimi. Tavoitteena saada nämä korjaukset tehtyä ja valmistella koko korjattu projekti testiympäristöä varten.

Loppu

Tein koko päivän korjauksia ja tein yhden selvityspyynnön asiakkaalle, joka ei päässyt allekirjoittamaan sopimusta mikä oli jo tuotannossa. Vika oli täysin käyttäjässä joka oli ymmärtänyt palvelun tarkoituksen väärin ja vastasin hänelle valmiiksi generoidulla vastauksella. Päivän lopuksi kun sain kaikki ison asiakkaan korjaukset toimimaan, suoritin testiasennuksen ja annoin projektipäällikölle uusien kenttien arvot, jotta hän voi korjata ne uusiin PDF-tiedostoihin.

Tiistai 12.12

Alku

Olen tänään asiakaspalveluvuorossa eli teen sekalaisia ongelman selvityksiä asiakkaille. Iltapäivälle varattu myös yhden vanhemman projektin katselmointi, joten laitan sen testiluokat pyörimään ja otan nauhalle mitä siinä tapahtuu. Pitää myös lukaista dokumentaatio kerran läpi, koska en muista kyseisestä projektista juuri mitään. Lisäksi saan ison asiakkaan projektiin tägitetyt PDF-tiedostot, mitkä pitää testata ja asentaa testiympäristöön. Tavoite tänään parantaa ongelmanselvitystaitoja.

Loppu

Ensimmäisessä tukipyynnössä loppuasiakas valitteli sekavista sivuista ja kyseli, että mistä hän näkee lainan määrän. Vastasin ystävällisesti takaisin, että me ylläpidämme ainoastaan kyseisen asiakkaan sähköistä allekirjoituspalvelua ja välitin hänelle oikean asiakaspalvelun numeron.

Vanhassa projektissa huomattiin looginen vika, missä eräs linkki mikä ohjaa käyttäjän takaisin asiakkaan omaan palveluun ilmestyi allekirjoituksen jälkeen sivulle, jos asiakas allekirjoitti jaetulla oikeudella, eikä sopimusta siis oltu vielä suljettu. Laitoin JSP-sivulle if-ehdon mikä tarkistaa sopimuksen tilan ja sen perusteella piilottaa kyseisen linkin sivulta. Loppupäivästä sain uudet PDF-tiedostot mitkä asensin lokaalisti palveluun. Tämän jälkeen ajoin kaikki testit läpi ja tarkistin että tiedot lomakkeilta siirtyivät oikein PDF-tiedostoihin. Otin kuvankaappaukset PDF-tiedostoista ja kävimme ne vielä varmuuden vuoksi projektipäällikön kanssa läpi ennen kuin suoritin asennuksen testiympäristöön.

Keskiviikko 13.12

Alku

Jatkan asiakaspalvelun päivystämistä koko päivän eli teen tukipyyntöjä ja selvityksiä sitä mukaan, kun niitä tulee. Erään asiakkaan palvelu pitäisi myös migratoida uuteen testausympäristöön, mikä sitten asennetaan tuotantoon ensi viikolla, jos testit menevät läpi. Selvitän kiireellisimmät tukipyynnöt ja migratoin palvelun, jos siihen jää aikaa.

Loppu

Ensimmäinen tukipyyntö oli lyhyt ja ytimekäs viestillä: "Herjaa käyttäjätunnusta". Tarkistin ensin, minkä asiakkaan palvelu oli kyseessä. Sen jälkeen aloin selvittää, miten kyseinen palvelu toimii. Avasin ensin meidän sisäisestä wikistä dokumentaation ja luin sen läpi, mistä selvisi, että palvelussa loppuasiakkaalta ei edes vaadita palveluun sisään kirjautumista. Tuotannon logeja tarkastellessa selvisi myös, että loppuasiakas oli vain päättänyt haluta kirjautua palveluun, vaikka tätä ei missään kohdassa pyydetty. Vastasin loppuasiakkaalle, joka oli täyttänyt sivuilla hakemuksen, että hakemus oli lähtenyt onnistuneesti eteenpäin, eikä hänen tarvitse tehdä muita jatkotoimenpiteitä. Loppupäivä meni toisen palvelun migratoinnissa.

Torstai 14.12

Alku

Tänään pitäisi migratoida ison asiakkaan palvelu integraatiotesti-haaraan. Tämä haara sitten jäädytetään illalla ja siitä branchataan uusi haara mikä menee näillä näkymin ensi maanantaina tuotantoon. Saan vielä muutamat korjaukset palveluun projektipäälliköltä, jonka jälkeen aloitan migratoinnin.

Loppu

Asiakas halusi vaihtaa sähköpostiosoitteen mihin valtakirjahakemukset lähetetään, koska huomasivat, että heillä oli vanha osoite jo käytössä toisessa palvelussa. Tein tarvittavat korjaukset tietokantaan ja backend-koodiin ja vaihdoin osoitteet sähköpostien footereista. Asiakas oli toimittanut myös uuden listan käyttäjistä, jotka voivat hakea sopimusta. Päivitin uudet käyttäjät dokumentaatioon, mutta en ajanut niitä vielä mihinkään kantaan, vaan odotan että palvelu asennetaan tuotantoon ja ajan ne sitten suoraan sinne.

Korjausten jälkeen muokkasin kaikki peruspalvelun SQL-lauseet uuteen ympäristöön sopiviksi ja ajoin ne integraatiokantaan. Monia viikkoja kestänyt iso projekti oli nyt loppusuoralla, odottamassa tuotantoasennusta. Tuntui kuin olisin nyt ensimmäistä kertaa päästänyt oman lapsen ovesta ulos karuun maailmaan.

Perjantai 15.12

Alku

Tulin töihin ja katselin Jenkinsin lokeista, että kaikki testit ovat epäonnistuneet ison asiakkaan projektissa. Nyt pitää selvittää mistä tämä johtuu. Tavoite tänään saada testit onnistumaan tuotantoasennusta varten. Loppupäivän autan projektipäällikköä asiakastuen teknisissä asioissa.

Loppu

Huomasin lokeista, että testit kaatuivat, kun käyttäjä yritti sähköpostilinkin kautta kirjautua palveluun. Alustin lokaalikannan integraatiohaaran mukaan ja ajoin testin, mistä huomasin, että eräs eväste-asetus oli unohtunut exporttaa päivän päätteeksi ja vika johtui siitä. Exporttasin asetuksen ja testit menivät läpi.

Tämän jälkeen sain tukipyynnön missä asiakas kyseli, onko tiedostonsiirto mennyt läpi vai onko hän vahingossa poistanut tiedoston heidän palvelimeltaan. Hain sopimuksen tunnuksella tuotantokannasta sen tiedostonsiirto lokit ja selvisi että, tiedosto oli siirretty heidän palvelimelle. Kuittasin projektipäällikölle, että tiedosto oli siirretty ja ajankohdan, milloin
tämä oli tapahtunut. Toisessa tukipyynnössä asiakas halusi tarkistaa, että hänen sopimuksensa oli mennyt testiympäristössä läpi ja että se oli allekirjoitettu. Tein tarvittavat tietokantahaut ja kuittasin, että oli mennyt läpi ja ilmoitin ajankohdan, milloin sopimus oli allekirjoitettu.

Sain myös projektipäälliköltä selvityspyynnön, missä ihmeteltiin miksi yrityksen toimitusjohtaja ei voinut allekirjoittaa sopimusta muussa kuin henkilökohtaisessa roolissa.

Tietokantahausta selvisi, että vaikka hänellä oli tässä yrityksessä toimitusjohtajan asema, ei hänellä ollut kaupparekisterissä oikeuksia allekirjoittaa yrityksen nimissä.

Iltapäivällä päivittelin ison asiakkaan dokumentaatiota valmiiksi tuotantoasennusta varten. Listasin ensin mitkä migraatiot menevät automaatio-scripteillä ja mitkä pitää tehdä manuaalisesti. Liputin myös listan loppukäyttäjistä, jotka saavat solmia sopimuksia asiakkaan kanssa ja että ne pitää ajaa suoraan tuotantokantaan, kun kaikki muut asiat on saatu tehtyä.

Viikkoanalyysi

Maanantaina kun tein asiakkaan palveluun korjauksia, laitoin IP-rajoitteet meille valmiiksi tehtyyn tietokanta tauluun, mihin lisäsin heidän IP-osoitteen lisäksi myös localhostin ja meidän toimiston, mikä helpottaa testaamista. Etusivun kentät piilotin lisäämällä css-tiedostoon muutamalle diville "display:none" attribuutin. Näiden korjausten jälkeen kirjoitin uuden metodin Java-luokkaan mikä hoitaa datatransformaatiot. Tiedoston nimessä ei tulisi käyttää skandeja, joten uusi metodi muuttaa urakoitsijan yrityksen nimestä skandit normaaleiksi aakkosiksi ja poistaa samalla kaikki muutkin merkit mitkä eivät kuulu ASCII-merkistöön. Muokkasin hieman vanhaa testiluokkaa niin, että siinä käytettiin urakoitsijaa, jonka yrityksen nimessä käytetään skandeja ja ajoin testin siihen pisteeseen, että sopimus oli luotu. Tarkistin lokaalikannasta, että uusi arvo mikä injektoidaan tiedoston nimeen, on nyt kannassa ilman skandeja. Transformaatio ei aluksi toiminut, koska olin kopioinut Slackistä osan koodista ja Slack muuttaa automaattisesti normaalit lainausmerkit double comma quotation markeiksi, mitkä eivät sitten käänny koodissa. Ulkonäöltään nämä merkit ovat lähes identtiset. Tästä syystä Slackissa tulisi aina koodit lähettää Code snippet -moodissa.

Torstaina kun migratoin tietokantojen dataa tein sen valmiiksi kirjoitetuilla "migraatioloitsuilla", mitkä on kirjoitettu Pythonilla. Nämä scriptit vertaavat ja siirtävät dataa yhdestä kannasta toiseen kantaan, mikä helpottaa huomattavasti migratointia eri ympäristöjen välillä. Olin kirjoittanut valmiiksi kaikki scriptit kyseisen asiakkaan dokumentaatioon, mistä sitten vain kopioin ja ajoin ne muuttamalla pari parametriä, mitkä määrittelivät, mistä kannasta mihin kantaan tiedot halutaan siirtää.

Kun olin saanut kaikki tietokanta-asetukset siirrettyä, yhdistin asiakkaan oman haaran integraatiohaaraan ja rukoilin että ei tulisi konflikteja. Muutama pieni konflikti tuli, mutta sain ne korjattua. Konfliktitilanteet ovat ikäviä, koska niitä ei pysty aina yksin selvittämään, vaan avuksi tarvitaan toista kehittäjää, joka tietää enemmän koodeista, missä konflikti on

tullut. Vähänkin isommat projektit tehdään aina omaan git-ominaisuus-haaraansa, mikä alustetaan kehitys-haarasta. Kun ominaisuus-haara yhdistetään esimerkiksi testi-haaraan, on tärkeää, että kehitys-haara yhdistetään taas ominaisuus-haaraan, ennen testi-haaraan viemistä. Näin voidaan ehkäistä konfliktitilanteita, koska jos ominaisuus viedään testiin, on koodeja voitu sillä aikaa muokata kehityshaarassa, mikä taas on viety aikaisemmin testiin.

3.10 Seurantaviikko 10

Maanantai 18.12

Alku

Tänään pitää valmistella tuotantoasennukset kahdelle eri asiakkaan uudelle palvelulle. Päivään sisältyy molempien asiakkaiden dokumentaation huolella läpikäyminen ja kaikki tärkeät tuotantoasennukseen liittyvät seikat pitää liputtaa ylös. Tavoite tänään saada kaikki nämä tärkeät asiat tehtyä, jotta huomenna kaikki toimii tuotannossa niin kuin pitääkin.

Loppu

Heti aamusta huomasin, että muutama itse kirjoittama yksikkötesti oli epäonnistunut jäädytetyssä git-haarassa, mistä on tarkoitus tehdä tuotantoasennus tänään illalla. Alustin ympäristön haaran mukaan ja ajoin kyseiset testit lokaalisti. Testit kaatuivat sopimuksen luontivaiheessa, ja Tomcatin logeista selvisi, että tietokannasta puuttui tiedot PDF-tiedostoille. Olin siis unohtanut migratoida edellisenä päivänä erään taulun, mitä vain yrityksen sisäinen prosessi käyttää. Korjailin tätä loppupäivän ja lopuksi päivitin uudet PDF-tiedostot.

Tiistai 19.12

Alku

Sain projektipäälliköltä lisää korjauspyyntöjä ison asiakkaan palveluun. Näihin korjauksiin on varattu nyt kaksi päivää aikaa. Tavoitteena saada tänään mahdollisimman paljon korjauksia tehtyä.

Loppu

Ensimmäiset korjaukset olivat melko yksiselitteisiä kirjoitusvirheitä. Korjasin virheet suoraan properties-tiedostoon mistä tekstit tulevat. Sähköpostiin haluttiin yksi teksti paragraaffi lisää, joten lisäsin sen Velocity enginen pohjaan mitä kautta sähköpostit generoituvat. Asiakas halusi myös, että kun he kopioivat ja liittävät y-tunnuksia

lomakkeelle, tulisi niistä poistaa automaattisesti turhat välit koska niiden kanssa yrityksen automaattinen syöttö ei toimi. Lisäsin JavaScriptiin jQueryn trim-metodin mikä poistaa nämä turhat välit, ennen kuin se hakee yrityksen nimen y-tunnuksen perusteella. Yritysten nimet haluttiin myös genetiivimuotoon, jotta ne näyttäisivät paremmalta PDF-tiedostossa mihin ne injektoidaan. Aloitin yritysten nimien perusmuodoista omistusmuotoihin kääntämisen, kunnes muutaman jälkeen tajusin, että kaikki nimet päättyvät "Oy", joten katenoin jokaisen yrityksen nimen jälkeen stringin ":n" sen perään. Tämä nopeutti hommaa huomattavasti.

Keskiviikko 20.12

Alku

Jatkan tänään edellisenä päivänä saatuja korjauspyyntöjen tekemistä. Urakoitsijoiden lista pitää vaihtaa kannasta toiseen paikkaan, jotta ne olisi järkevämpi migratoida toisiin ympäristöihin jatkossa. Nimikentät henkilöiden ja yrityksen kohdalla tulee lomakkeilla muuttaa niin että ne vastaavat asiakkaan toiveita paremmin. Tavoitteena käydä asiakkaan kanssa läpi muutokset ja toteuttaa ne järkevästi.

Loppu

Olin asiakkaan kanssa palaverissa projektipäällikön kanssa, missä kävimme läpi lomakkeiden muutoksia. Asiakas ei ollut tyytyväinen siihen, miten nimet generoituivat verkkolomakkeilta heidän toimittamiin PDF-tiedostoihin. Kävimme tätä aika pitkään heidän kanssaan läpi ennen kuin pääsimme yhteisymmärrykseen asiasta.

Loppupäivän suunnittelin ja toteutin urakoitsijoiden migraation. Urakoitsijat siirrettiin kannassa eri tauluun, mille on kirjoitettu valmis migraatioscripti. Jouduin muuttamaan Java-koodia niin, että tiedot haetaan nyt eri paikasta.

Torstai 21.12

Alku

Jatkan edellisenä päivänä kesken jääneitä tehtäviä. Tänään tavoitteena opetella lisää JSTL:ää ja JavaScriptiä, koska pitäisi näillä toteuttaa lomakkeelle uusi ominaisuus.

Loppu

Sama asiakas haluaa lomakkeelle uudet tekstikentät mihin annetaan projektikohtaisia tietoja, näitä kenttiä tulisi olla aluksi vain yksi. "Lisää" -nappia painamalla saa lisää rivejä ja jokaisen muun kentän paitsi ensimmäisen viereen tulee "Poista" -nappi. Tätä nappia

painamalla kenttä piilotetaan antamalla sille "display: none" arvo. "Lisää" -nappi poistaa kyseisen arvon. Toteutin tämän JSP-sivulle JSTL:llä ja JavaScriptillä. Kun sain itse logiikan toimimaan, lisäsin vielä tietokantaan uudet validaatiosäännöt niin, että vähintään yksi tekstikenttä on pakko täyttää. Jos käyttäjä on lisännyt ylimääräisiä rivejä, nämä on myös pakko täyttää tai poistaa käytöstä, muuten lomaketta ei voi lähettää.

Perjantai 22.12

Alku

Tänään tarkoitus aloittaa projekti asiakkaalle, joka solmii sopimuksia sisäisen lomakkeen kautta ja heillä on kymmeniä eri osastoja ja jokaisella sopimuksella on kolme eri sopimustyyppiä. Projekti on jo aloitettu, mutta toteutusta pitää muokata selkeämmäksi.

Loppu

Nykyisellä toteutuksella asiakkaalle näytetään siis kaikki noin. 60 eri vaihtoehtoa sivulla ennen sopimuksen luontia, ja osasto määräytyy tämän valinnan mukaan. Tarkoitus on siis siistiä valintaa niin, että asiakkaalle näytetään aluksi vain yksi selkeä apuri-lomake mistä valitaan osasto. Valinnan jälkeen käyttäjä ohjataan takaisin sisäisen sopimuksen luontiin niin, että sen osasto on valmiiksi määritelty käyttäjän aikaisemman valinnan mukaan. Iltapäivästä kävimme vanhemman kehittäjän kanssa läpi epäselviä asioita ns. kyselytunnilla, missä kaikki tammikuussa aloittaneet työntekijät saivat kysyä vapaasti satunnaisista asioista.

Viikkoanalyysi

Migratoin PDF-asetukset integraatiohaaraan ja cherry pickkasin sieltä tietokanta-asetukset jäädytettyyn haaraan. Tämän jälkeen testit menivät läpi ja huokaisin helpotuksesta. Tämän jälkeen tein oman muistilistan kaikista asioista mitä pitää tehdä sen jälkeen, kun vanhempi kehittäjä on saanut ajettua uudet taulut tuotantokantaan. Kun uudet taulut olivat kannassa, otin SSH-yhteyden meidän sisäiselle admin-koneelle, etsin migraattori-ohjelman ja ajoin sitä kautta migraatioloitsut mitkä siirsivät tiedot integraatiotesti-haarasta tuotantoon. Tämän jälkeen tein vielä ne tarvittavat muutokset, mille ei ole valmiita scriptejä, vaan ne piti tehdä manuaalisesti. Syy siihen, että osa muutoksista pitää kirjata manuaalisesti on yksinkertaisesti se, että tietokantaan lisäillään jatkuvasti uusia ominaisuuksia, eikä vuosia sitten kirjoitetut scriptit aina pysy perässä.

Asiakas halusi, että verkkolomakkeiden etu- ja sukunimi-kentät yhdistetään kokonimi-kentiksi, koska he eivät tykännyt siitä, miten nimet generoituivat PDF-tiedostoihin niin, että

nimien väliin jäi turhaa tilaa sen takia, että ei voi tietää kuinka pitkät nimet kaikilla on ja PDF-tiedostossa niille on varattu tietty määrä tilaa. Asiakas oli päättänyt tämän toteutustavan projektipäällikön kanssa, mutta kyseenalaistin koko homman, koska paljon järkevämpi olisi tehdä uusi transformaatiosääntö, missä etu- ja sukunimi yhdistetään piilotettuun arvoon "kokonimi" ja tämä sitten injektoidaan suoraan PDF-tiedostoon. Näin lomakkeet pysyvät selkeänä, koska "kokonimi" kentän alle ei tarvitse ohjeistaa asiakasta kirjoittamaan nimi muodossa "etunimi + sukunimi". Tämä nopeuttaa myös koko kehitysprosessia, koska jos lomakkeen kentät muuttuvat pitäisi kaikki yksikkötestitkin korjata niiden mukaiseksi. Mainitsin tämän projektipäällikölle ja hän hyväksyi, että toteutetaan se mielummin tällä tavalla.

Myöhemmin tuli kuitenkin ilmi, että tämä toteutus ei ollut tarpeeksi hyvä. Heidän toimittamissaan PDF-tiedostoissa lakiteksti oli kirjoitettu siihen muotoon, että sopimuksen osapuolten nimet tulisi injektoida siihen genetiivi eli omistusmuodossa. Ilmoitin ystävällisesti, että tässä ei ole mitään järkeä, koska on lähes mahdotonta kirjoittaa sellaista logiikkaa, mikä osaisi kääntää kaikki maailman nimet omistusmuotoon. Asiakas ymmärsi tämän pointin, ja otti yhteyttä omaan lakiosastoonsa, ketkä muokkasivat PDF-tiedostojen tekstit parempaan muotoon.

3.11 Seurantaviikko 11

Maanantai 25.12

Joulupäivä

Tiistai 26.12

Tapaninpäivä

Keskiviikko 27.12

Alku

Jatkan tänään perjantaina saatua tehtävää, missä asiakkaalle pitää tehdä uusi apuri-lomake mistä valitaan osasto ja sopimustyyppi. Autan myös päivän aikana asiakastukivastaavaa teknisten asioiden selvittelyssä. Tavoitteena kehittää JavaSript taitoja.

Loppu

Sain projektipäälliköltä tehtäväksi selvittää miksi asiakkaalle ei ole lähtenyt muistutusviestejä allekirjoittamattomista sopimuksista. Tarkistin ensin asiakkaan omat muistutus-asetukset, minne oli määritetty, että allekirjoitusvuorossa olevalle loppukäyttäjälle tulisi lähteä muistutussähköpostiviesti kolmen päivän jälkeen, jos sopimusta ei allekirjoita. Tarkistin tietokannasta, milloin sopimus oli luotu ja kuka oli seuraavana allekirjoitusvuorossa. Seuraavaksi etsin kannasta taulun minne muistutusviestien lähetys aikataulutetaan. Viestit olivat aikataulutettu oikeille päiville, mutta ne eivät olleet lähteneet. Selvitin lisää ja huomasin allekirjoitettavien osapuolien taulussa asetuksen mikä esti viestien lähtemisen. Viestit siis aikataulutettiin oikein, mutta lähdekoodia tarkastellessa selvisi, että ne poistettiin eivätkä ne siis lähteneet, jos tämä asetus oli päällä. Korjasin virheen jokaiseen eri ympäristöön ja kuittasin projektipäällikölle, että ongelma on selvitetty ja korjattu.

Torstai 28.12

Alku

Edellisen päivän hommat jatkuvat. Sain eilen tehtyä tietokannan ja frontin kuntoon, nyt alkaa backendin työstö Javan parissa. Olen hyvissä fiiliksissä, koska Java backend on minulle erittäin mielestä puuhaa ja juuri siinä haluaisin kehittyä lisää. Autan myös asiakaspalveluvuorossa aamupäivän, koska yksi työntekijä joutuu olemaan sen ajan poissa. Tänään tavoita opetella lisää Javaa ja meidän erittäin laajaa backend puolen koodia.

Loppu

Aamusta sain yhden selvityspyynnön, missä asiakas kysyi, tarvitseeko heidän allekirjoittajilla ja loppuasiakkailla olla prokuda oikeus, jotta he voivat allekirjoittaa sopimuksen. Tarkistin tietokannasta heidän kyseisen sopimustyypin osapuolien asetukset ja sieltä selvisi, että heidän puolella oikeudet ovat y-tunnus sidonnaiset, joten kaikissa rooleissa saa allekirjoittaa. Myöskään loppuasiakas ei tarvinnut prokuda oikeuksia allekirjoittaakseen, koska he olivat yksityishenkiöitä, eivätkä siis sidonnaisia y-tunnuksiin. Vastasin asiakkaalle suoraan ja suljin tiketin.

Loppupäivän kirjoitin Javalla backendiä. Katsoin ensin mallia, miten sopimuksenluonti tapahtuu yrityksen sisäisessä mallissa, jotta pystyin soveltamaan sitä uuteen Flow-malliin. Kirjoitin uuden Flow-luokan (Fluent API / State machine), mikä syöttää sopimuksen tietoihin oikean sopimustyypin ja osaston, riippuen siitä mitkä valinnat käyttäjä tekee apuri-lomakkeella.

Perjantai 29.12

Alku

Sain eilen jo apuri-lomake perusprosessin toimimaan, mutta siinä on vielä muutama vika mitkä pitää korjata ja ajattelin myös kirjoittaa sille muutaman yksikkötestin. Loppupäivästä saan projektipäälliköltä lisää tekemistä jos saan hommat hoidettua.

Loppu

Sain korjattua vian, missä osasto ei päivittynyt oikein tietokantaan. Vika johtui siitä, että metodissa millä ajoin sopimustyypin tiedot kantaan, ei ollut osastokohtaista logiikkaa.

Loppupäivästä sain tehtäväksi selvittää, miksi IP-rajoite ei toiminut erään asiakkaan palvelussa. Heidän palveluunsa on siis asetettu rajoite, mikä estää lomakkeelle pääsyn kaikista muista paitsi tietyistä IP-osoitteista. Vanhempi kehittäjä demonstroi vian menemällä lomakkeelle puhelimellaan, mikä oli Elisan 4G-verkossa.

Viikkoanalyysi

Keskiviikkona aloin suunnitella uutta apuri-lomake ominaisuutta. Dokumentoin aina nopeasti kaiken, mitä tulen tarvitsemaan projektiin, ennen kuin aloitan itse koodaamisen. Teen isot otsikot kuten tietokanta, frontend, backend, testit, migraatiot, sähköpostit jne. Sitten kirjoitan niihin yksinkertaisin ranskalaisin viivoin tarvittavat toimenpiteet. Kun aloitan itse kehitystyön, korjaan dokumentaatiota yksityiskohdilla, sitä mukaan, kun saan hommia tehtyä. Teen sen näin sen takia, koska yleensä dokumentaatiovaiheessa on mahdoton ottaa aivan kaikki huomioon ja se toimiikin aluksi vain yksinkertaisena listana, mitä nopeasti katsomalla tietää mitä tehdä seuraavaksi.

Tämä on varsinkin nyt tärkeää, koska otimme uuden työnseurantatyökalun käyttöön. Kirjaamme järjestelmään kaikki tehtävät, arvioimme paljon niihin tulee käyttää aikaa ja kirjaamme sitten päivän päätteeksi todellisen ajan mitä tehtävään käytettiin. Tämän avulla projektipäälliköt näkevät kätevästi missä tilanteessa eri projektit menevät ja kenelle voi antaa lisää tehtäviä. Altex Softin (2017) artikkelissa dokumentaation merkitys kiteytetään seuraavasti: Dokumentaation tarkoitus on selittää tuotteen toiminnallisuutta, yhdistää projektikohtaiset tiedot ja mahdollistaa keskustelu tärkeistä kysymyksistä muiden sidosryhmien ja kehittäjien välillä.

Perjantain viat korjasin päivittämällä sopimuksen vasta sen luonnin jälkeen toisella metodilla, millä tämän pystyi tekemään. Tämä metodi tarkisti kannasta, että osaston

nimellä löytyi samasta taulusta id, mikä vastasi sen tagia ja sitten linkitti kyseisen id:n sopimukseen. Poistin sitten model ja view -palautuksen luokasta mikä valmistelee sopimuksen ja kirjoitin kokonaan uuden luokan, mikä hoitaa palautuksen oikeaan näkymään. Näin eri käskyt ovat helpompi eritellä, kun jatkossa kirjoitetaan Flow-mallilla toimivia sopimuksenluontiprosesseja. Tämä tarkoittaa siis, että kehittäjä voi jatkossa valita pieniä valmiiksi kirjoitettuja käskyjä ohjelmalle mitkä sopivat sen projektin logiikkaan. Koodipätkät on aina hyvä eritellä pieniin osiin niin, että ne toimivat silti yhdessä, joten niistä voi sitten kätevästi yhdistelemällä koota isomman paketin.

3.12 Seurantaviikko 12

Maanantai 1.12

Vuoden ensimmäinen päivä.

Tiistai 2.1

Alku

Asiakaspalvelujärjestelmään oli tullut asiakkaalta pyyntö, missä he haluavat uudet PDF-tiedostot palveluun. Vertailen näitä uusia PDF-tiedostoja vanhoihin ja arvioin, kuinka paljon aikaa minulla menee muutoksien tekemiseen. Tavoitteena tänään parantaa omia tehtävänkuvaustaitoja, eli kirjoittaa ylös kaikki tekeminen mitä uusi projekti vaatii ja arvioida näiden tietojen perusteella kuinka paljon aikaa tulen siihen käyttämään. Aika-arvio tarvitaan, jotta projektipäällikkö voi välittää asiakkaalle tarjouksen siitä kuinka paljon uusi muutos tulee heille maksamaan.

Loppu

Vertasin PDF-tiedostoja keskenään ja arvioin, että muutoksiin menisi noin 1,5 henkilötyöpäivää. Uudet PDF-tiedostot vaativat siis lomakkeelle muutoksia, koska idea on, että asiakas täyttää meidän verkkolomakkeen mitä kautta sen tiedot injektoidaan PDF-muodossa olevaan sopimukseen. Uuteen sopimukseen tulee paljon uusia kenttiä ja varsinkin ison työn aiheuttaa muutama liitesivu sopimuksessa, mihin pitää voida lisätä maksimissaan vajaa 20 tosiasiallista edunsaajaa. Koska aikaisemmalla lomakkeella näitä henkilöitä pystyi lisäämään ainoastaan kolme, ei siinä ollut tarvetta selkeyttää lomaketta erillisillä JavaScript –koodeilla mitkä tekevät lomakkeesta luettavamman.

Keskiviikko 3.1

Alku

Jatkan edellisenä päivänä saatua lomakemuutos tehtävää. Sain edellisenä päivänä tehtyä uudet tietokantamuutokset ja JSP-sivun. Tänään pitäisi koodata jQueryllä sivulle härveleitä, jotka näyttävät ja piilottavat divejä käyttäjän valintojen mukaan. Tavoitteena kerrata jQueryä ja toteuttaa härvelit noudattamalla parhaita toimintamalleja.

Loppu

Uudella lomakkeella kysytään tosiasiallisten edunsaajien tietoja ja käyttäjä saa lisätä näitä henkilöitä lomakkeelle maksimissaan 10. Ensin kysytään, onko näitä henkilöitä ollenkaan, jos on, niin aukeaa uusi div, mihin syötetään henkilöiden lukumäärä. Lukumäärän perusteella lomakkeelle aukeaa jokaiselle henkilölle omat kentät mihin syötetään heidän tiedot. Näitten jälkeen on vielä toinen kyllä/ei –kysymys, minkä perusteella aukeaa vielä yksi iso div missä samankaltainen select –valikko ja tähän voi valita maksimissaan kahdeksan henkilöä. Jokaisella henkilöllä on vielä erikseen yksi kyllä/ei kysymys, mistä aukeaa yksi kenttä lisää. Logiikka pitää kirjoittaa niin, että aina kun käyttäjä tekee valintoja, mitkä poistavat jo valmiiksi kirjoitettuja kenttiä, kenttien arvot pitää nollata. Tämä sen takia, että validaatiosäännöt eivät päästä käyttäjää läpi lomakkeesta, jos hän on esimerkiksi täyttänyt kolmen henkilön tiedot ja muistaakin sitten, että näitä henkilöitä on vain kaksi. Kolmannen henkilön kenttä piilotetaan sivulta, mutta tiedot jäävät kenttiin. Koko prosessi on paljon monimutkaisempi kuin mitä arvelin ja joudun käyttämään tähän aikaa. Ilmoitin projektipäällikölle, että 1,5 henkilötyöpäivän arvio oli liian lyhyt ja että tarvitsen lisää aikaa toteutukseen.

Torstai 4.1

Alku

Edellisen päivän hommat jatkuvat, painin lomakkeen jQueryn kanssa. Pääosin lomake toimii niin kuin pitääkin, mutta manuaalisesti testatessa ilmeni vielä muutamia vikoja, mitkä pitää korjata. Tavoitteena jäsennellä jQueryt paremmin luettavaksi ja korjata viat. Loppupäivästä pitäisi vielä asentaa korjaukset testiympäristöön, koska asiakkaalle oli luvattu, että saisimme lomakkeen valmiiksi päivän loppuun mennessä.

Loppu

Painin loppupäivän JavaScript-härvelini kanssa ja ehdin jo turhautuakin hetkeksi, koska suoraan sanottuna koodini oli todella vaikeasti luettavaa. Logiikka näytti toimivan aluksi hyvin, mutta en ollut ottanut huomioon, mitä tapahtuu lomakkeelle, jos käyttäjä yrittää lähettää sitä virheellisillä kenttätiedoilla. Osan koodista olin kopioinut suoraan eräästä

toisesta projektista, mikä teki siitä vielä sekavampaa. Päätin refaktoroida koodin luettavammaksi, ennen kuin aloin korjailla vikoja.

Perjantai 5.1

Alku

Sain asiakkaalta korjauksia, mitkä koskivat edellisen päivän korjauksia. Lomakkeelta pitää muuttaa muutama kenttä oikeanlaisiksi. Tämä sisältää JavaScript muutoksia ja JSP-sivun muokkausta. Property –tiedostot pitää myös muokata suomeksi ja ruotsiksi. Tavoitteena saada nämä muutokset tehtyä niin, että testit menevät läpi. Tämän jälkeen suoritan testiasennuksen, katson että hommat toimivat testiympäristössä ja sen jälkeen siirrän kamat Git-haaraan, mikä jäädytetään ja siitä tehdään maanantaina tuotantoasennus.

Loppu

Poistin vanhan TLS 1.2 varoituksen palvelusta, koska siirryimme jo käyttämään sitä tuotannossa, eli varoitusta ei tarvitse enää näyttää. Jos käyttäjä yrittää nyt tulla palveluun selaimella joka ei tue uutta TLS-suojausta, hänet ohjataan sivulle mikä kertoo hänelle, että hänen tulee päivittää selain. Loppupäivän jatkoin edellisenä päivänä saatujen korjauspyyntöjen tekemistä.

Viikkoanalyysi

Tiistaina jouduin siis muokkaamaan lomakkeen JSP-sivut, kirjoittamaan JavaScriptillä sivulle uudet härvelit, jotka tekevät siitä käyttäjäystävällisemmän, muuttamaan/lisäämään uusia kenttiä tietokantaan, kirjoittamaan PDF-tiedoston tietojen mukaan uudet tekstit kentille suomeksi ja ruotsiksi ja vielä lopuksi muokkaamaan uudet testiluokat. Lähes jokaiselle kentälle tulee kirjoittaa myös uudet validointisäännöt. Validointisäännöt verkkolomakkeella takaavat sen, että loppukäyttäjä ei voi lähettää lomaketta, jos hän on valinnut select –valikosta neljä tosiasiallista edustajaa, mutta täyttänyt vain kahden henkilön tiedot. Muita sääntöjä ovat esimerkiksi prosentti-kentät, mihin pystyy syöttämään vain numeroita väliltä 0-100, tai y-tunnus kentät mihin kelpaa vain oikeassa muodossa oleva y-tunnus.

Torstain jQuery koodit olin kirjoittanut hieman virheellisesti, koska en ollut ottanut huomioon, että lomakkeen tietyt kentät pysyvät samoina, jos asiakas yrittää lähettää formia, mutta validaatiosäännöt estävät sen lähetyksen. Olin siis tehnyt kenttien tyhjennyksen niin, että kun vaikka select –valikosta vaihdetaan arvoksi ”1”, ohjelma ensin tyhjentää kentät 2-10 ja piilottaa niiden divit. Jos käyttäjä kuitenkin lähettää formin ja sivu

latautuu uudelleen, nämä funktiot eivät pyöri, koska ne ajoivat vain silloin kun itse select –valikkoa muutettiin.

Koska koodi oli kirjoitettu aluksi hieman sekavasti, ennen kuin korjaamaan vikaan refaktoroin sen luettavammaksi. Ratkaisin lopulta ongelman kirjoittamalla yhden uuden funktion mikä ajoi aina kun sivu ladattiin uudestaan. Funktio tarkistaa kaikki käyttäjän antamat valinnat ja näyttää/tyhjentää kenttiä sen mukaan. Sain viat korjattua ja kirjoitin sen jälkeen testiluokat uudestaan. Testit pyörivät läpi, suoritin testiasennuksen.

Perjantaina olin ymmärtänyt PDF:ltä yhden asian väärin, en huomannut, kun speksasin työtehtäviä, että erään kohdan vieressä oli kenttä mihin piti tulla tietoa. Tämä johtui pääosin siitä, että itse kenttä oli ns. näkymätön, koska sillä ei ollut näkyviä reunoja. Kenttä paljastui PDF:ltä vasta kun sen kohdalta painoi hiirellä. Korjasin JSP-sivulle uudet kentät ja jouduin samalla muokkamaan jQuery koodeja ja validaatiosääntöjä kannasta. Tämän jälkeen käänsin tekstit ruotsiksi suoraan PDF:ltä saatujen tekstien perusteella, koska en ole hyvä ruotsissa. Lomakkeella oli myös tekstejä mitkä olin lisännyt sinne suomeksi eikä niitä ollut alkuperäisissä PDF-tiedostoissa. Nämä ohjetekstit lisäsin oma-aloitteisesti, jotta lomake olisi käyttäjäystävällisempi. Tarvitsin apua näiden tekstien ruotsiksi kääntämisessä toiselta projektipäälliköltä.

Kun kaikki korjaukset oli tehty, muokkasin uudet testiluokat ja ajoin ne läpi. Migratoin uudet setit testiympäristöön ja kuittasin suoraan asiakkaalle, että korjaukset ovat tehty ja että he voivat nyt aloittaa testauksen. Tämän jälkeen liputin muutokset mitä tarvitaan maanantain tuotantoasennukseen ja migratoin kamat develop-haaraan.

4 Pohdinta ja päätelmät

Menneet 12 viikkoa ovat olleet erittäin opettavaiset ja olen kehittynyt omasta mielestäni paljon näiden aikana. Päiväkirjan pitäminen jokaisesta päivästä on parantanut kykyäni hahmottaa oma työmäärä ja siihen käytettävä aika. Varsinkin alussa, kun emme vielä käyttäneet aktiivisesti minkäänlaista työnseurantatyökalua, oli päiväkirja varsin hyödyllinen. Näin helposti mitä olin tehnyt tietyn viikon aikana ja kuinka paljon olin käyttänyt aikaa jokaiseen eri tehtävään.

Olen omasta mielestäni kehittynyt ajanjakson aikana paljon eri alueilla. Suurin kehitys on tapahtunut Javassa, tietokannoissa ja projektinhallinnassa.

Koulussa tekemiämme projekteja voi hädin tuskin verrata Signomin kokoiseen ohjelmaan. Back endin kirjoittaminen Javalla oli alkuun hieman pelottavaa, koska en ollut aikaisemmin ollut tekemisissä näin ison ohjelman kanssa. Moni nyt jo tutuksi tulleista asioista oli vielä itsellä aivan pimennossa, koska koulussa ei juurikaan käyty läpi parhaita toimintamalleja tai sovellusarkkitehtuuria. Jouduin usein omalla ajalla töiden jälkeen tutkimaan läpi koodia, että sain siitä jotain tolkkua.

Yksi suurin puute koulun opetussuunnitelmassa on se, että siellä ei käydä kunnolla läpi versionhallintaa. Mielestäni jokaisen ohjelmistokehittäjän tulisi hallita Gitin käyttö, ennen oikeaan työelämään siirtymistä. Tietämättömyys Gitin käytössä voi aiheuttaa paljon ongelmia niin itselle kuin muillekin työntekijöille, jotka sitten joutuvat korjaamaan sinun tekemiä virheitä ja näin menettävät paljo kallisarvoista työaikaa.

Tietokannat eivät koulun aikana kiinnostaneet itseäni niin hirveästi, joten kurssitkin suoritin melko pienellä motivaatiolla. Itseäni kiinnosti vain koodaaminen ja ajattelin, etten tule tarvitsemaan tietokantoja työelämässä. Voi kuinka väärässä olinkaan. En siihen aikaan vielä ymmärtänyt kuinka isossa osassa tietokannat ovat ohjelmistonkehityksessä. Karu totuus valkeni kun aloitin työharjoittelun ja tiedostin heti, että joudun opiskelemaan lisää tietokantoja itsenäisesti. Päiväkirjan pitäminen auttoi itseäni tiedostamaan sen, missä aiheissa minulla oli selviä puutteita ja tietokannat olivat yksi näistä.

Tavoitteeni jatkossa on keskittyä Javaan ja varsinkin serveripuolen back endiin. Tykkään Javasta, koska vaikka sillä kirjoittaminen saattaakin olla hieman työläämpää kuin muilla kielillä, se tavallaan pakottaa tekemään asiat parhaaksi todetuilla toimintamalleilla, mitä voi sitten jatkossa soveltaa muihin kieliin. Python on toinen kieli mitä aloin opiskelemaan töiden jälkeen kotona, koska huomasin että sillä oli kirjoitettu paljon töissä käytettäviä automaatio scriptejä ja halusin opetella kirjoittamaan niitä itse. Robinsonin (2017) artikkelissä näkee kuinka nopeasti Pythonin käyttö on kasvanut viimeisinä vuosina, ja on nyt mennyt Javan ja JavaScriptin ohi katsotuissa kysymyksissä.

Lähteet

Mitra, Shamik 2016, Design a Fluent API in Java.
https://dzone.com/articles/java-fluent-api-design. Luettu: 8.12.2017.

Shvets, Alexander 2017, Design Patterns Explained Simply.
https://sourcemaking.com/design_patterns/state/java/5. Luettu: 15.11.2017.

StackOverflow 2010, How to avoid Java code in JSP files?.
https://stackoverflow.com/questions/3177733/how-to-avoid-java-code-in-jsp-files. Luettu: 1.12.2017

Kolodiy, Sergey 2014, Unit Tests, How to Write Testable Code and Why it Matters
https://www.toptal.com/qa/how-to-write-testable-code-and-why-it-matters Luettu: 29.4.2018

Altex Soft 2017, Software Documentation Types and Best Practices
https://www.altexsoft.com/blog/business/software-documentation-types-and-best-practices/ Luettu: 29.4.2018

Robinson, David 2017, The Incredible Growth of Python
https://stackoverflow.blog/2017/09/06/incredible-growth-python/ Luettu: 29.4.2018

Cygnis Media Editor 2015, What is Git? How can it help your development skills?
https://www.cygnismedia.com/blog/what-is-git/ Luettu: 14.5.2018