OULUN AMMATTIKORKEAKOULU

Riihimäki Petri

TESTIAUTOMAATIO CUCUMBER-YMPÄRISTÖSSÄ

Päiväkirjamuotoinen opinnäytetyö

TESTIAUTOMAATIO CUCUMBER-YMPÄRISTÖSSÄ

Päiväkirjamuotoinen opinnäytetyö

Riihimäki Petri
Opinnäytetyö
Talvi 2019
Tietojenkäsittely
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietojenkäsittely

Tekijä: Petri Riihimäki
Opinnäytetyön nimi: Testiautomaatio Cucumber-ympäristössä
Työn ohjaaja: Pekka Ojala
Työn valmistumislukukausi ja -vuosi: 01/2019 Sivumäärä: 43

Opinnäytetyö on päiväkirjamuotoinen tutkielma, jonka tarkoituksena on kuvata automaatiotestaajan työtä kymmenen viikon ajan. Opinnäytetyössä analysoidaan päivittäistä sekä viikoittaista työtä.

Toimeksiantaja ei tahdo, että yrityksen nimeä mainitaan opinnäytetyössä salassapitovelvollisuuden takia. Käytän toimeksiantajasta nimeä X.

Opinnäytetyössä kuvataan testiautomaation tekoa sekä ylläpitoa. Päiväkohtaisesti kerrotaan mitä on tehty kyseisenä päivänä. Viikkoanalyysissä käydään läpi tarkemmin joku tietty aihe sekä paneudutaan asiaan syvällisemmin. Viikkoanalyysissä on käytetty eri lähteitä havainnollistamaan asiaa tarkemmin.

Päiväkirjajakson jälkeen pohditaan testausautomaation kehittymistä ja verrataan sitä alun tilanteeseen. Eli pohditaan miten testiautomaatiossa ollaan kehitytty ja miten sitä vielä voisi parantaa. Käydään myös läpi, miten testaaja voisi vielä kehittyä työssään.

Asiasanat: Cucumber, testiautomaatio, päiväkirja, tietojenkäsittely

ABSTRACT

Oulu University of Applied Sciences
Degree programme in Business Informations Systems

Author: Petri Riihimäki
Title of thesis: Test automation in Cucumber environment
Supervisor: Pekka Ojala
Term and year when the thesis was submitted: 01/2019 Number of pages: 43

This thesis is a diary form study that depicts the work of an automation tester over a period of ten weeks. Both daily and weekly work is analysed.

The employer wishes that the company is left anonymous due to the non disclosure agreement. The employer is referred as X.

The thesis depicts the creation and the administration of the test automation. The work done is reported daily, and the weekly analysis goes through a specific subject more deeply. Different references are used in the weekly analysis to illustrate the matter.

The final part of the thesis reflects the development of the test automation and compares it to the start situation: how test automation has developed and how it still could be improved. In addition the thesis takes a look how the tester could improve more in his work.

Keywords: Cucumber, test automation, diary, information technology

SISÄLLYS

1 JOHDANTO

Tämän opinnäytetyön tavoitteena on seurata automaatiotestaajan työtä päiväkirjamuodossa. Päiväkirjaosuuden aikaväli on 50 päivää ja se sijoittuu ajalle 17.9.2018 – 23.11.2018.

Työ, jota päiväkirjaraportoinnissa kuvaan, on testiautomaatiotehtävät. Työnkuvani on suunnitella ja toteuttaa automaatiotestausta yrityksen nettisivustolle. Sivustoon kuuluu useita alaosioita, joita kutsumme moduuleiksi. Minun työtehtäviini kuuluu lähinnä yhden moduulin testien tekeminen sekä tämän moduulin automaatiotestauksen ylläpitäminen. Jokaiselta työpäivältä olen kirjoittanut päiväkohtaisen kuvauksen siitä, mitä olen tehnyt. Jokaiselta viikolta teen viikkoanalyysin, jossa kokoan yhteen edellisen viikon tapahtumat.

Opinnäytetyö suoritetaan yrityksessä X. Yritys, jossa työskentelen, on kansainvälinen yritys. Se rakentaa älykästä yhteiskuntaa. Yrityksellä on monta eri toimialaa. He hyödyntävät vahvaa kokemusta ja osaamista rakentamalla tietoliikenneyhteyksiä sekä älykkäitä sähköverkkoja. Niissä hyödynnetään yrityksen omaa ohjelmistokehittämistä.

1.1 Työympäristön esittely

Yrityksessä on monta eri toimialaa, joista yksi on oman ohjelmiston kehittäminen isoille suomalaisille asiakkaille. Yrityksellä on useita kumppaneita, joiden kanssa tehdään yhteistyötä. Oulun toimistolla on noin 100 henkilöä töissä. Työntekijöitä on myös muissa kaupungeissa sekä ulkomailla. Englantia käytetään puhekielenä suomen lisäksi. Työhöni kuuluu päiväpalavereihin osallistuminen. Päiväpalavereissa käydään läpi mitä on tehnyt eilen ja mitä tullaan tekemään tänään. Palavereihin osallistuu Skypen kautta myös tiimiimme kuuluvat, ulkomaan toimistolla olevat, henkilöt.

Yrityksen asiakkaat koostuvat tällä hetkellä suomalaisista isoista yrityksistä. Toimintaa on tarkoitus laajentaa myös ulkomaille. Minun toimenkuvaani ei kuulu asiakkaiden kanssa kommunikointi. Minun tehtävänäni on testien ylläpito sekä uusien testien tekeminen.

1.2 Käsitteitä

IntelliJ IDEA = Kehitysympäristö

Gherkin = Cucumberissa käytetty kuvauskieli

Java = Ohjelmointikieli

Git = Versionhallintajärjestelmä

Gerrit = Koodinkatselmointiohjelma

Jenkins = Avoimen lähdekoodin sovellus, jonka kautta voidaan julkaista ohjelmistoja

Cucumber = Avoimen lähdekoodin automaatiotestauksen työkalu

Maven = Javan käännöskone

Jira = Tehtävienhallintajärjestelmä

Scrum = Ketterässä ohjelmistokehityksessä käytettävä projektihallinnan viitekehys

Skype = Videon ja tekstin välitysohjelma

Rajapinta = Eri ohjelmien ja järjestelmien välisten pyyntöjen ja tiedon vaihtoyhteys

2 LÄHTÖTILANNE

2.1 Työn nykytilanne

Olen yrityksessä testiautomaatiotehtävissä. Työnkuvani on suunnitella ja toteuttaa automaatiotestausta yrityksen nettisivustolle. Sivustoon kuuluu useita alaosioita, joita kutsumme moduuleiksi. Moduuleita ovat esimerkiksi asiakas, laskutus ja niin edelleen. Asiakas-moduuli on ainoa, johon itse olen vain tehnyt testejä. Asiakas-moduulissa luodaan asiakas, joka linkittyy muihin moduuleihin. Kun asiakas saa laskun, tiedot yhdistetään moduulien avulla toisiinsa. Testiautomaatioryhmäämme kuuluu minun lisäkseni kolme muuta henkilöä. Kokenein työntekijä on minun ohjaajani. Hän tekee muutamaan eri moduuliin testejä sekä opastaa meitä muita. Me kolme, jotka olemme vähemmän aikaa olleet testiautomaation parissa, keskitymme vain yhteen moduuliin.

Työtehtävät ovat monipuolisia ja niihin kuuluu testien tekemistä ja ylläpitämistä. Osa työntekijöistämme on muissa maissa. Käytämme suomen lisäksi englantia kommunikointiin. Yhteydenpitovälineinä meillä toimivat Skype, sähköposti sekä oma chatti, jonka kautta voimme kommunikoida toistemme kanssa.

Minun työtehtäviini kuuluu lähinnä yhden moduulin testien tekeminen sekä tämän moduulin automaatiotestauksen ylläpitäminen. Koska testiympäristö on niin laaja ja moduulit linkittyvät vahvasti toisiinsa, testit voivat ulottua myös muihin moduuleihin. Näin pääsen näkemään ja tekemään myös muihin moduuleihin testejä. Testiympäristön laajuuden vuoksi en ole tämän kymmenen viikon aikana vielä sisäistänyt edes oman moduulini toimintaa kokonaan.

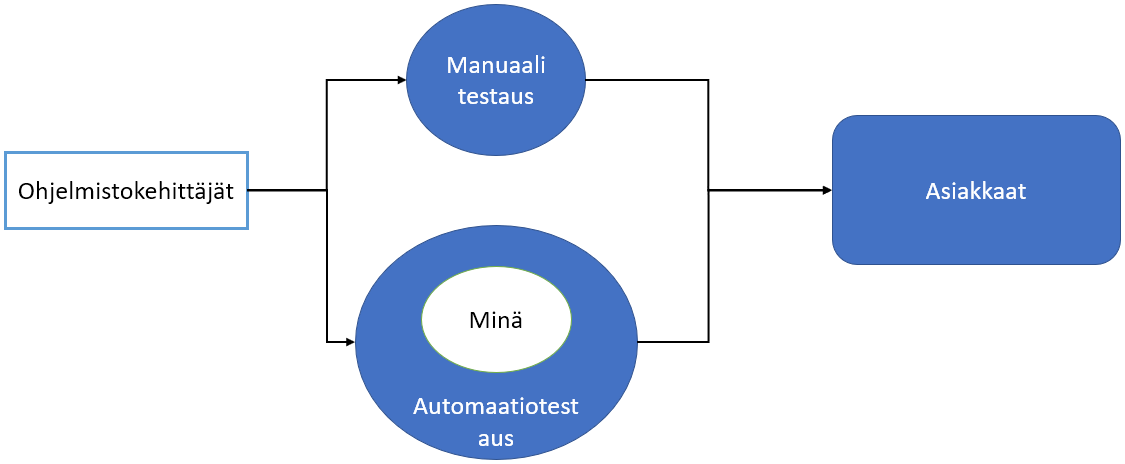
Työssä tarvitaan hyvää ongelman ratkaisukykyä ja loogista päättelyä. Iso osa työstä koostuu koodin kirjoittamisesta ja lukemisesta. Kun testit on ajettu läpi, pitää osata tulkita tuloksia. Jos testit ovat menneet läpi, ei niihin tarvitse puuttua. Jos testit eivät mene läpi, pitää alkaa tutkia missä kohti testit jumittavat. Kun on selvinnyt missä kohti virhe on, pitää tutkia mistä se johtuu. Välillä sivusto on muuttunut sen verran, että testit eivät mene sen takia läpi, ja joutuu tekemään muutoksia testeihin.

Töitä on mahdollista myös tehdä kotoa käsin. Skypen, sähköpostin sekä chatin välityksellä voi kommunikoida työkavereiden kanssa. Avokonttorissa on välillä ääntä, joten kotona on helpompi keskittyä. Työpaikalla on liukuva työaika. Oletetaan kuitenkin, että kaikkiin tarpeellisiin palavereihin osallistutaan. Työn joustavuus antaa vapauden sovittaa työn ja vapaa-ajan tasapainoon. Ei ole tarkkaa, mihin aikaan työt tehdään, kunhan oikea tuntimäärä tulee päivässä täyteen.

Ennen opinnäytetyön aloittamista tein tähän samaan yritykseen opintoihin sisältyvän työharjoittelun. Minulla oli jo käsitys, miten yrityksessä toimitaan. Tämä helpotti töiden aloittamista. Työharjoittelun ajan tein ja ylläpidin testejä käyttäen Robot Frameworkia. Yritys on kuitenkin vaihtamassa Robotin kokonaan Cucumberiin, joten siirryin käyttämään sitä, kun opinnäytetyöni alkoi.

Yrityksessä käytetään useita ohjelmointikieliä, esimerkiksi C#, Javascript, Java sekä monia muita. Työkaluja, joita yrityksessä käytetään, ovat Jira, Jenkins, Git, Gerrit, IntelliJ IDEA ja Postman.

2.2 Sidosryhmät työpaikalla



KUVIO 1. Työpaikan sidosryhmät.

Kuviossa 1 on esitetty yksinkertaistettuna, miten meillä on jaoteltu manuaalitestaajat ja automaatiotestaajat. Manuaalitestaajat ovat jokainen omassa moduulissaan osana tiimiä. Automaatiotestaajat eivät kuulu suoraan mihinkään tiimiin, vaan muodostavat oman tiiminsä, joka tekee kaikkiin tiimeihin testitapauksia. Minä olen keskittynyt poikkeuksellisesti vain yhden tiimin testien tekoon.

Olen osallistunut tämän kyseisen tiimin päiväpalavereihin, mutta en kuitenkaan suunnittelupalavereihin.

3 PÄIVÄKIRJARAPORTOINTI

3.1 Viikko 1

3.1.1 Päiväkirja

17.9.2018

Ensimmäisessä päiväpalaverissa käytiin läpi edellisen sprintin tehtävät. Olin aloittanut Cucumberin käytön kaksi viikkoa sitten. Päivällä oli toinen palaveri, jossa kävimme ohjaajan kanssa läpi, mitä olin saanut tehtyä ja mitä pitää vielä tehdä. Iltapäivällä jatkoin edellisellä viikolla kesken jäänyttä testin tekoa. Testitapauksessa tarvitaan taulukon luentaa ja kun taulukko on luettu, sitä verrataan arvoon, jonka olen antanut arvoksi.

18.9.2018

Aamulla ohjaajaltani kysyin neuvoa eteeni tulleeseen ongelmaan. Ymmärsin, miten kenttien tarkastuksen saisi koodattua. Ymmärtäessäni, miten kyseinen ominaisuus koodataan ja miten sen pitää toimia, testailin erilaisia vaihtoehtoja ja löysin yhden hyvän tyylin rakentaa kyseinen tarkastus.

Lounaan jälkeen aloin koodata kyseistä ominaisuutta kaikille kentille, jotka tarvitsevat kyseistä tarkastusta. Sain työn valmiiksi iltapäivällä ja lähetin Git-versiohallinan kautta sen tarkistettavaksi ohjaajalleni. Ohjaajani tarkisti koodini ja hyväksyi sen. Sen jälkeen koodi siirtyy muillekin, jotka lataavat tai päivittävät oman koodinsa Git-versionhallinnan kautta.

19.9.2018

Aamulla olin auditoriossa, jossa yrityksen isoin johto esitteli yrityksen uutta strategiaa. Ennen lounasta ehdin vielä testata tähän asti tekemieni testien läpimenon. Huomasin vian. Testit pysähtyivät aina samaan kohtaan. Aloin tutkia asiaa ja huomasin, että tietyt kohdat koodissa eivät toimi oikein. Ohjaajan kanssa tutkimme ja huomasimme, että hänen koneellaan vanhat koodit menevät läpi. Omalla koneella jouduin tekemään pieniä muutoksia, jotta saisin testit menemään läpi.

Lounaan jälkeen ohjaajan kanssa mietimme, mitä tehdään. Sovimme, että päivitän kaikki ohjelmat. Tästä seurasi se, että Intellij IDEA -ohjelma meni sekaisin ja sen toimintakuntoon saamiseksi meni loppupäivä.

Tänään varasin myös työkaverin opastuksella ensi viikon maanantaiksi neuvotteluhuoneen. Silloin olisi tarkoitus näyttää muille oman moduulin työntekijöille automaatiotestausta ja testitapauksia, joita olen saanut aikaseksi Cucumberilla

20.9.2018

Aamulla oli päiväpalaveri, jossa kävimme läpi, mitä teimme eilen ja mitä tulemme tekemään tänään. Ennen palaveria sain lähetettyä Git-versionhallintaan eiliset koodit. Ohjaajani tarkisti koodien toimivuuden.

Päivällä tein pieniä lisäyksiä vanhoihin testeihin, jotta ne olisivat kattavampia ja tarkistaisivat paremmin, että ympäristö toimii. Katsoin mallia Robot Framefork -ohjelman testitapauksista, joita voisin muokata Cucumberille sopiviksi.

Iltapäivällä aloin koodata toimintoa, jolla voidaan rajata hakua tietyin ehdoin. Sen verran vaativa on kyseinen toiminto rakentaa, että huominenkin päivä varmaan menee tätä tehdessä.

21.9.2018

Sain lähes valmiiksi eilen aloittamani hakujen rajaamisen. Pientä korjailua vielä vaatii, että kaikki palaset toimivat oikein. Nyt melkein kaikki toimii oikein, mutta joissakin kentissä on pieniä ongelmia toiminnassa. Sain myös käännettyä suomesta englanniksi joitakin kenttiä. Tähän asti koodin kommentointi sekä koodaaminen on tapahtunut suomeksi. Testiautomaatiotiimiin on tulossa yksi ulkomaalainen, joten muutamme suomen englanniksi, jotta hänelläkin olisi helpompaa työskennellä. Hieman työnteko tulee varmaan alussa hidastumaan, kun joutuu tarkistamaan välillä käännöksiä, mutta eiköhän se ajan kanssa parane.

3.1.2 Viikkoanalyysi

Ensimmäisellä viikolla pääsin tutustumaan ketterään ohjelmistokehitykseen käytännössä. Meillä koulussa oli käyty teoriassa läpi millä tavalla ketterä ohjelmistokehitys toimii, mutta käytännön kokemus minulta vielä puuttui.

Scrum-oppaassa sanotaan Scrum-tiimin koostuvan tuoteomistajasta, kehitystiimistä ja scrum masterista. Oppaassa kerrotaan myös, että tiimit ovat itseohjautuvia ja monitaitoisia. Ulkoisen ohjauksen sijaan, itseohjautuvat tiimit päättävät itse, kuinka parhaiten tekevät työnsä. Monitaitoiset tiimit eivät ole riippuvaisia tiimin ulkopuolisista henkilöistä, sillä tiimit ovat monitaitoisia ja heillä on kaikki työn tekemiseen vaadittava osaaminen. Scrumin tiimimallilla optimoidaan joustavuus, luovuus ja tuottavuus. (Schwaber ja Sutherland 2017. Viitattu 28.11.2018.)

Jotta voidaan luoda säännöllisyyttä ja minimoida muiden kuin Scrum-palavereiden tarve, käytetään Scrumissa ennalta sovittuja tapahtumia. Jokaisella Scrum tapahtumalla on maksimi pituus eli ne ovat aikarajattuja. Ennen maksimipituuden täyttymistä tapahtumat voidaan päättää, kunhan on käytetty riittävästi aikaa eikä hukkaa pääse syntymään prosessissa. Itse sprintti on ainoa poikkeus, koska se sisältää muut tapahtumat. (Schwaber ja Sutherland 2017. Viitattu 28.11.2018.)

Mikäli halutaan tarkastella jotain, tarjoaa Scrumin tapahtumat siihen mahdollisuuden. Mikäli halutaan lisätä tuotekehitykselle kriittisen tärkeää läpinäkyvyyttä ja mahdollistaa tarkastelu, on tapahtumat suunniteltu sitä varten. Läpinäkyvyyttä vähentää yhdenkin tapahtuman pois jättäminen. Se johtaa menetettyyn tilaisuuteen tarkastelussa ja sopeuttamisessa. (Schwaber ja Sutherland 2017. Viitattu 28.11.2018.)

Vaikka itse en osallistu kaikkiin scrum-tiimin rutiineihin, pääsen kuitenkin hieman näkemään, millaista se on. Päivittäin osallistun päiväpalaveriin, joka on joka aamu. Siellä käydään läpi mitä kukakin on tehnyt eilen ja mitä tulee tekemään tänään. Päiväpalavereissa on yleensä aina kaikki paikalla, joten on helppo ryhmässä käydä läpi, jos jollakin on jotakin kysyttävää.

3.2 Viikko 2

3.2.1 Päiväkirja

24.9.2018

Tälle iltapäivälle on varattu neuvotteluhuone, jossa esittelen testiautomaatiota muille. Aamulla oli päiväpalaveri, missä kävimme läpi, mitä kaikki olivat tehneet perjantaina ja mitä tullaan tekemään tänään maanantaina. Palaverin jälkeen olisi tarkoitus tarkistaa, että kaikki testitapaukset toimivat iltapäivän palaverissa. Tätä jännitin kovasti, kun osa osallistuu Skypen välityksellä ja esitys piti pitää englanniksi. Itse esittelypalaveri meni omasta mielestäni hyvin. Isoimman ongelman tuotti se, että englannin kielen sujuva puhuminen tuottaa vielä jonkun verran hankaluuksia. Kun kysymyksiä tuli englanniksi, ymmärsin kyllä ne, mutta niihin vastaaminen oli hieman kangertelevaa. Onnekseni mukana oli myös ohjaajani, joka tietää Cucumberista ja testiautomaatiosta paljon enemmän kuin minä.

25.9.2018

Aamu meni kuunnellessa yhtiön tulevaisuuden suunnitelmia ja tietoturva-asioita. Tietoturvasta meille oli puhumassa henkilö yhtiön tietoturvayksiköstä. Vaikka aiheesta ei kauaa luennoitu, on aihe tärkeä, eikä sitä pidä missään nimessä aliarvioida. Täällä painotettiin, että kaikkien pitää olla valppaana kaikennäköiselle tietoturvan väärinkäytölle.

Iltapäivällä jatkoin viime viikolla aloittamaani tehtävää ja sain sen valmiiksi. Keksin lisätä siihen hieman toiminnallisuutta, joka helpottaisi tulevaisuudessa tätä kovasti. Eli haetaan json-sanomaan tallennettua tietoa, joka rajataan oikealle paikalle. En ihan kokonaan saanut toimintoa rakennettua valmiiksi. Ohjaaja näytti, mistä ottaa esimerkkiä ja sen perusteella olen rakentanut kyseistä ominaisuutta.

26.9.2018

Aamulla oli päiväpalaveri. Palaverin jälkeen kyselin yhdeltä työkaverilta olisiko jossain valmiita manuaalitestejä. Voisin ottaa mallia ja tehdä niistä automaatiotestejä. Sain muutaman hyvän linkin Jiraan, jossa oli dokumentoitu testitapauksia. Sain tehtyä haun rajauksen siihen asti valmiiksi, että siihen on helppo lisätä lisää kenttiä, joita hakea. Aloin päivällä tekemään testitapauksia lisää Jiran

dokumentoinnin pohjalta. Tutkin kyseisiä testejä ja huomasin että pitää jonkun verran tehdä lisää toiminnallisuutta, että saa testit rakennettua oikein

27.9.2018

Aamulla aloin tekemään lisää toiminnallisuutta moduuleihin, jotta testejä voi tehdä lisää. Arvojen vertailutarkastus pitää rakentaa, jotta voidaan tarkastaa, että tietyt arvot ovat oikeat. Tätä ennen ei voi testiä jatkaa eteenpäin.

Ohjaajani on jo tehnyt monenlaisia tarkastuksia, mutta ne ovat liian monipuolisia minun tarpeisiini. Minulle riittää, että tietyssä kentässä lukee tietty arvo. Tämän tekeminen osoittautuikin hieman hankalammaksi, kuin aluksi kuvittelin. Alussa tuntui, että eihän se ole kuin verrata tietyn kentän arvoa annettuun arvoon. Sitähän se on, mutta minulla ei ole vielä niin paljon osaamista, että osaisin ihan vielä kyseistä tarkastusta tehdä. Aamupäivä menikin lukiessa ohjaajan koodia ja etsiessä sieltä itselle sopivaa tietoa. Hieman joutui miettimään, miten toteutan kyseisen tehtävän. Lopulta keksin, miten saan kyseisen tarkastuksen tehtyä.

28.9.2018

Aamulla oli päiväpalaveri ja sitä ennen ja sen jälkeen tein taas yhden pienen tarkastuskoodin. Tarkastetaan, onko tietyille riveille kirjotettu mitään. Tämä oli nopea ja helppo tehdä. Kun tämä oli tehty, sain yhden kokonaisen testitapauksen tehtyä valmiiksi. Päivällä aloitin tekemään uutta testiä. Se oli aika nopeasti valmis. Iso osa koodista oli jo valmiina. Uutta koodia ei tarvinnut kirjottaa kuin Gherkiniin, jolla annetaan käskyt, joita ohjelma testaa.

3.2.2 Viikkoanalyysi

Aiemmin tällä viikolla, kun meillä kävi yrityksen tietoturvahenkilö puhumassa tietoturvasta, aloin miettimään, miten yritykset tietoturvariskeihin varautuvat. Minä pääsin käymään yrityksen oman tietoturvakurssin heti, kun tulin taloon sisään. Tämä on minun mielestä hyvä tyyli kertoa kaikille uusille työntekijöille heti alussa, että yritys panostaa tietoturvaan. Tällä kurssilla käytiin läpi ihan perusasioita, esimerkiksi muistitikun käytöstä sekä siitä, miten kannettavaa tietokonetta käytetään yleisellä paikalla sekä muita tietoturva-asioita.

Tietoturvaopas yrityksille -oppaassa sanotaan, että on ratkaisevan tärkeää olla tietoinen, että internet, yritystietoverkot ja laitteet eivät ole turvallisia ilman asianmukaisia varotoimia. Nykyaikaiset

yritystietojärjestelmät ovat monenlaisten vihamielisten toimijoiden kohteena. (Keskuskauppakamari 2016. Viitattu 28.11.2018.) On varmaankin ihan normaalia, varsinkin mitä isompi yritys on kyseessä, että hakkerointiyrityksiä sekä muita tietoturvauhkia on paljon.

Tietoturvaoppaassa kerrotaan myös, että yrityksessä voi olla lukuisia organisaatioon, ihmisiin ja tekniikkaan liittyviä haavoittuvuuksia. Teknologiatoimittajien, palveluntarjoajien ja oman henkilöstön hyvästä työstä huolimatta täydellistä turvallisuutta ei voida saavuttaa. Kyberturvallisuusriskien hallintaprosesseissa onkin arvioitava juuri oman yrityksen heikkouksia ja siihen kohdistuvia uhkia ja suhteutettava ne organisaation tärkeimpiin omaisuuksiin. (Keskuskauppakamari 2016. Viitattu 28.11.2018.) Tämä on mielestäni ihan totta, kun miettii, ettei kukaan ihminen ole täydellinen, joten virheitä ja vahinkoja sattuu välillä. Kun jotain sitten kuitenkin tapahtuu, pitää vain yrittää saada minimoitua vahinko ja ottaa opikseen siitä.

3.3 Viikko 3

3.3.1 Päiväkirja

1.10.2018
Aamulla aloin siistiä koodia ja parantaa sen luettavuutta. Tulevaisuudessa koodista on helpompi ymmärtää, mitä siinä tapahtuu. Huomasin myös, että ensimmäiset testitapaukset oli tehty hieman eri tavoin. Muokkasin suurinta osaa vanhasta koodista yksinkertaisemmaksi.

Seuraavaksi minulle tuli vastaan uusi vertailutehtävä. Siinä piti verrata json-sanoman tietoa jonkun tietyn kentän arvoon. Aloitin tehtävän, mutta en ehtinyt saada sitä kokonaan valmiiksi tänään.

2.10.2018
Aamulla jatkoin vertailukoodin tekoa. Päivällä koodi oli valmis. Seuraavaksi pääsin tekemään uutta testitapausta. Testin lopuksi sivulle tuli teksti ”Virhe sivustolla”. Näytin kyseisen moduulin manuaalitestaajalle testin ja hän oli hämillään viasta. Sovimme, että kysymme huomenna tuoteomistajalta, onko kyseessä bugi vai joku muu vika.

3.10.2018

Aamun palaverin jälkeen näytin eilistä löytöä tuoteomistajalle. Hän sanoi, että kyseessä on selvä bugi. Virheilmoitusta ei pitäisi ilmestyä kyseisessä tapauksessa. Tein Jiraan bugiraportin ongelmasta. Loppupäivän rakensin uutta testitapausta, jossa huomasin hieman samanlaisen virheen, kun edellisessä bugissa. Tässä tapauksessa ei tullut virheilmoitusta, mutta toiminta ei minun mielestäni ollut oikea.

4.10.2018

Aamulla aloin tekemään uutta testitapausta. Huomasin, että pitää tehdä uusi toiminto, jolla saa lisättyä uutta tietoa helpommin. Tällä hetkellä kaikki tieto on kovakoodattu eli sille ei ole annettu muuttujaa. Tarkoitus olisi, että tiedot luetaan muistiin ja asetetaan tiettyyn määrättyyn paikkaan, josta ne voidaan tarkastaa. Tehtävä on aika haastava tehtävä minulle. Onneksi sain ohjaajaltani hyviä vinkkejä, miten ongelmaa kannattaa alkaa ratkaista.

Aiemmin raportoimastani bugista kävin kysymässä tilannetietoja, mutta se ei tainnutkaan olla bugi vaan ominaisuus. Toisilla sivuilla toiminnot on tehty erilaisiksi, joten välillä on haastavaa tietää, mikä on bugi ja mikä ominaisuus. Onneksi asiat selviävät kysymällä. Iltapäivä meni etsiessä tietoa, miten saan koodattua toiminnon, jolla saadaan lisättyä tekstiä tiettyyn sarakkeeseen. Tämä osoittautui todella haastavaksi.

5.10.2018

Aamulla jatkoin eilen aloittamaani sarakkeen tarkastustoiminnon koodaamista. Sen verran haastava tehtävä on, että minun taidoillani ei vielä saada rakennettua kyseistä toiminnallisuutta. Päätän, että maanantaina kysyn apua toiminnon valmiiksi rakentamiseen. Ja jos näyttää, että avusta huolimatta en saa toimintoa tehtyä, siirryn tekemään muuta. Tänään en ehtinyt tehdä muuta, kuin etsiä tietoa, miten kyseisen toiminnon saisi rakennettua.

3.3.2 Viikkoanalyysi

Tällä viikolla tutustuin syvemmin Jira-nimiseen työkaluun, jota yrityksessä käytetään päivittäin. Alussa Jira tuntui minusta tosi sekavalta. En ole ennen päässyt tutustumaan Jiraan. Jiran sivuilla kerrotaan Jira Softwaren olevan ketterä projektinhallintatyökalu. Se tukee kaikkia ketteriä menetelmiä (Scrum, Kanban tms.). Mikäli haluat suunnitella, hallinnoida ja seurata ketteriä ohjelmistokehitysprojektejasi yhdellä ainoalla työkalulla voidaan se toteuttaa ketterien taulujen ja raporttien avulla.

Scrum-tiimi suoriutuu kaikista tapahtumista helposti, koska Jira Software sisältää kattavan valikoiman ketteriä työkaluja. (Atlassian 2018b. Viitattu 28.11.18.) Jirassa voi asettaa kaikki työtehtävät näkymään kiireellisyysjärjestyksessä. Kaikki bugit sekä muut työtehtävät ovat kaikkien näkyvillä. Bugeista ja työtehtävistä ilmoitetaan, missä vaiheessa ne ovat. Ollaanko tietyn bugin korjaus jo aloitettu tai tiettyä toimintoa jo alettu tehdä. Bugi voi olla esimerkiksi laitettu tilaan ”in progress”, jolloin tiedetään, että sitä ollaan korjaamassa. Jokaiseen tapaukseen voidaan myös käydä kommentoimassa tietoja tai kysymyksiä. Tapausten linkittäminen toisiinsa on isossa osassa. Tämä helpottaa etsimistä, kun tapaukset on linkitetty toisiinsa.

Confluencea käytetään myös paljon. Sieltä löytyy kaikki dokumentaatio. Confluencesta luetaan erilaisia ohjeita ja etsitään muuta tietoa yrityksen toiminnasta. Confluencea kuvaillaan, että sillä voi luoda, jakaa ja tehdä yhteistyötä. Yhdestä paikasta löytyvät projektit edistyvät nopeammin. Julkaise, organisoi ja käytä yrityksen sisäisiä tietoja kätevästi yhdestä keskitetystä paikasta, jotta voit auttaa asiakkaitasi auttamaan itse itseään. (Atlassian 2018a. Viitattu 9.12.18.)

Confluence yhdistää itsenäisen työskentelyn nopeuden yhdessä työskentelyn etuihin. Confluencen avulla dokumenttien luominen tiiminä on helppoa. Confluence ja Jira mahdollistavat projektin läpinäkyvyyden ja automaattisen linkityksen jira-tehtävien ja Confluencen välillä. (Atlassian 2018a. Viitattu 9.12.18.)

3.4 Viikko 4

3.4.1 Päiväkirja

8.10.2018

Aamulla katsomme yhdessä ohjaajan kanssa sarakkeen tarkastusta. Sovimme että siirryn tekemään uusia testejä ja ohjaajani jatkaa sarakkeen tarkastuksen parissa jossain vaiheessa. Aamupäivällä aloin tekemään uutta testitapausta, jossa pitää painella painikkeita. Painikkeen painamista ei oltu vielä rakennettu, joten aloin tekemään tällaista ominaisuutta.

9.10.2018

Aamu alkoi sillä, että eilinen lähetys Git-versionhallintaan meni jotenkin pieleen. Latasin uusimman version itselleni, muokkasin siihen muutokset ja lähetin sitten Git-versionhallintaan. Aamulla oli tarkoitus myös jatkaa uusien testien tekoa, mutta huomasin, että kääntäminen suomesta englannin kielelle on aika pahasti kesken. Loppupäivä menikin käännöksiä tehdessä.

10.10.2018

Aamulla ohjaajani tuli kertomaan, että oli saanut korjattua koodia niin, että nyt siinä toimii uuden alasivun avaus taustalle. Tätä mietimme edellisellä viikolla. Nyt kun tämä toimii, voin jatkaa vanhojen testien tekoa, joissa tätä ominaisuutta tarvitaan.

Tietyltä sarakkeelta pitää lukea tiedot ja tallettaa niistä yksi arvo json-tiedostoon. Tehtävä kuulosti helpolta, olenhan jo tehnyt useita samantyylisiä. Tämä eroaa kuitenkin aiemmista siinä, että tässä tallennetaan vain yksi arvo kaikkien arvojen sijasta. Tästä syystä tehtävä tuntui haastavalta. Päivän aikana ajattelin, etten saa ominaisuutta tehtyä. Iltapäivällä huomasin kuitenkin, että tehtävä olikin yksinkertaisempi kuin olin kuvitellut. Tehtävä alkoikin edetä ja sain muokattua vanhaa testiä niin, että testi tuli valmiiksi jo ennen kotiin lähtöä.

11.10.2018

Aamupäivällä jatkoin muutosten tekoa lähes valmiisiin testeihin. Muokkasin niitä ohjeiden mukaan. Toista testiä tehdessä tuli eteen pieni ongelma. Saman nimen alta piti saada tallennettua tiedot json-tiedostoon, mutta asettaessa kyseistä tietoa oli kentän ID eri. Kyseisen kentän ID on eri sen takia, kun yritykset käyttävät y-tunnusta ja ihmisillä on henkilötunnus. Nämä tunnukset on asetettu eri ID:n alle, jotta ne voidaan yksilöidä. Tunnuskentästä haettiin tieto ja asetettiin y-tunnus kenttään. Aluksi minulla oli vaikeuksia ymmärtää, miten testin voi tehdä. Ohjaajan kanssa juteltuani, pääsin kuitenkin etenemään, ja sain muokattua koodin mieleiseksi.

Iltapäivä meni palaverissa. Siellä käytiin perusteellisesti läpi, miten testattava ohjelma on koodattu. Ensi viikolla on samanlainen palaveri UI-asioista. Siellä käydään läpi ympäristön toimintaa UI-näkökulmasta. UI eli käyttöliittymä on se osa sivustoa, joka näkyy käyttäjälle. Isoin osa toiminnoista tehdään UI-kautta eli sen merkitys sivustolle on iso.

12.10.2018

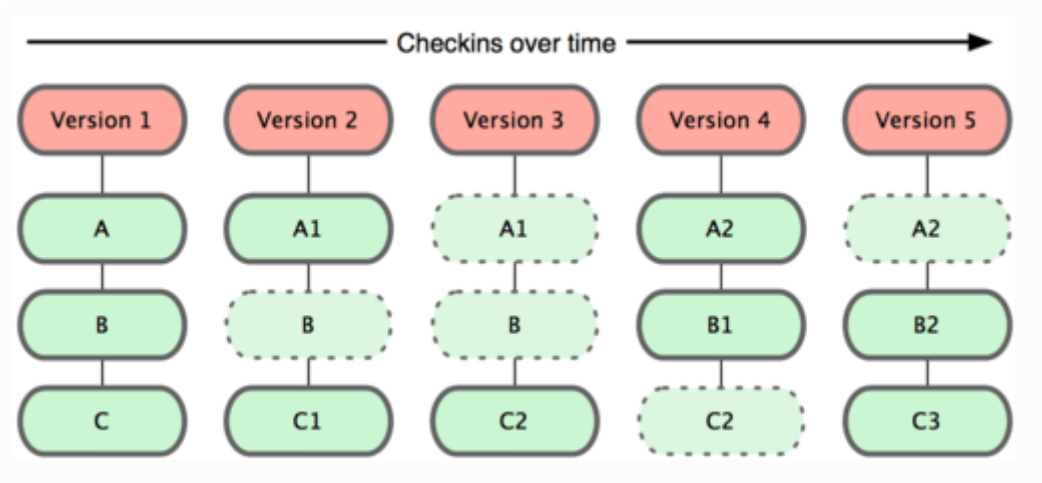
Aamulla aloitin tekemään valintaruudun painamista. Toiminto perustuu tarkistukseen, onko valintaruutu valittu. Jos valintaruutu on valittu, sitä ei valita uudestaan, muussa tapauksessa valitaan.

Aika nopeasti sain toiminnan tehtyä. Testiä tehdessä huomasin kuitenkin, että valintaruutu pitää saada pois päältä jossain tapauksissa. Tämän huomattuani jatkoin ohjauskoodin tekemistä. Toiminnon pitää olla sellainen, että Gherkin-lauseeseen määritellään, pitääkö valintaruutu saada päälle vai pois päältä. Tämä onkin jo hieman vaativampi tehtävä rakentaa. Jatkan toiminnon tekemistä ensi viikolla.

3.4.2 Viikkoanalyysi

Näiden työviikkojen aikana on tullut tutuksi Git, joka on versionhallintajärjestelmä. Gitin sivuilla kerrotaan, että Git-versionhallinta on järjestelmä, joka ajan kuluessa tallentaa muutoksia tiedostoon tai joukkoon tiedostoja, jotta sinä voit palata tiettyihin versioihin myöhemmin (Git 2018. Viitattu 1.12.18).

Sen sijaan Git ajattelee dataansa enemmän kokoelmana tilannekuvia pikkuruisesta tiedostojärjestelmästä. Joka kerta, kun tekee pysyvän muutoksen (commitin), tai tallentaa projektin tilan Gitissä, Git ottaa periaatteessa kuvan siitä, miltä tiedostot näyttävät kyseisellä hetkellä, ja varastoi viitteen tähän tilannekuvaan. Ollakseen tehokas, jos tiedostoa ei ole muutettu, Git ei varastoi sitä uudestaan - vaan linkittää sen edelliseen identtiseen tiedostoon, jonka se on jo varastoinut. (Git 2018. Viitattu 1.12.18.)



*KUVIO 2. Git-versiohallinnan toimintaa (*Git 2018, Viitattu 1.12.18*)*

Normaali Git-työnkulku menee jokseenkin näin:

1. Muokkaa tiedostoja työskentelyhakemistossa.
2. Valmistele tiedostot, lisäten niistä tilannekuvia valmistelualueelle.
3. Tee pysyvä muutos, joka ottaa tiedostot sellaisina, kuin ne ovat valmistelualueella, ja varastoi tämä tilannekuva pysyvästi Git-tietolähteeseen.

(Git 2018. Viitattu 1.12.18.)

Yrityksessä ei vielä ole muilla kuin testiautomaatiolla käytössä Git-versionhallinta. Git-työkalun käyttäminen oli itselleni uutta. Koulussa meillä ei ollut ollenkaan siitä kurssia. Nopeasti kuitenkin omaksuin perusasiat Gitin käytöstä. Kun Gitin käytön oppii hyvin, siitä on tosi iso apu koodien hallitsemisessa. Aina voi palata vanhoihin koodeihin ja tarkistaa, jos joku on mennyt rikki.

3.5 Viikko 5

3.5.1 Päiväkirja

15.10.2018
Sain valmiiksi viime viikolla aloittamani valintaruutujen painamisen. Toiminto on sellainen, että Gherkin-lauseessa annetaan valintaruudun arvo ja tarkistetaan, onko se oikein.

Toinen toiminta, jonka sain rakennettua, on sellainen, että kun tietty valintaruutu oli valittuna, lähtee osasta muita valintaruutuja valinta pois. Tälle toiminnolle piti myös rakentaa tarkastustoiminto, jotta voidaan tietää mitkä arvot ovat valittuna. Tätä toimintoa en ehtinyt saada vielä tänään valmiiksi.

16.10.2018
Aamulla oli koko yksikön kattava palaveri, jossa kerrottiin tulevien julkaisujen ajankohdat sekä käytiin läpi eri tiimien työtilanteet.

Iltapäivällä tarkistin, että kaikki testit toimivat kaikissa ympäristöissä. Lähiaikoina olemme ottamassa testikäyttöön Jenkins-työkalun. Jenkins on avoimen lähdekoodin sovellus, jolla voidaan automatisoida testit, joita ajamme öisin läpi. Kävimme ohjaajani kanssa yhdessä läpi vanhoja Robot Framework -testejä ja ohjaajani otti sieltä hieman mallia uusiin testeihin.

17.10.2018

Päivän aikana sain tehtyä valmiiksi valintaruutujen tarkastuksen ja painamisen. Iltapäivällä oli kahvipalaveri, jossa käytiin läpi yrityksen asioita. Iltapäivällä sain myös tehtyä valmiiksi yhden testin.

18.10.2018

Aamulla kävimme ohjaajan kanssa läpi, millaisia testejä teen lisää. Sovimme, että nykyisistä testeistä tehdään hieman erilaisia variaatioita. Kun variaatioita on tarpeeksi, aletaan tehdä hieman laajempia testejä. Näin saadaan testimassaa lisää ja erilaisia mahdollisuuksia, joilla bugit löytyisivät entistä varmemmin.

Iltapäivällä oli palaveri, jossa käytiin läpi UI-arkkitehtuuria. Tämä palaveri selvensi, miten ohjelma on rakennettu.

Iltapäivällä jatkoin aikaisemmin tehtyä toimintoa. Tässä toiminnossa tarkastellaan historiatietoja ja verrataan niitä nykyisiin arvoihin. Pääsin hyvin alulle, mutta en ehtinyt tehdä toimintoa valmiiksi asti.

19.10.2018

Aamulla sain valmiiksi yhteen kenttään historiatietojen avaamisen ja sulkemisen. Osoitehistoriatietojen tarkastus on jo valmiina, joten ajattelin, että muidenkin historiatietojen tarkastus olisi nopea tehdä. Ei tarvitsisi tehdä muuta, kun vaihtaa olemassa olevaan koodiin oikeat tiedot. Kyseinen toiminto ei ollut kuitenkaan ihan niin nopea tehdä kuin aluksi ajattelin, joten jätin sen myöhemmäksi.

Omaa Jenkins-palvelinta ei ole vielä saatu toimimaan. Ohjaajani aikoo omalla koneellaan ajaa testit läpi, jotta saadaan tietoa, miten testit ja sivut toimivat. Uusia hyviä testejä en saanut tehtyä kuin kaksi kappaletta. Kolmatta testiä aloittaessani huomasin tarvitsevani uutta tarkastusta. Onneksi olin tehnyt jo aiemmin samantyylisen tarkastuksen. En ihan ehtinyt tekemään valmiiksi kyseistä ominaisuutta.

3.5.2 Viikkoanalyysi

Katsoimme ohjaajani kanssa vanhoja Robot Framework testejä. Mietin, mitä kaikkea testiautomaatiolla saadaan tehtyä. Smartbear-sivustolla on hyvin kerrottu asiasta. Jokainen ohjelmistokehitysryhmä testaa tuotteitaan, mutta tuotteeseen jää siitä huolimatta aina ohjelmointivirheitä. Manuaaliset testaajat pykivät löytämään ne ennen kuin tuote julkaistaan, mutta useimmiten ainakin osa ohjelmistovirheistä päätyy tuotantoon asti. Pahimmassa tapauksessa ohjelmointivirheet uusiutuvat, vaikka ne olisivat jo kerran löydetty ja korjattu. Testiautomaatio on paras tapa lisätä ohjelmistotestauksen tehokkuutta, tarkkuutta sekä kattavuutta. (Smartbear 2018, Viitattu 1.12.18.) Olen samaa mieltä siitä, että manuaalitestaajat ja testiautomaatio täydentävät toisiaan hyvin. Varsinkin, kun on kyseessä näinkin iso ohjelma. Testiautomaatio jaksaa tehdä sitä jatkuvaa toistoa, mitä manuaalitestaajan ei mielestäni kannatakaan tehdä.

Tivin artikkelissa, Jos it-järjestelmä on täysi susi, testaus on epäonnistunut, kirjoitetaan Yhdysvaltojen terveydenhuollon uudistuksesta. Barack Obama ajoi hanketta, jossa Healthcare.gov-verkkopalveluun kohdistui suuria odotuksia. Palvelun kautta yhdysvaltalaiset voisivat hankkia itselleen uudenlaisia sairausvakuutuksia. Kun Healthcare.gov avautui lokakuun ensimmäisenä päivänä vuonna 2013, kävi tuskallisen ilmeiseksi, että järjestelmä on viallinen. Vain noin joka sadas vakuutuksen hankintaa yrittänyt pystyi käyttämään palvelua, eikä suuri osa heistäkään saanut hakemusta tehtyä. Palvelun ja hallinnon maineen pelastamiseksi jouduttiin lopulta järjestämään kuukausia kestänyt pelastusoperaatio, johon osallistui it-ammattilaisia ympäri Yhdysvaltoja. (Kotilainen 2017. Viitattu 5.12.2018.)

Tämä esimerkki kertoo karua kieltä siitä, että mitä tapahtuu, kun testaus epäonnistuu. Kyse ei ole pelkästään testiautomaatiosta, vaan koko testauksen onnistumisesta. Kun maltetaan testata kunnolla ja rauhassa loppuun asti, säästytään isoimmilta takaiskuilta. Tällä on myös iso taloudellinen merkitys, kun ei tarvitse korjata useaan kertaan.

Testauksen merkitys kasvaa sitä isommaksi, mitä isommista järjestelmistä on kyse. Näillä isoilla järjestelmillä voi olla jopa henkilöiden henkilökohtaisia tietoja tallennettuna. Jos nämä tiedot pääsevät vääriin käsiin, siitä aiheutuu paljon vahinkoa monelle henkilölle.

3.6 Viikko 6

3.6.1 Päiväkirja

22.10.2018

Aamulla sain koodattua valmiiksi uuden tarkastusominaisuuden. Siinä verrataan asiakastaululta tiettyä arvoa löytyvään arvoon. Hakukenttään annetaan haettava arvo ja verrataan kyseisen haun arvoa. Jos arvon pitää olla true, mutta se on false, tarkastus ei mene läpi vaan tulee virheilmoitus. Loppupäivän aikana sain rakennettua muutaman uuden testin, joissa käytin edellä tekemääni koodia.

23.10.2018

Aamulla jatkoin kesken jäänyttä testiä. Testissä annetaan arvot ja avataan historiatiedot, joihin arvoja verrataan. Lupakentässä historiatietojen tarkastaminen oli hankalampaa. Json-sanoma toimi eri tavalla. Piti tehdä toiminto, joka erottelee Gherkin-lauseesta |-merkillä kaksi eri lausetta. Ensimmäiseen lauseeseen tuli json-sanomasta haku ja toiseen lauseeseen kohde, johon verrataan. Näin sain samaan lauseeseen molemmat arvot.

24.10.2018

Aamu alkoi sillä, että jatkoin laskutusosoitteen ja tilitietojen historiatietojen käsittelyä. Sain rakennettua valmiiksi laskutusosoitteen historian tarkastuksen. Siellä oli yksi valinta, joka aiheutti ongelmia. Kun linkkiä painaa, aukeaa uusi ikkuna. Kun ikkunassa olevaan kenttään yrittää kirjoittaa tietoja, se ei onnistu. Siihen piti rakentaa kiertotie. Ohjaajani keksi, miten asian saa helposti ratkaistua ja minä pääsin jatkamaan koodin kirjoittamista.

Iltapäivällä oli keskiviikkoinen kahvipalaveri, jossa kävimme läpi loppuvuoden toimituksia.

25.10.2018

Aamulla jatkoin tilitietojen historian tarkastusta. Päiväpalaverin jälkeen tuoteomistaja tuli kysymään, millä mallilla testit ovat ja mitä aion seuraavaksi tehdä. Tuoteomistaja kysyi, että voisinko tehdä REST-rajapintatestejä. Niitä ei tule testattua manuaalisesti tarpeeksi, mutta ne olisi hyvä testata. Ohjaajani kanssa asiasta juteltuamme tulimme siihen tulokseen, että REST-rajapintatestejä ei ole vielä tehty. Ohjaajani aikoi ottaa selvää, miten ne saadaan toimimaan.

Iltapäivällä oli testiautomaatiohenkilöiden oma palaveri, jossa kävimme läpi mitä kukanenkin on saanut tehtyä ja miten tästä jatketaan. Vielä ei ole saatu toimimaan Jenkins-serveriä, joten ohjaajani aikoi ajaa testit omalla koneellaan sen aikaa, että Jenkins saadaan laitettua kuntoon.

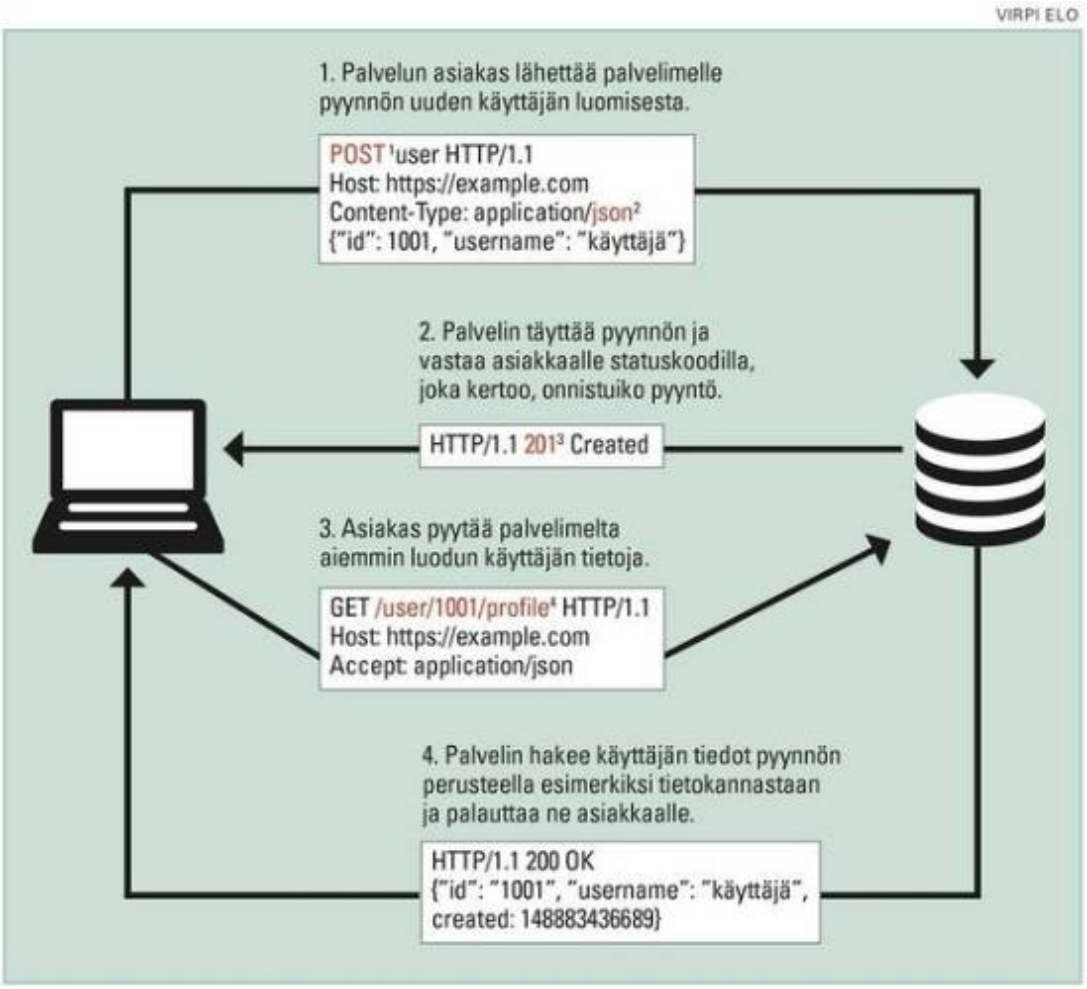
26.10.2018
Sain rakennettua yhden uuden testin. Lähetin oman koodin Git-versionhallintaan ja testasin, että toimivatko testit. Muutama testi ei toiminut, joten niitä pitää katsoa tarkemmin, mikä niissä on vikana.

Ohjaajani sai toisen koneen, jolla aletaan ensi viikolla ajamaan testejä. Näistä testeistä tehdään yhteenvedot näkyville.

3.6.2 Viikkoanalyysi

REST-rajapintaan ei vielä tällä viikolla päästy tekemään testejä, mutta aloin kuitenkin alustavasti tutustua aiheeseen jo nyt. Mikkonen kertoo Tivin artikkelissa, REST:in olevan yleinen arkkitehtuurimalli, jolla voidaan toteuttaa rajapintoja. REST määrittelee, millaisilla operaatioilla palvelinten dataa lisätään, käsitellään ja pyydetään. (Mikkonen J. 2017. Viitattu 1.12.2018.)

Tilattomuus on REST-rajapinnan tärkeä määrittävä tekijä. Tällä tarkoitetaan, että kaksi sellaista pyyntöä, jotka ovat erillisiä, eivät suoraan tiedä toisistaan. Jokaisella pyynnöllä siirretään kaikki pyyntöön liittyvä tieto. Tilan säilyttäminen on pyynnön tekijän vastuulla (esim. selaimessa suoritettava sovellus). Tällä voidaan tarkoittaa esimerkiksi evästeiden siirtoa käyttäjän pyyntöjen mukana, jotta kirjautumistiedot voidaan välittää. (Mikkonen 2017. Viitattu 1.12.2018.)



KUVIO 3. Kuvattu REST-rajapinnan toimintaa (Mikkonen 2017, Viitattu 1.12.2018)

REST:in rakentuminen http-protokollan päälle tarkoittaa, että sillä on käytössään kenties kattavimmin ikinä testattu siirtoprotokolla, jonka päällä vähän karrikoituna koko internet pyörii. Tämän myötä kaikki ne ongelmat, jotka vuosien varrella on http:n osalta ratkaistu, on ratkaistu myös REST:in osalta. (Kivisaari 2018. Viitattu 5.12.2018.)

3.7 Viikko 7

3.7.1 Päiväkirja

29.10.2018

Aamulla ohjaajani kanssa juteltuani tulimme siihen tulokseen, että säädämme testitapaukset niin, että ne toimivat oikein kaikissa ympäristöissä. Tällä hetkellä 26 testistä meni 20 läpi. Osassa vikana oli se, että ne oli tehty eri ympäristöön. Eri ympäristöissä on jonkin verran eroa keskenään. Tulevaisuudessa olisi tarkoitus, että koodi osaisi tarkistaa ympäristön mukaan oikeat asetukset.

Kahdessa testissä oli vikana OK-painike, jota painetaan sen jälkeen, kun päivämäärä on asetettu. Testissä tämä oli toteutettu niin, että painetaan ylhäällä olevaa tyhjää kenttää, jotta saadaan kalenteri edestä pois. Tämä ei toiminutkaan toivotulla tavalla. Ongelma saatiin korjattua siten, että painetaan kyseistä kenttään uudestaan, jolloin kalenteri lähtee edestä pois.

30.10.2018

Aamulla ohjaajani sanoi, että yön testit eivät menneet kaikki läpi. Suoraan koneelta ajettuna olivat kuitenkin menneet läpi. Mietimme, mistä tämä voisi johtua, ja päättelimme, että Jenkins-testien ajaminen on nopeampaa. Siinä ei näy selainta ollenkaan. Tästä johtuu, että jos koodissa ei ole asetettu tarkastusta, joka katsoo, onko sivusto näkyvillä, Jenkins ehtii mennä ohi ja koodi ei pysy perässä testissä.

Lisäilin koodiin lokin kirjotusta. Tämä helpottaa siinä vaiheessa vian etsintää, kun testit hajoavat.

31.10.2018

Aamulla katsottiin, että viime yönä kaikki testit olivat menneet läpi. Tämän aamuna rakentamani testit ovat samantyylisiä kuin edelliset. Muutan hieman arvoja sekä teen muita pieniä muutoksia, joilla saadaan erilaisia testejä aikaseksi.

Iltapäivällä aloitin tekemään uutta testiä. Huomasin nopeasti, että se on hieman haastavampi toteuttaa. Testissä piti asettaa uusia arvoja ja hakea tietynlaisia asiakkaita, joita yhdistetään nykyi-

seen asiakkaaseen. Sitten tarkastetaan, että muutokset ovat tulleet kaikkialle mihin pitääkin. Lisähaastetta tuo se, että lähes kaikki painettavat napit tai kentät, mihin kirjoitetaan, ovat muuttuvia eli ID vaihtuu joka kerta.

1.11.2018
Aamulla olisi tarkoitus rakentaa sellainen testi, jossa yhdistetään kaksi eri osaa yhdeksi. Väliin luetaan json-sanomasta tiedot. Tämä on todella mielenkiintoinen tehtävä ja sopivan haastava minulle.

Aamupäivällä oli Q3-tilanne katsaus, jossa käytiin läpi meidän divisioonamme tietoja. Tosi hyvältä näytti kaikki. Katsauksessa myös julkistettiin uusi innovaatiosivusto, johon kaikki firman työntekijät voivat käydä ilmoittamassa omia ideoitaan. Näistä parhaita ideoita lähdetään jatkojalostamaan eteenpäin.

2.11.2018
Aamulla oli päiväpalaveri, jossa kysyttiin, joko olen saanut tehtyä REST-rajapintatestejä. Niille olisi tarvetta kovasti. Ohjaajallani on kuitenkin ollut kiirettä Jenkins-serverin laittamisessa, eikä minulla ole vielä niin paljon kokemusta, että osaisin tehdä testejä itsenäisesti. Ohjaajani lupasi yrittää saada REST-rajapinnan toimimaan tänään.

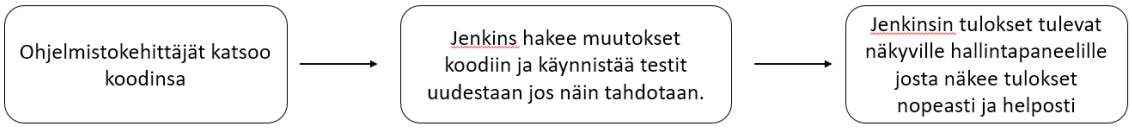
Päivällä sain valmiiksi eilen aloittaman testin. Ohjaajani kanssa juteltuani huomasin, että olin lähtenyt rakentamaan testiä liian monimutkaisesti. Ohjaajani antoi muutaman idean ja kertoi muutenkin, miten testi kannattaa toteuttaa. Tällä tavalla sain ongelman ratkaistua.

3.7.2 Viikkoanalyysi

Tällä viikolla ohjaajani sai asennettua toiselle tietokoneelle Jenkins-serverin, jotta pääsimme ajamaan testejä joka yö. Jenkins on minulle jo tuttu. Tein yritykseen työharjoittelun ennen kuin aloin tekemään opinnäytetyötäni. Työharjoittelun ajan ylläpidin Robot Framework -testiautomaatioympäristöä, joka on Jenkins-serverillä. Tällöin seurasin joka aamu, miten yön ajot olivat menneet. Tein muutoksia Robot Framework -koodiin sen mukaan, miten testit oli menneet.

Jenkins on itsenäinen avoimen lähdekoodin automaatiopalvelin, jota voidaan käyttää automatisoimaan kaikenlaisia ohjelmien rakentamiseen, testaamiseen ja toimittamiseen tai käyttöönottoon liittyviä tehtäviä (Jenkins 2018. Viitattu 1.12.2018). Jenkinsiä on helppo käyttää. Käyttöliittymä on yksinkertainen ja visuaalisesti houkutteleva. Jenkinsissä on tosi alhainen oppimiskäyrä eli voit aloittaa Jenkinsin käytön muutamassa minuutissa. Jenkisin etuja ovat ilmaiset lisäosat sekä suuri ja aktiivinen yhteisö, joka helpottaa mahdollisten kysymysten ja ongelmien kohdatessa. (Smart 2018. Viitattu 1.12.2018.)

Jenkins asennetaan palvelimelle, jossa koodi rakennetaan. Alla olevassa kuviossa 4 on yksinkertainen kuva, miten Jenkins toimii.



KUVIO 4. Kuvattu Jenkins toimintaa

Jenkisin käyttö on mielestäni helppoa ja testien tulokset ovat heti ja hyvin selkeästi näkyvillä. Tämä helpottaa ja nopeuttaa yön testien tuloksien läpikäymistä. Raportin jokaisesta testistä näkee tuloksen kaavioina. Lisäksi tulee myös lokitiedot, josta helppo katsoa, missä kohdassa testi on mennyt rikki. Tämä helpottaa testien tulosten seuraamista varsinkin siinä vaiheessa, kun testejä on paljon. Sivustolla näkyy heti, jos joku on mennyt rikki, tai jos vika on jossain muualla.

3.8 Viikko 8

3.8.1 Päiväkirja

5.11.2018

Aamulla huomasin, että kaikki ympäristöt ovat alhaalla. Teimme testit uudestaan yhdessä ympäristössä, joka toimi. Täällä tuli useita virheitä ja päivän suunnitelmat muuttuivat saman tien. Aloin tutkimaan, mistä ongelmat johtuvat. Huomasin, että testit eivät menneet läpi sen takia, ettei kyseisessä ympäristössä ollut ollenkaan asiakkaita. Tai oikeastaan asiakkaita kyllä on, mutta olen määrittänyt testiin, että etsii ainoastaan minun tekemiäni asiakkaita. Monessa testissä valitaan esimerkiksi neljäs asiakas ja tehdään sille muutoksia. Kun asiakkaita ei löytynyt ollenkaan, testi ei voinut jatkua.

Muut testit eivät menneet läpi sen takia, että eri sivuilla on annettu eri parametrit. Tällä kyseisellä sivulla, missä nyt ajettiin testejä, oli eri parametrit kuin toisessa ympäristössä.

6.11.2018

Aamun aloitin tarkistamalla yön testit. Kaikki testit olivat menneet viime yönä läpi. Jatkoin kesken jääneen testin tekoa. Ensimmäisen testin sain tehtyä nopeasti, kun se oli jo valmiiksi melkein valmis.

Toista testiä tehdessä huomasin bugin ohjelmassa. Tiettyjä lomakkeita avatessa tietyssä järjestyksessä tulee virheilmoitus. Tein tästä bugiraportin Jiraan. Toista testiä en ehtinyt saada valmiiksi.

7.11.2018

Aamulla katsoimme ohjaajani kanssa eilistä ongelmaa, joka minulla oli. Ohjaajani kanssa testattiin, mutta hänkään ei heti keksinyt mikä siinä on vialla. Ohjaajani aikoi jatkaa asian tutkimista jossakin välissä.

Tuli pyyntö tehdä testejä, joissa lähetetään tiedosto ja tarkastetaan tämän jälkeen, ovatko muutokset tulleet näkyville UI:lle. Tätä ohjaajani aikoi katsoa, kun on aikaa. Itselläni ei ole vielä niin paljoa tietoa, että osaisin tätä tehdä.

Koska emme saaneet ratkaistua tuota aiempaa ongelmaa, aloin katsomaan Jirasta jotain helppoa testiä, minkä voisin tehdä tälle päivälle. Iltapäivällä oli kahvipalaveri, jossa käytiin läpi yleisiä firman asioita.

8.11.2018

Aamulla katsottiin eilisen yön testit ja kahdeksan testiä ei ollut mennyt läpi. Tuloksia ja lokia tutkiessani en osannut sanoa, missä vika on. Ajoin omat testit läpi ja tarkistin, ovatko samat viat myös itselläni, vai johtuvatko virheet Jenkins-koneesta. Vika löytyi siitä, että koodiin oli tehty pieni muutos. Muutos aiheutti sen, että koodi hyppäsi oikean kohdan yli, eikä osannut arpoa oikeaa arvoa tietylle kentälle. Olimme tehneet muutoksia koodiin edellisenä päivänä, emmekä olleet miettineet loppuun asti, mihin kaikkeen se vaikuttaa. Näin ongelma tuli esille, kun testit pyörivät joka yö.

Iltapäivällä oli kaikkien testaajien yhteinen palaveri (manuaali ja automaatio). Kävimme läpi ensi vuoden testausta ja sen muutoksia. Pohdimme myös, miten yrityksessä tullaan testaus järjestämään tulevaisuudessa. Sovimme, että pidämme ensi viikolla työpajan, jossa käydään lisää näitä asioita läpi ja sovitaan yhteisistä pelisäännöistä testaajien kesken.

9.11.2018
Aamulla katsoin, että yön testeistä suurin osa ei ollut mennyt läpi. Syy tähän oli, että ympäristöt olivat alhaalla. Laitoin testit pyörimään uudestaan.

Jatkoin eilen aloittamani testin tekoa. Eteen tuli tilanne, että kun yritti tallentaa tietoa, tuli outo virheilmoitus vastaan. Kysyin toiselta testaajalta neuvoa, joka neuvoi tekemään bugiraportin.

Sain korjattua yhden testin, joka ei mennyt läpi. Siinä oli yksi tarkastus, joka ei toiminut ja sen takia testi ei mennyt läpi. Testi oli helppo korjata. Otti vain virheellisen tarkastuksen pois käytöstä. Kun testi meni läpi, siirsin korjatun version Jenkins-serverille.

3.8.2 Viikkoanalyysi

Tällä viikolla meillä oli manuaalitestaajien ja testiautomaation yhteinen palaveri. Kävimme läpi, miten voisimme tehdä yhteistyötä testaajien kesken ja miten voisimme auttaa toisiamme. Mielestäni tällaiset yhteiset palaverit ovat hyvä tapa saada yhdistettyä testausta ja muutenkin ideoida yhdessä.

Talouselämä lehdessä oli minun mielestäni kirjoitettu hyvin palavereiden merkityksestä yrityksessä. Palavereilla ja neuvotteluilla tavoitellaan aina tulosta. Se voi olla myyntiennätyksen saavuttaminen, työpaikan ristiriitojen selvittäminen tai kehittämistarpeiden tiivistäminen. (Vaahtio 2009. Viitattu 9.12.2018.)

Onnistuakseen palaveri tarvitsee kantavat rakenteet. Roolit, rituaalit ja retoriikka eivät viittaa tyhjänpäiväisiin seremonioihin, vaan ovat kaikkien sujuvien työpalavereiden taustalla, kuten myös hiljainen tieto. (Vaahtio 2009. Viitattu 9.12.2018.)

Hiljainen tieto on mielestäni yritykselle sekä hyvä että huono asia. Hiljaista tietoa pitäisi osata jakaa koko yrityksen kesken. Kun yrityksessä pitkään palvellut henkilö jää eläkkeelle tai vaihtaa työpaikkaa, hän on kerännyt hiljaista tietoa paljon. Tämän tiedon saaminen uudelle työntekijälle on tosi iso prosessi. Kun hiljaista tietoa jaetaan eteenpäin, se opettaa kaikille jotain uutta. Myös henkilö, joka jakaa omaa tietoaan, oppii jotain uutta toisilta, kun asiasta keskustellaan. Jos hiljainen tieto jää vain tietyille henkilöille ja nämä henkilöt lähtevät yrityksestä, se on iso menetys yritykselle.

Testiautomaation puolelta ehdotettiin, että manuaalitestaajat, joilla on enemmän tietoa oman moduulinsa toiminnasta, voisivat miettiä, mitä testitapauksia he tahtoisivat automatisoida ja tällä tavalla helpottaa omaa taakkaansa. Tämä myös helpottaisi heidän työtänsä, kun helpot ja puuduttavat testitapaukset voitaisiin automatisoida. Manuaalitestaajat pääsisivät tekemään monipuolisempia ja haastavampia testitapauksia, jota ei kannata automatisoida.

Manuaalitestaajien puolelta tuli kyselyä, että voiko testiautomaatiolla tehdä esimerkiksi dataa sivustolle, jota testaajat voisivat hyödyntää itse omissa testeissään. Asiakkaiden luonti on nopeaa testiautomaatiolla ja asiakkaille voidaan antaa tietyt arvot jo valmiiksi. Toinen asia, mitä mietimme, oli, että onko testiautomaatiolla mahdollista tehdä testin alku, joka on monesti samanlainen, ja jättää testi kesken. Manuaalitestaaja voisi jatkaa tästä eteenpäin testin tekoa. Tällä tavalla pääsisi alun helposta osiosta nopeasti tekemään itse testiä. Tämä myös nopeuttaisi manuaalitestaajan arkea, kun rutiini, joka pitää toistaa tiettyjä testejä tehdessä, vähenisi.

3.9 Viikko 9

3.9.1 Päiväkirja

12.11.2018

Aamulla tarkastin viikonlopun testit. Lauantaina kaikki testit olivat menneet hyvin läpi, mutta sunnuntaina ympäristöt olivat olleet alhaalla, joten testit eivät olleet menneet sen takia läpi. Jatkoin viime viikolla aloittamani testin tekoa, josta tein bugiraportinkin. Tätä testiä ei voinut jatkaa kovin pitkälle, kun ei voinut testata miten testi toimii. Bugia ei oltu ehditty korjata vielä.

Ruoan jälkeen aloin muokkaamaan vanhoja testejä. Yhdistelin ja muokkasin niitä sen verran, että niistä sai tehtyä uusia, monipuolisempia testejä. Näin on mahdollista löytää uusia bugeja.

13.11.2018

Aamulla yritin lähettää Git-versiohallintaan omia muutoksiani. Ohjaajani ei kuitenkaan saanut tarkastettua koodia. Jonkun aikaa yritimme ratkaista ongelmaa. Huomasimme, että helpointa on, kun otan uusimman version ja teen käsin muutokset siihen. Lähetän sitten uudestaan koodit Git-versionhallintaan.

Kun sain tehtyä tämän, ajoin testit läpi omalla tietokoneella. Huomasin että kaikki testit eivät menneetkään läpi. Viidessä testissä oli jotain vikaa. Vikaa etsiessä huomasin, että tekemämme muutos, joka oli siirretty aikaisemmin, ei ollut jostain syystä tallentunut Git-versionhallintaan. Muutos, joka oli jäänyt pois Git-versionhallinnasta, oli sellainen, että |-merkillä erotetaan toisistaan asiat. Tietyissä tapauksissa |-merkki muuttuu _ -merkiksi. Tämä teki sen, että kun json-sanomassa piti olla |-merkki käytössä, eikä sitä saatu muuttaa _ -merkiksi, niin koko testi hajosi. Kun tämän muutoksen teki, kaikki testit menivät läpi ja sain lähetettyä Git-versionhallintaan uudet koodit.

Testauksen työpajassa käytiin läpi, miten lähdemme parantamaan testausta. Mitä muutoksia tulemme tekemään testaukseen. Yhtenä ehdotuksena oli, että testiautomaatio saisi hieman lisää tukea manuaalitestaajilta, joilla on erikoisosaaminen omaan moduuliin. Tätä kautta parantaisimme testiautomaation käyttöä kaikkien hyväksi. Manuaalitestaajille taas saadaan automaatiotestiohjelmista hyvää apua. Ideana olisi, että helpot ja puuduttavat testit olisivat testiautomaation tehtävänä. Monipuolisemmat ja haastavammat testit tekisivät manuaalitestaajat.

14.11.2018

Ohjaajani sai valmiiksi REST-rajapinnan käsittelyn. Tarkoitus olisi tutustua, miten tietoja haetaan, laitetaan ja verrataan REST-rajapinnan kautta. Pitää myös selvittää löytyvätkö rajapinnan kautta tehdyt muutokset UI:lla. Tähän tutustumiseen meni koko päivä.

15.11.2018

Ohjaajani oli tehnyt muutaman testin, joista otin mallia REST-rajapinnan testien tekoon. Kaksi helppoa testiä tein esimerkkien mukaan. Samalla kun rakensin testejä, tutkin miten ohjaajani oli koodannut REST-rajapinnan käytön Cucumberiin. Esimerkkien avulla oli helppo tehdä uusia testejä.

16.11.2018

Aamulla jatkoin REST-rajapintatestien tekoa sekä asensin Postman-nimisen ohjelman, jolla voi lähettää json-sanomia. Postmanilla voi lähettää tietoja serverille ja tietokantaan. Postman-ohjelmaa voidaan käyttää esimerkiksi siihen, että lähetetään json-sanoma, jossa on muutettuja tietoja. Kun tiedot on lähetetty, käydään tarkastamassa UI:n kautta, näkyvätkö ne samalla tavalla myös siellä.

3.9.2 Viikkoanalyysi

Tällä viikolla keskityin Cucumberiin. Cucumberi kuvataan olevan ohjelmistotyökalu, jota ohjelmoijat käyttävät muiden ohjelmistojen testaamiseen. Se suorittaa automatisoidut testit, jotka on kirjoitettu käyttäytymislähtöistä kehittämistä (BDD) hyödyntäen. Cucumberissa käytetään Gherkinin kieltä. Gherkin on kieli, jota Cucumber käyttää määritelläkseen testitapauksia. (Wikipedia 2018a. Viitattu 1.12.2018.)

Gherkinin kieli koostuu otsikoista, skenaarioista ja askeleista (given, when, then). Otsikkoon tulee tieto, mitä kyseisessä testissä tehdään. Jokainen otsikko on kokoelma skenaarioita. Skenaarion alle tulee askeleet, miten testi etenee.

```
Scenario: Eric wants to withdraw money from his bank account at an ATM
    Given Eric has a valid Credit or Debit card
    And his account balance is $100
    When he inserts his card
    And withdraws $45
    Then the ATM should return $45
    And his account balance is $55
```

KUVIO 5. Gherkin-kieltä (Wikipedia 2018a, Viitattu 1.12.2018)

Cucumber on testaustyökalu, jossa kirjoitetaan normaalilla kielellä erilaisia testitapauksia. Testitapaukset Cucumber ottaa vastaan Gherkin-kielellä, joka muistuttaa normaalia kirjoitettua kieltä. Gherkin ymmärtää 70 eri kieltä, joista yksi on englanti. Suomen kielikin on tuettujen kielien joukossa. Alla kuva, joka on kirjoitettu norjan kielellä

```
# language: no
Funksjonalitet: Gjett et ord

  Eksempel: Ordmaker starter et spill
    Når Ordmaker starter et spill
    Så må Ordmaker vente på at Gjetter blir med

  Example: Gjetter blir med
    Gitt at Ordmaker har startet et spill med ordet "bløtt"
    Når Gjetter blir med på Ordmakers spill
    Så må Gjetter gjette et ord på 5 bokstaver
```

KUVIO 6. Gherkin-kieltä norjaksi (Cucumber 2018b, Viitattu 1.12.2018)

language kertoo, millä kielellä Cucumberia käytetään. # language fi on suomen kieli. Jos tämän jättää pois, Cucumber olettaa kielen olevan englanti. (Cucumber 2018b. Viitattu 8.12.2018.)

Jokainen askel määritellään Java-kielellä. Tämä yhdistää sen yhteen tai useampaan Gherkin-askeleeseen. Kun Cucumber suorittaa skenaarion, jossa on monta askelta, siitä muodostuu lause, josta tulee testitapaus. Näitä testitapauksia voi olla useita. Testitapaukset ajetaan yleensä öisin Jenkinsillä, josta tulee raportti aina ajojen jälkeen. Raportit kertovat, miten sivusto on toiminut.

Cucumberissa ei ole sisäänrakennettua selaimen automaatiota. Cucumber tukee kuitenkin Selenium-WebDriveria, joka hoitaa selaimen toiminnot. Selenium-WebDriverista sanotaan Cucumberin sivuilla, että Selenium-WebDriver kehitettiin tukemaan paremmin dynaamisia verkkosivuja, joissa sivun elementit voivat muuttua ilman, että itse sivua ladataan. WebDriverin tavoitteena on tarjota hyvin suunniteltu oliosuuntautunut API, joka tarjoaa paremman tuen nykyaikaisille kehittyneille web-app testausongelmille. (Cucumber 2018a. Viitattu 8.12.2018.)

3.10 Viikko 10

3.10.1 Päiväkirja

19.11.2018

Aamulla katsoin viikonlopun testit läpi ja siellä oli useita testejä, jotka eivät olleet menneet läpi. Aloin katsoa helpoimmasta päästä, mikä niissä voisi olla vikana. Huomasin, että kaksi testiä jää

kohtaan, jossa tarkastetaan, onko virheilmoitusta päällä sivustolla. Tässä on se ongelma, että näissä kahdessa testissä testataan, tulevatko kyseiset virheilmoitukset näkyville. Testit sain menemään läpi, kun otin näiden kahden kohdan virheilmoitusten tarkastuksen pois. Pitää keskustella ohjaajani kanssa, että mitä tehdään. Otetaanko testit pois ajosta vai muutetaanko tuota tarkastusta.

Kaikki REST-rajapintatestit olivat epäonnistuneet viikonloppuna. En päässyt aamupäivällä testaamaan, mistä tämä johtuu. REST-rajapintatestit toimivat vasta yhdessä ympäristössä ja kyseinen ympäristö on alhaalla. Muut testit toimivat kaikissa ympäristöissä, joten niitä voi ajaa muissakin ympäristöissä. Kun ehdimme, siirrämme yötestit myös useammalle muulle ympäristölle.

20.11.2018

Aamulla olin päiväpalaverissa, joka pidettiin Skypen kautta, kun palaverihuone oli varattu. Tämän jälkeen jatkoin eilen aloittamaani testiä. Sain testin valmiiksi, mutta kun ajoin testiä, se ei mennyt kään läpi. Testi jumittui oudosti sellaiseen kohtaan, jossa pitää tehdä kaksi kertaa sama asia eli poistaa asiakas tietystä paikasta. Ensimmäisen asiakkaan testi poistaa oikein. Kun testi yritti poistaa toista asiakasta, ok-painike ei jostain syystä toiminut. Kun testin keskeyttää ja painaa hiirellä painiketta, se toimii oikein. Ohjaajani kanssa mietitimme, mikä siinä voisi olla vikana, mutta emme keksineet ratkaisua.

Ohjaajani oli saanut tehtyä valmiiksi ominaisuuden, jolla REST-rajapinnan kautta voi myös luoda uusia asiakkaita json-sanomaa hyväksi käyttäen. Tähän asti on voinut vain hakea REST-rajapinnan kautta tietoja serveriltä, mutta nyt toiminta toimii myös toisin päin. Aloin katsoa ohjaajani testistä mallia, miten kyseinen ominaisuus toimii ja sitten tehdä samantyylisiä testejä.

21.11.2018

Aamulla jatkoin REST-rajapintatestien tekoa. REST-rajapinnan kautta lähetetään json-tiedosto serverille ja tarkastetaan UI:n kautta, ovatko tiedot oikein. Näitä testejä sain tehtyä kolme valmiiksi asti ja kahta muuta aloitin. Niissä oli pieniä ongelmia, enkä alkanut vielä etsiä, mistä ne johtuivat. Tein helpot testit ensin, jotta testien määrää saataisiin kasvatettua.

Päivällä oli testaajien työpaja, jossa aiheena oli kohti parempaa testiautomaatiota. Työpajassa juttelimme manuaalitestaajien ja automaatiotestaajien kanssa, että miten manuaalitestaajat voivat hyödyntää testiautomaatiota. Työpajassa tuli esille hyviä aiheita, joilla manuaalitestaajat saavat

helpotettua omaa työtään. Esimerkiksi testiautomaatiolla voidaan tehdä kaikki helpot ja yksinkertaiset työt ja manuaalitestaajalle jäisivät sitten haastavammat työt.

Iltapäivällä jatkoin REST-rajapintatestien tekoa. Niissä oli hieman ongelmia, joita ei ehditty ihan korjata tai katsoa, missä vika on. Json-sanomat lähtevät oikein. Mutta kun tiettyjä kenttiä muokkaa, muutokset eivät mene perille, vaan ovat alkuperäisiä tietoja. Osa muutoksista taas menee läpi. Huomenna testaan Postman-nimisellä ohjelmalla, saisiko sen avulla muutokset menemään oikein. Jos se onnistuu, ongelmaa on helppo lähteä ratkomaan.

22.11.2018

Aamulla aloin testata Postman-ohjelmalla toimiiko meidän json-sanomamme oikein. Laitoin Postmanilla uudet arvot ja lähetin serverille. Tarkistin UI:n kautta, että ovatko kaikki tiedot muuttuneet minun asettamikseni. Kaikki tiedot oli vaihtuneet oikein. Json-sanoma toimi oikein. Nyt aloin selvittää, mistä vika voi johtua. Koodia tutkiessani huomasin, että uusi arvo asetetaan, mutta se ei asetu oikeaan paikkaan. Ohjaajani aikoo selvittää asiaa ja minä jatkoin muiden testien tekoa.

23.11.2018

Aamulla juttelin ohjaajani kanssa ongelmasta, jossa tallennetaan json-sanoma. Hän löysi muutaman hyvän linkin, mitkä laittoi sähköpostilla minulle. Aloin tutkimaan kyseisiä sivuja. Päivällä jo luulin löytäneeni oikean koodin, millä luetaan json-sanomasta, mutta se ei toiminut ihan oikein.

Iltapäivällä oli kahden viikon välein pidettävä perjantaipalaveri auditoriossa. Palaverissa oli puhumassa ulkopuolisen yrityksen puhuja testauksesta ja sen laadusta. Hänellä oli mielenkiintoisia ideoita testaukseen liittyen. Tiivistettynä hän sanoi, että kun testaaja löytää bugin, niin hänen kannattaa heti olla yhteydessä kehittäjään kasvotusten. Kehittäjällä voi olla vielä mielessä jotain, mitä hän on miettinyt. Siinä tilanteessa pääsee keskustelemaan parhaiten ongelmasta. Jos on yhteydessä sähköpostilla, tai jollain muulla tavalla, voi asia unohtua tai ainakin kulua paljon aikaa. Kehittäjällä voi olla vaikea muistaa, mitä on miettinyt silloin, kun on tehnyt kyseistä ominaisuutta.

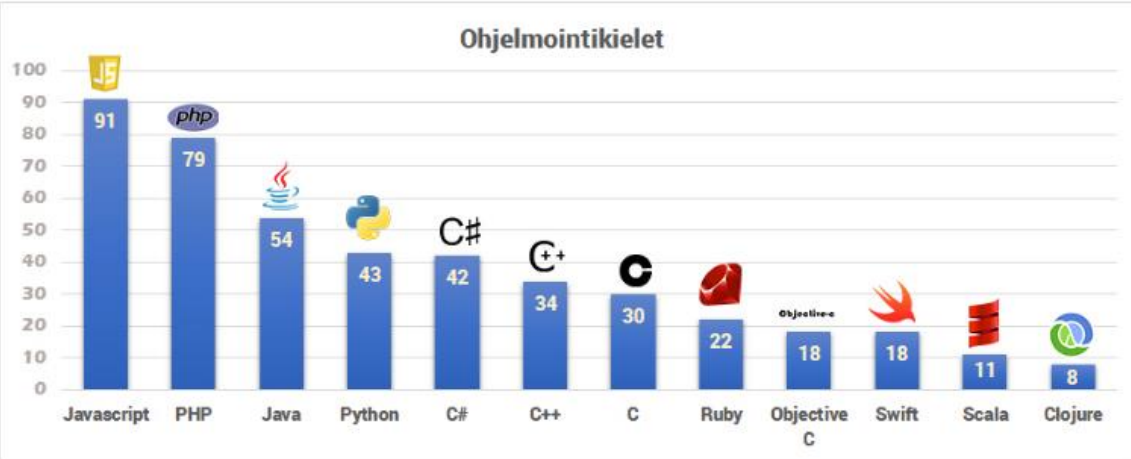
3.10.2 Viikkoanalyysi

Viimeisessä viikkoanalyysissa keskityn Java-kieleen. Laitteistoriippumaton oliopohjainen ohjelmointikeli sekä ajoaikainen ympäristö virtuaalikoneineen ja luokkakirjastoineen kuuluvat ohjelmistoalustaan ja teknologiaperheeseen, jonka Sun microsystems on kehittänyt ja tämän kielen nimi on Java. Noin 3,8 miljardissa laitteessa aina matkapuhelimista supertietokoneisiin, on käytössä Java-alusta. (Wikipedia 2018b. Viitattu 10.12.18.)

Syksyllä vuonna 1995 ilmestyi Java, ja se soveltui www-applettien tekemiseen. Se nosti nopeasti Javan ohjelmointimaailman kuumimmaksi puheenaiheeksi. (Wikipedia 2018b. Viitattu 10.12.18.)

Minulle Java ei ollut ihan uusi kieli. Koulussa olin muutamalla kurssilla käyttänyt Javaa. Ja nyt kun olen töissä kymmenen viikon ajan koodannut Java-kielellä, olen alkanut päästä selville, miten se toimii. Vielä on kuitenkin paljon opittavaa Java-kielestä ja monesta muustakin asiasta koodauksessa sekä testiautomaatiossa, että voi sanoa itseään ammattilaiseksi.

Itewiki sivusto teki vuonna 2017 kyselyn suomalaisille ohjelmistoyrityksille. Täällä Java sijoittui kolmanneksi Javascriptin ja PHP:n jälkeen (Puro 2017. Viitattu 10.12.18).



KUVIO 7. Suosituimmat ohjelmointikielet (*Puro 2017, Viitattu 10.12.18*)

Java-kielellä koodatessa olen huomannut, että kun etsin tietoa internetistä, tietoa löytyy mielestäni tosi hyvin. Paljon löytyy hyviä esimerkkejä, mistä voi ottaa mallia itselleen. Tämä kertoo mielestäni siitä, että Java on suosittu ohjelmointikieli. Tämä helpotti minua tosi paljon, kun löytyi paljon esimerkkejä, joita käyttää hyväkseen.

4 POHDINTA

Tämän kymmenen viikon aikana olen mielestäni kehittynyt tosi paljon koodin ymmärtämisessä sekä sen kirjoittamisessa. Käsitykseni, mitä testiautomaatio on, oli aluksi tosi vähäistä. Ainoat kokemukset rajoittuivat kesällä tekemääni työharjoitteluun, jossa käytettiin Robot Framework-nimistä testiautomaatio-ohjelmaa. Lisäksi olin käynyt yhden ohjelmistotestauskurssin koulussa. Ennen kuin aloin koodaamaan Java-kielellä Cucumberia, minulla oli tosi vähän kokemusta Java-kielestä. Nyt kun sillä on tehnyt töitä päivittäin kymmenen viikon ajan, siihen on tullut jo perusosaaminen. Nyt osaan lukea koodia sen verran, että näen, miten voin sitä hyödyntää.

Alussa tuntui, että isoin osa päivästä meni siihen, että ymmärsi, mitä koodi tekee ja miten se toimii. Työkalutkin olivat minulle uusia, joten niiden opetteluun meni iso osa ajasta alussa. Nyt tuntuu, että minusta on jo apua testien tekemisessä sekä niiden tulosten seuraamisessa. Lokien hyödyntäminen omassa työssäni vaatii vielä hieman parantamista. Tämä tosin on jo parantunut tosi paljon ja siitä on ollut myös tosi paljon apua kun on etsinyt vikaa. Minulle luontaisinta on kasvokkain kommunikointi, mutta työn kautta on tullut sähköpostin sekä Jiran kautta tapahtuva kommunikointikin luontevammaksi.

Opin analysoimaan omaa tekemistäni, kun joka päivä kirjoitin päiväkirjamerkinnän päivän tekemisistä. Alussa, kun kirjoitin päiväkirjaa, tuntui oudolta kertoa päivän tapahtumista. Nopeasti se muuttui luontevammaksi. Kirjoittamisen kautta oppi miettimään päivän päätteeksi, mitä olin päivän aikana tehnyt. Tämä oli itselleni tosi hyvä. Aikaisemmin töitä tehdessä ei ole tullut mietittyä, mitä on päivän aikana tehnyt ja saanut aikaseksi.

Opin koodiin kommentoinnin kantapään kautta. Ennen olen kuvitellut, että tietenkin muistan kaikki, mitä olen itse koodannut ja mitä kirjoittamani koodi tekee. Nyt on niin iso järjestelmä, että kun katsoi viikkojen takaisia omia koodejaan, joutui miettimään, että mitähän se tekee. Tällä tavalla tajusin, että kommentoinnilla on iso merkitys koodatessa. Tätä olen pyrkinyt parantamaan joka päivä ja olen huomannut, että se auttaa myös muistamaan, mitä on koodannut. Uskon, että kun tämän tavan omaksuu vielä paremmin, se helpottaa tekemistä tosi paljon tulevaisuudessa. Tässä on vielä hieman parannettavaa.

Mielestäni tällä päiväkirjamuotoisella opinnäytetyöllä on iso merkitys siinä, että opin analysoimaan omaa tekemistäni. Entiseltä ammatiltani olen automaatioasentaja, jossa työn analysointi ei kuulunut työnkuvaan. Nyt kun joka päivä kirjoitti päiväkirjaa ja analysoi samalla päivän tekemisiä, oppi, mitä on tehnyt päivän aikana ja oppinut samalla. Olen ottanutkin itselleni tavan miettiä hetken työpäivän päätteeksi, mitä olen tehnyt ja mitä olen oppinut päivän aikana. Tämä tapa on helpottanut työn omaksumista.

LÄHTEET

Atlassian 2018a. Confluence. Viitattu 9.12.18, https://fi.atlassian.com/software/confluence

Atlassian 2018b. Jira Software. Viitattu 28.11.2018, https://fi.atlassian.com/software/jira/agile

Cucumber 2018a. Browser automation. Viitattu 8.12.2018, https://docs.cucumber.io/guides/browser-automation/

Cucumber 2018b. Gherkin reference. Viitattu 8.12.2018, https://docs.cucumber.io/gherkin/reference/#spoken-languages

Git 2018. Alkusanat – versionhallinnasta. Viitattu 1.12.18 https://git-scm.com/book/fi/v1/Alkusanat-Versionhallinnasta

Jenkins 2018. Jenkins user documentation home. Viitattu 1.12.2018, https://jenkins.io/doc/

Keskuskauppakamari 2016. Tietoturvaopas yrityksille. Viitattu 28.11.2018, https://kauppakamari.fi/wp-content/uploads/2016/11/tietoturvaopas-yrityksille.pdf

Kivisaari T. 2018. API:t ovat modernin integraatiostrategian ydin. Viitattu 5.12.2018, http://blog.digia.com/rest-api

Kotilainen S. 2017. Jos it-järjestelmä on täysi susi, testaus on epäonnistunut. Viitattu 5.12.2018, https://www.tivi.fi/Kaikki_uutiset/jos-it-jarjestelma-on-taysi-susi-testaus-on-epaonnistunut-6620018

Mikkonen J. 2017. Rest on nettipalveluiden yhteinen kieli. Viitattu 1.12.2018, https://www.tivi.fi/Kaikki_uutiset/rest-on-nettipalveluiden-yhteinen-kieli-6652501

Puro J. 2017. Käytetyimmät ohjelmistokehityskielet, ohjelmointikehykset ja julkaisujärjestelmät suomessa. Viitattu 10.12.2018, https://www.itewiki.fi/blog/2017/05/kaytetyimmat-ohjelmistokehityskielet-ohjelmointikehykset-ja-julkaisujarjestelmat-suomessa/

Schwaber K. ja Sutherland J. 2017. Scrum-opas Scrumin määritelmä ja pelisäännöt. Viitattu 28.11.2018, https://scrumwell.files.wordpress.com/2018/03/2017-scrum-guide-fi-v1-02.pdf

Smartbear 2018. Resources: Automated testing. Viitattu 1.12.18 https://smartbear.com/learn/automated-testing/

Smart J. 2018. Jenkins the definitive guide. Viitattu 1.12.2018, http://barbra-coco.dyndns.org/student/Jenkins-%20The%20Definitive%20Guide.pdf

Vaahtio E. 2009. Ota porukka haltuun palaverilla. Viitattu 9.12.2018, https://www.talouselama.fi/uutiset/ota-porukka-haltuun-palavereilla/515d9200-5438-3cbd-b811-7dbb553a705d

Wikipedia 2018a. Cucumber (software). Viitattu 1.12.2018, https://en.wikipedia.org/wiki/Cucumber_(software)

Wikipedia 2018b. Java. Viitattu 10.12.2018, https://fi.wikipedia.org/wiki/Java