Tampereen ammattikorkeakoulu

Yhtäaikaisen käyttäjämäärän kipupisteen selvittäminen ohjelmallisesti

Nico Bäckman

OPINNÄYTETYÖ
Toukokuu 2022

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

BÄCKMAN, NICO:
Yhtäaikaisen käyttäjämäärän kipupisteen selvittäminen ohjelmallisesti

Opinnäytetyö 24 sivua
Toukokuu 2022

Opinnäytetyössä tutkitaan QAutomate Oy:n kehittämän QAutoCloud-tuotteen yhtäaikaisen käyttäjämäärän kipupiste QAutoRun-suorituskykytestaustyökalulla. Käyttämällä suorituskykytestaustyökalua halutaan löytää käyttäjämäärälle kipupiste, jonka jälkeen palvelu hidastuu merkittävästi.

Opinnäytetyössä käsitellään yleisellä tasolla mitä on ohjelmistotestaus ja miten sitä tehdään. Lisäksi opinnäytetyössä käsitellään tutkimuksen kannalta tärkeitä ohjelmistotestaustekniikoita, joita ovat testiautomaatio, suorituskyky-, kuormitus- ja yksikkötestaus.

Käyttöliittymätestit toteutettiin Robot Framework Browser-kirjaston avulla. Ensimmäinen testi on sisään- ja uloskirjautuminen ja toinen rajapinnasta dataa hakeva testi. Sisään- ja uloskirjautumisella pyritään havainnoimaan, kuinka kauan sivustolle kirjautuminen kestää ja rajapintahaulla rajapinnan nopeutta käyttäjämäärän noustessa. Kuluvaa aikaa mitattiin eri mittauspisteiden avulla. Tavoitteena on, että toteutettuja käyttöliittymätestejä voidaan suoraan hyödyntää QAutoRun-ohjelmistolla.

Tutkimuksen lopputuloksena oli, että molempien testien kipupisteet ylitettiin, kun käyttäjämäärä nousi yli 20 käyttäjään. Tällöin testejä alkoi epäonnistumaan, mikä tarkoittaa sitä, että palvelun käytettävyys alkoi kärsimään. Käyttäjämäärän kasvaessa yli 30 saman aikaiseen käyttäjään sisäänkirjautumiseen kuluva aika alkaa kasvamaan huomattavasti.

Tulosten perusteella saatiin selville, että mikäli uusi QAutoCloud-Webpalvelu pystytettäisiin ja yhtäaikainen käyttäjämäärä ylittäisi 20 käyttäjää, tarvitsisi palvelu joko AWS:n eli Amazon Web Services tarjoaman kuormituksen tasapainottamistoiminnallisuuden tai tehokkaamman AWS-virtuaalitietokoneen.

Asiasanat: suorituskykytestaus, QAutoRun, käyttöliittymätestit

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Software Engineering

BÄCKMAN, NICO:
Programmatically investigating pain point of simultaneous user count

Bachelor's thesis 24 pages
May 2022

This thesis research the pain point of simultaneous user count for QAutoCloud product using performance testing tool QAutoRun which are both developed by QAutomate Ltd. Pain point refers to a simultaneous user count where the products usability starts to get worse from the load caused by the performance testing tool.

Thesis handles what software testing generally is and how it's done. It also handles testing techniques which are relevant to this research. The techniques are test automation, performance, load, and unit testing.

User interface tests were implemented using Robot Framework Browser-library. The first test is login and logout and the second one fetches data from the application programming interface also known as API. Login and logout aim to demonstrate how long it takes to login to the application and the API test how it starts to behave when the user count is starting to increase. Time in these tests were measured by different measuring points. The objective is to utilize these Robot Framework tests directly with QAutoRun.

The result of this research was that the paint point of both tests was exceeded when the user count rose above 20 simultaneous users. When this user count was exceeded, some of the tests started to fail which means that the usability of the product started to get worse. When the user count exceeded 30 simultaneous users, the time to login started to raise notably.

According to the results it was found out that if a new QAutoCloud Webservice is set-upped and the simultaneous user count would exceed over 20 users, the service would either need a load balancing provided by Amazon Web Services or a more powerful AWS virtual machine.

Key words: performance testing, QAutoRun, user interface tests

SISÄLLYS

ERITYISSANASTO

QAutoRun	Kuormitustestaukseen suunniteltu ohjelmisto
QAutoCloud	Keskitetty robotinhallinan Web-käyttöliittymä, jossa hallitaan ohjelmistorobotteja ja niihin liittyviä raportteja
RFW	Robot Framework, testiautomaatio ohjelmistokehys
Framework	Ohjelmistokehys
E2E	End-to-end, Päästä päähän-testaus
API	Application programming interface, ohjelmistorajapinta
Keyword	RFW-avainsanoja, perusta mihin kaikki robottitestit rakennetaan
AWS	Amazon Web Services, alusta erilaisille tietojenkäsittelypalveluille pilvessä.

1 JOHDANTO

Nykypäivänä sovelluksia ja palveluita on enemmän kuin koskaan aikaisemmin ja tämän myötä niihin kohdistuu myös enemmän odotuksia. Loppukäyttäjän näkökulmasta niiden pitäisi vastata ilman viivettä ja mikäli siinä epäonnistutaan, niin pahimmassa tapauksessa asiakas voidaan silloin menettää. Ajatellaan, että palvelu, joka on hidas tai kaatuu avattaessa, ei yksinkertaisesti toimi. Suorituskykytestaus auttaa vastaamaan kysymykseen, että kuinka paljon eri käyttäjiä voi samanaikaisesti käyttää palvelua, ennen kuin se hidastuu tai pahimmassa tapauksessa kaatuu.

Opinnäytetyössä tutkitaan QAutoCloud-Webpalvelun yhtäaikaisen käyttäjämäärän kipupistettä RFW-käyttöliittymätesteillä ja QAutoRun-ohjelmalla. QAutoRun on pilvessä toimiva rasitus- ja testaustyökalu, jolle pystytään ajoparametreina syöttämään sekä virtuaalisten käyttäjien että tietokoneiden määrä. Näiden avulla pystytään suorittamaan kuormitustestausta QAutoCloud-Webpalveluun, kirjaten ylös vasteajat. Jossain vaiheessa saavutetaan kipupiste, jolloin sivuston vasteaika kasvaa ja suurin osa testeistä ei enää mene läpi. Testaus toteutetaan QAutoCloudin eri osa-alueisiin, joita ovat sisään- ja uloskirjautuminen sekä datan hakeminen rajapinnasta.

1.1 QAutomate Oy

QAutomate Oy on vuonna 2015 perustettu ohjelmistotuoteyhtiö, joka tähtää kansainvälisille markkinoille ja toimii teknologiapartnerina Q-Factory Oy:n kanssa. Q-Factory Oy on vuonna 2010 perustettu ohjelmistotestaukseen ja laadunvarmistukseen erikoistunut yritys.

QAutomate Oy on kasvuyritys, joka omaa huikean potentiaalin kasvavassa ohjelmistorobotiikan alueessa. QAutomate-tuotteet tarjoavat yrityksille Web-maailmassa testauksen automatisointia, RPA-robotteja ja DevOps-ratkaisuja kustannustehokkaasti. Asiakkaisiin kuuluu esimerkiksi isoja verkko-operaattoreita ja pienempiä yrityksiä.

2 OHJELMISTOTESTAUS

Ohjelmistotestaus on ohjelmistotekniikassa tapa tutkia ohjelmiston toimintaa verifiointi- ja validointimenetelmillä. Näillä varmistetaan, että tuote, palvelu tai järjestelmä täyttää sille asetetut vaatimukset ja että se sopii aiottuun käyttötarkoitukseen. Nämä ovat tärkeä osa laadunhallintaa. Ohjelmistotestaus voi myös tarjota tasapuolisen näkökulman ohjelmistoon, minkä avulla asiakas pystyy ymmärtämään ohjelmiston käyttöönottoon liittyviä riskejä. (Singh & Singh, 2012)

KUVA 1. Ohjelmistotestauksen alalajeja

2.1 Kuormitus- ja suorituskykytestaus

Kuormitus- ja suorituskykytestauksella pyritään tutkimaan miten testattava tuote vastaa ja suoriutuu käyttäjäkuorman noustessa. Lisäksi sen avulla voidaan tutkia tuotteen vasteaikaa, resurssien hallintaa ja skaalautuvuutta. (Molyneaux, 2014)

KUVA 2. Suorituskykytestaukseen liittyvät testaustavat

Kuormitustestauksen tärkeys yleisesti ymmärretään, mutta usein niitä ei kuitenkaan ajeta niin hyvin tai usein, kuin pitäisi, sillä kuormitustestaus ei ole helppoa. Kehittäjällä tulee olla sekä tarvittavat taidot käyttää testaukseen käytettäviä työkaluja oikein ja tehokkaasti että ymmärrys, mitä kuormitustestauksella pyritään saavuttamaan ja kuinka tavoitteet saavutetaan (Sogeti, 2021). Kuormitustestauksen tarkoituksena ei ole löytää ohjelmiston vikoja, vaan sillä pyritään pääsemään eroon suorituskyvyn pullonkauloista.

Kuormitustestauksen avulla pyritään löytämään alueita, jotka vaativat kehitystä ennen tuotteen viemistä esimerkiksi markkinoille. Useimmat suorituskykyyn liittyvät ongelmat kohdistuvat nopeuteen, vasteaikaan, latausaikaan ja huonoon skaalautuvuuteen. Näistä tärkeimpänä kuitenkin pidetään nopeutta, sillä hitaasti toimiva ohjelmisto voi pahimmassa tapauksessa menettää käyttäjänsä. Kuormitustestauksella tarkastetaan, että ohjelmisto toimii tarpeeksi nopeasti säilyttääkseen käyttäjän huomion ja mielenkiinnon.

2.2 Yksikkötestaus

Yksikkötestaus on osa ohjelmistosuunnittelua ja tarkoittaa testausta, jossa varmistetaan ohjelmiston tietyn osa-alueen koodi. Yksikkötestit kirjoitetaan erilleen ohjelmistosta käymään läpi esimerkiksi funktioita, metodeja tai luokkia yksitellen ja riippumattomasti. Yleensä kehittäjät kirjoittavat yksikkötestit ohjelmointityön yhteydessä varmistaakseen, että haluttu toiminto toimii kuten on suunniteltu. Yksikkötestaamisella ei pystytä yksistään testaamaan koko ohjelman toiminnallisuutta, vaan ennemminkin sen avulla varmistetaan, että ohjelmiston osa-alueet toimivat toisistaan riippumatta. (Runeson, 2006)

Yksikkötestejä pyritään kirjoittamaan koko ohjelman kehitystyön ja elinkaaren aikana pyrkien minimoimaan siinä esiintyvät virheet. Pitkällä aikavälillä virheiden poistaminen varsinkin alkuvaiheessa on tärkeää, koska niiden löytäminen voi aiheuttaa valtavia kuluja myöhemmin.

2.3 Testiautomaatio

Testiautomaatio on prosessi, jossa hyödynnetään automaatiotyökaluja testitietojen ylläpitämisessä, testien suorittamisessa ja testitulosten analysoimisessa tavoitteena ohjelmiston laadun parantaminen. Sen tehtävänä on varmistaa, että melkein jo kaikille elämänalueille ulottuvat ohjelmistot, joista jopa ihmishenget voivat olla kiinni, toimivat käyttäjän tarpeisiin vastaten virheettömästi ja turvallisesti.

Testiautomaatiolla pystytään ratkomaan ja säästämään aikaa tärkeissä, mutta itseään toistavissa prosesseissa. Testiautomaatio on tärkeä osa ”jatkuvan toimituksen” ohjelmistoja, esimerkiksi sellaisia, joista uusi versio tulee tuotantoon kuukausittain tai jopa viikoittain, sillä sen avulla pystytään nopeasti selvittämään, että onko päivityksen yhteydessä jokin osa-alue mennyt rikki. Kaksi yleisintä osa-aluetta, johon testiautomaatiota tehdään, ovat graafisten käyttöliittymien testaus ja API-testaus. Testiautomaatiolla ei pystytä korvaamaan täysin kaikkea testausta, mutta oikein suunniteltuna ja toteutettuna se pystyy vapauttamaan testaajan testaamaan kohdennetusti tai tutkivasti ohjelmiston kriittisiä osa-alueita.

Testiautomaatio-frameworkeja on erilaisia, joista jokainen tarjoaa oman arkkitehtuurin ja hyvät sekä huonot puolet. Opinnäytetyössä käytetään Keyword-painotteista Robot Framework:ia, joka on suunnattu hyväksymistestaukseen ja hyväksymistestausvetoiseen ohjelmistokehitykseen.

TAULUKKO 1. Kuvitteellinen sisäänkirjautumisprosessi verkkosivustolle Keyword-tyyppisellä testiautomaatio-frameworkilla

Vaihe	Kuvaus	Keyword	Kohde
1.	Avaa kirjautumissivu	openSite	
2.	Syötä käyttäjätunnus	inputUser	Käyttäjätunnus kenttä
3.	Syötä salasana	inputPassword	Salasana kenttä
4.	Varmista kirjautumistiedot	verifyInformation	
5.	Kirjaudu sivustoon	loginSite	Kirjaudu-nappi

Hyviä puolia keyword-pohjaisessa frameworkissä on muun muassa:

- Minimalistiset skriptaustaidot riittävät
- Yhtä keywordia voidaan käyttää useissa testeissä, joten koodi on uudelleenkäytettävää
- Testit voidaan rakentaa erilleen testattavasta ohjelmistosta
- Selkeä ja helppolukuinen raportti ajotuloksista

Huonoja puolia:

- Ympäristön asennus on kallista, aikaa vievää ja monimutkaista
- Vaatii testaus- ja automaatioymmärrystä
- Ylläpito voi olla haastavaa laajassa projektissa

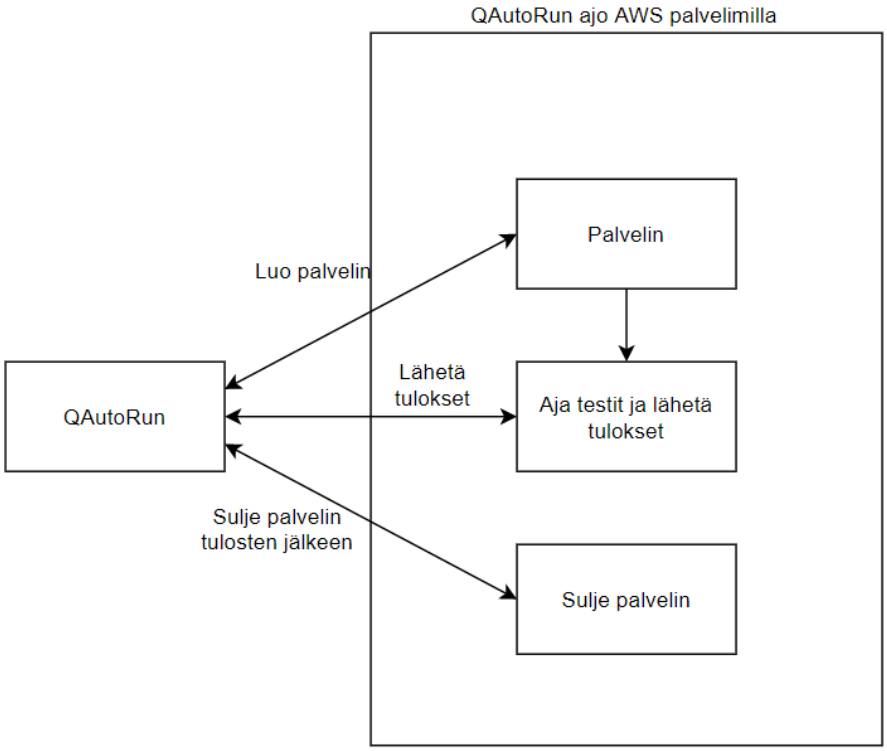
3 TYÖKALUT

3.1 Robot Framework

Robot Framework on geneerinen avoimen lähdekoodin automaatio-framework, jota voidaan käyttää testiautomaatioon ja ohjelmistorobotiikkaan. RFW omaa helposti ymmärrettävän syntaksin hyödyntäen helppolukuisia keywordeja. Toiminnallisuutta voidaan halutessaan laajentaa myös muilla ohjelmointikielillä, esimerkiksi Pythonilla tai Javalla. (Robot Framework, 2022)

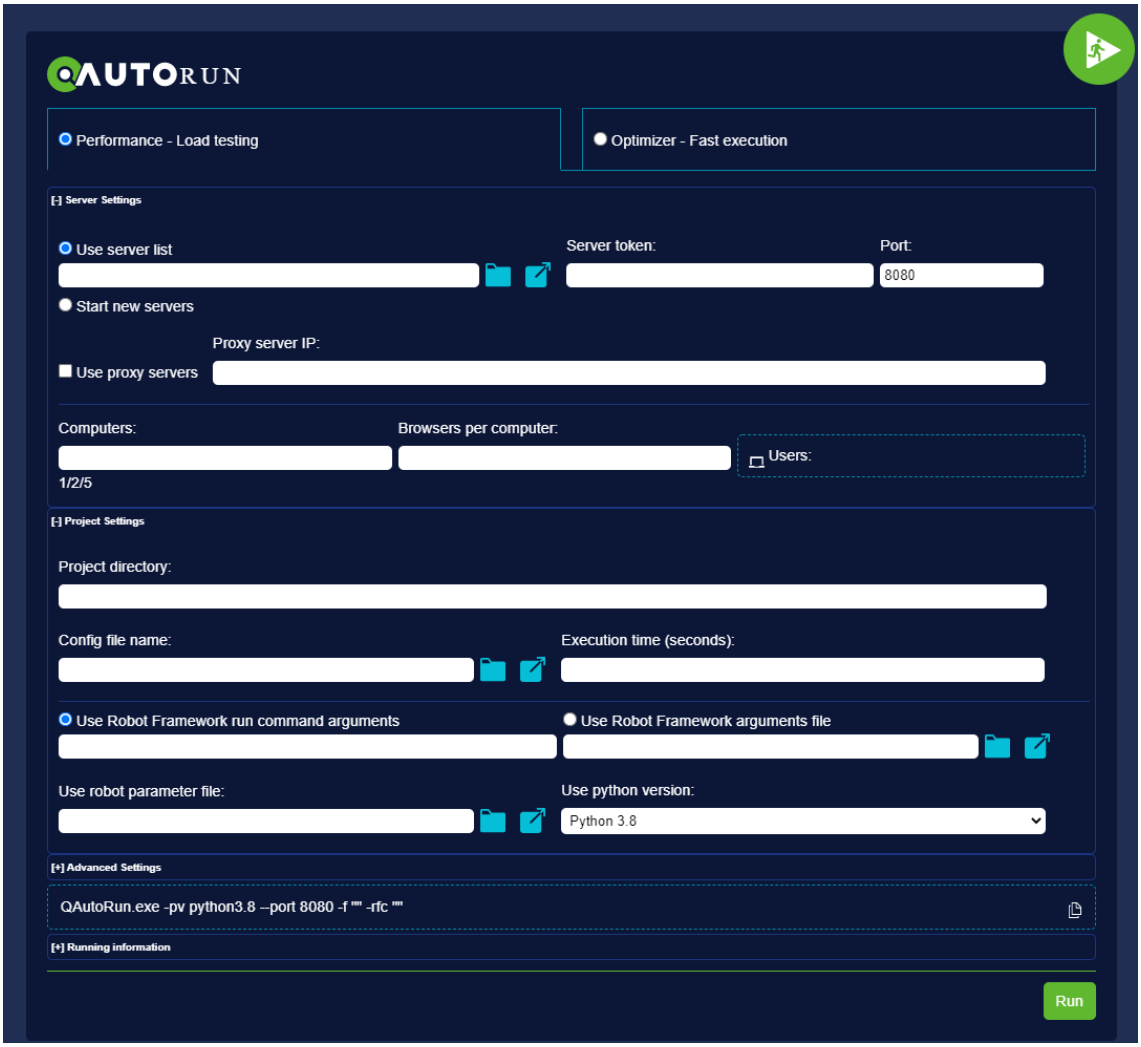
3.2 QAutoRun-tuote

QAutoRun on pilvessä toimiva verkkopalveluiden rasitus- ja testaustyökalu. Sen avulla voidaan helposti luoda erilaisia stressitestejä palveluita varten määrittelemällä halutut virtuaalikäyttäjät ja -koneet, testin ajotiheyden ja testien kokonaismäärän. Se sisältää myös testiajojen optimointiominaisuuden, jonka ansiosta tunteja kestävät testit voidaan ajaa vain minuuteissa (QAutomate Oy, 2022). QAutoRun:lla voidaan kuormittaa RFW-automaatioskriptejä.



KUVA 3. QAutoRun-ajo AWS-palvelimella

QAutoRun-ohjelmistoa voidaan käyttää komentorivin tai Visual Studio Code -lisäosan kautta. Kuvasta 4 katsottuna ”Use server list”-riville syötetään verkkosivuston ip/osoite, ”Computers”-riville tietokoneiden lukumäärä ja ”Browsers per computer”-riville selainten määrä per tietokone. ”Use Robot Framework run command arguments”-riville voidaan määritellä vielä erikseen ajoparametrejä. Tämän jälkeen ”Run”-nappia painamalla testit lähtevät käyntiin valituilla asetuksilla ja ajon päätteeksi on nähtävillä loki. Mikäli ohjelmistoa ajetaan komentoriviltä, niin parametrit syötetään silloin ajokomentoon.



KUVA 4. QAutoRun Visual Studio Code-lisäosa

3.3 QAutoCloud-tuote

QAutoCloud on pilvessä tai paikallisesti toimiva, ohjelmistorobottien hallinta- ja seurantatyökalu (QAutomate Oy, 2022). QAutoCloud koostuu Dashboard eli kojelauta toiminnallisuudesta sekä robottien orkestrointi kokonaisuudesta. Dashboardin avulla voidaan sekä julkaista että päivittää robotteja, seurata robotin ajonaikaista toimintaa sekä raportoida robotin tallentamia tietoja. Robotti voidaan käynnistää manuaalisesti tai se voidaan ajastaa käynnistymään automaattisesti tiettyinä ajanhetkinä. Robotin asetuksissa voidaan määrittää robotin toimintaan liittyviä asioita. Yksi keskeisimmistä toiminnallisuuksista ajastuksen lisäksi on sähköpostien lähettäminen onnistuneista tai epäonnistuneista ajoista. Orkestroinnin avulla hallitaan robotin ajoympäristöä. Robotteja voidaan ajaa joko Linux- tai Windows-ympäristöissä. Orkestroinnin tärkein tehtävä on tarjota robotille oikeanlainen ajoympäristö.

4 TESTIEN SUUNNITTELU JA TOTEUTUS

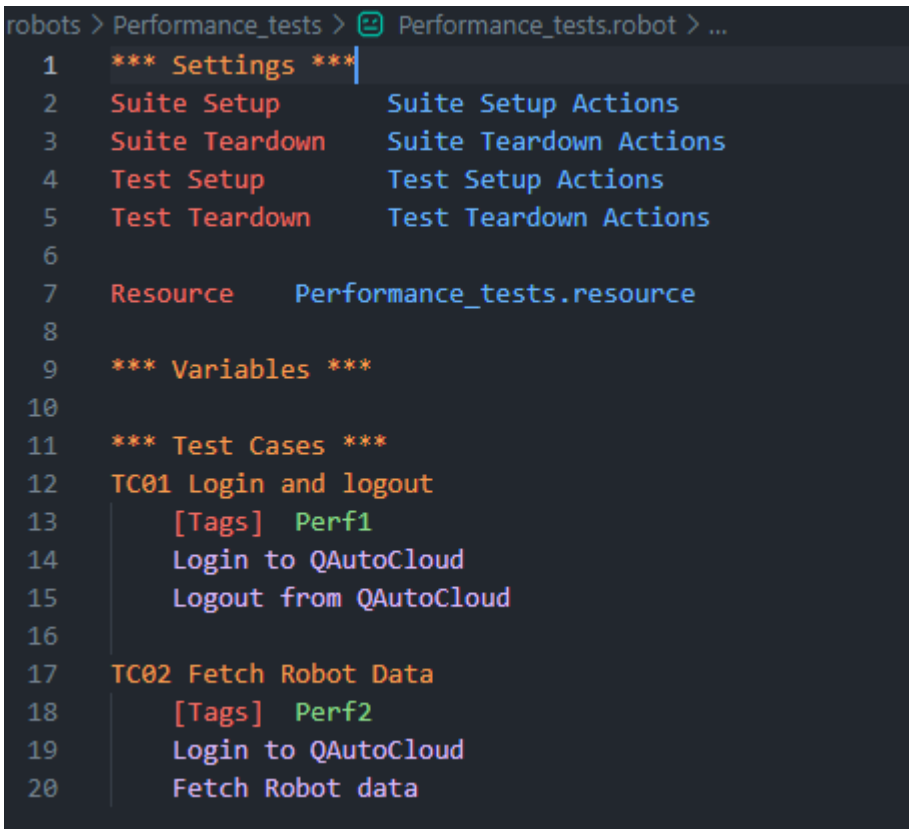
Automaatiotestit toteutetaan QAutoCloud-Webpalvelua vasten RFW Browser-kirjastolla. Browser-kirjasto hyödyntää JavaScript pohjaista teknologiaa nimeltään Playwright. Playwright on Microsoftin kehittämä avoimen lähdekoodin framework, joka on suunniteltu E2E-testaukseen.

Testien tavoitteena on selvittää QAutoCloud-Webpalvelun yhtäaikaisen käyttäjämäärän kipupiste. Koska QAutoCloud-palvelu toimii AWS-virtuaalipalvelimella, virtuaalitietokoneen suorituskyvyllä on suuri merkitys, kuinka paljon kuormaa se kestää. Toinen vaikuttava tekijä on se, että onko sivustolla käytössä AWS:n tarjoamaa kuormituksen tasapainottamistoiminnallisuutta (Load balancing). Opinnäytetyössä testattavan QAutoCloud-palvelun AWS-instanssityyppi on t3.medium eli palvelu toimii virtuaaliietokoneella, jossa on kaksi ydintä ja neljä gigatavua keskusmuistia ja käytössä ei ole kuormituksen tasapainotusta.

4.1 Testisetit

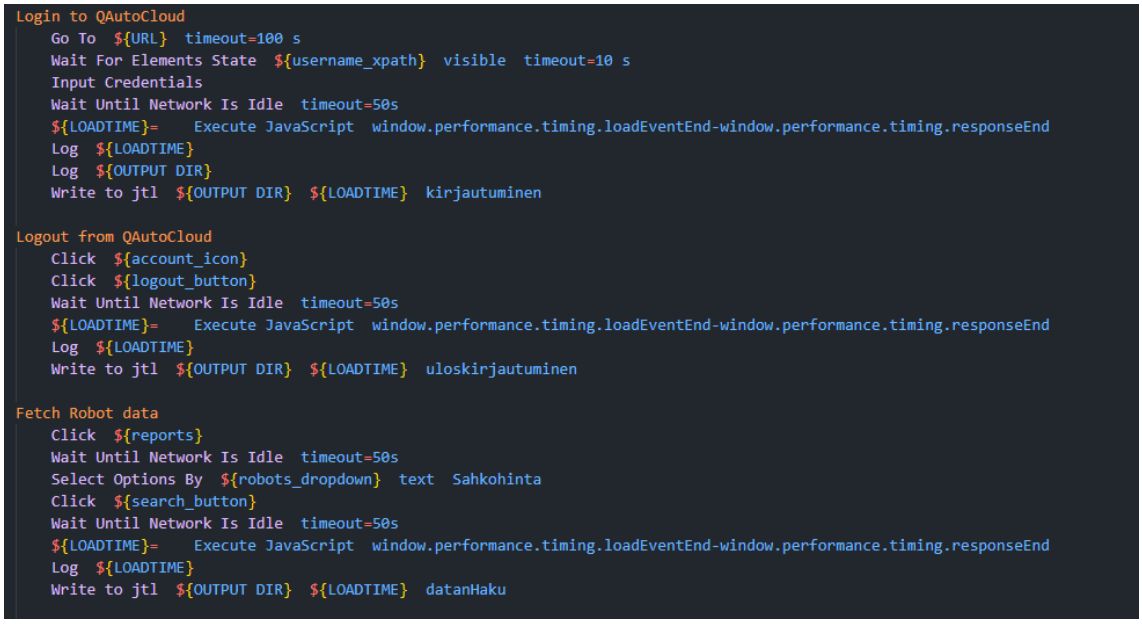
Testeihin on toteutettu perinteinen sisään- ja uloskirjautuminen TC01 (kuva 5). Testissä mitataan aikaa sekä sisään- että uloskirjautumisessa, jotta voidaan havainnoida, kuinka käyttäjämäärän noustessa sivusto alkaa käyttäytymään. Käyttäjämäärän noustessa voidaan olettaa, että sisään- ja uloskirjautuminen hidastuu ja lopulta sivustoon ei saada enää ollenkaan yhteyttä. Testin tavoitteena on löytää käyttäjämäärä, jossa sisäänkirjautuminen hidastuu merkittävästi ja mahdollisesti suurin osa testeistä ei enää mene onnistuneesti läpi.

Toinen testi TC02 (kuva 5) kirjautuu sivustolle, hakee robotin kirjaamaa dataa rajapinnasta ja mittaa tähän kuluvaa aikaa. Testin ideana on havainnoida, kuinka käyttäjämäärän noustessa rajapintaan kerätyn datan hakeminen hidastuu.



KUVA 5. Robotin ajamat testit

RFW-testien toiminnallisuus kirjoitetaan tyypillisesti keyword:inä. Kuvasta 6 nähdään keywordit, joita kuvan 5 testit käyttävät.



KUVA 6. Robotin käyttämät keywordit

Muuttujat toimivat argumenttien tavoin testeissä. Kuvassa 7 on muutama robotin käyttämä muuttuja, jotka ilmoitetaan XPath-kielellä. XPath-kieli perustuu HTML-dokumentin puumuotoiseen esitystapaan ja antaa siten mahdollisuuden poimia eri osia dokumentista tietyillä valintakriteereillä ja automaatio-framework käyttää näitä osia suorittamaan ohjelmoituja käskyjä.

```
*** Variables ***
${account_icon} =  //a[@id="account_icon"]
${logout_button} =  //a[contains(.,"Log out")]
${reports} =  //span[contains(., "REPORTS")]
${robots_dropdown} =  //select[@id="robot_name"]
${search_button} =  //button[@class="btn btn-success border-light check_sensitive float-right"]
```

KUVA 7. Robotin käyttämiä muuttujia

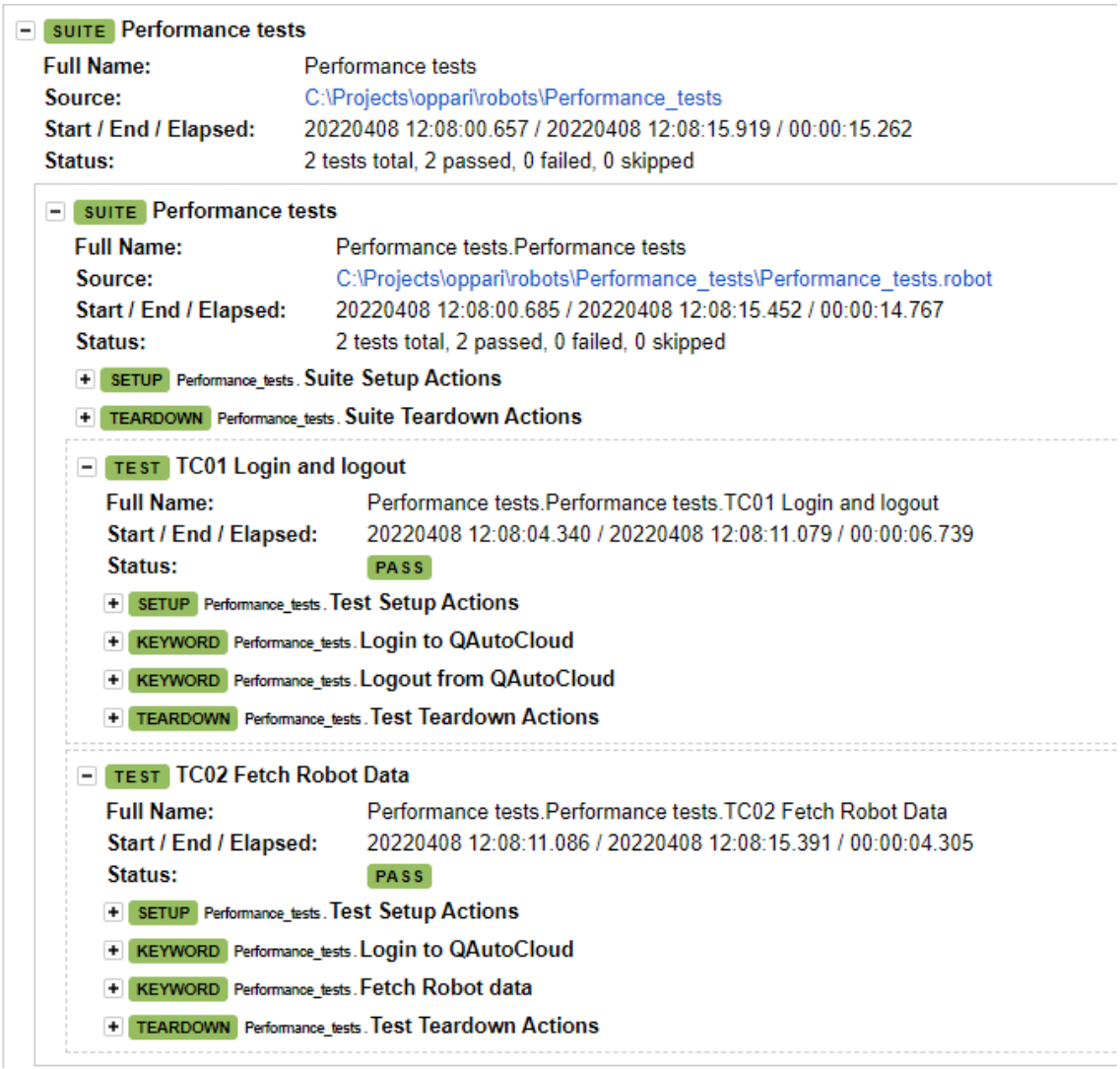
XPath:it voidaan hakea esimerkiksi Google Chromen kehittäjätyökalun avulla verkkosivulta. Kuvasta 8 nähdään esimerkkielementti, joka viittaa sisäänkirjautumisessa käyttäjätunnuskenttään. Näitä tietueita voidaan tallentaa muuttujiin koodin paremman ymmärrettävyyden kannalta ja siten käyttää erilaisiin datan syöttöihin.

```
<div class="input-group form-group"> flex
  <i id="input-icons" class="fas fa-user">…</i>
  <input id="email" name="email" required size="35" style="text-align:center;font-size:16pt;padding:5px
  0;border-radius:5px;" type="text" value> == $0
</div>
```

KUVA 8. Elementin XPath selaimesta haettuna

5 TULOKSET

Kun testejä ajetaan onnistuneesti Robot Framework:llä, siitä palautuu lokitiedosto. Kuvasta 9 nähdään onnistuneen ajon lokitiedosto ja tiedostolta pystytään seuraamaan jokaisen keywordin suorittamat tehtävät. Jos testi ei mene läpi, niin epäonnistunut keyword näkyy lokilla punaisella, joka mahdollistaa helpon seurannan.



KUVA 9. Onnistuneen ajon lokitiedosto

Robottiskriptiin on kirjattu mittauspisteitä, joiden avulla pystytään seuraamaan kuluvaa aikaa ja ne tallennetaan tiedostoon. Kuvasta 10 nähdään yhdellä käyttäjällä kirjatut mittaustulokset sisään- ja uloskirjautumisessa sekä rajapintahaussa.

KUVA 10. Kirjatut mittauspisteet testeissä

5.1 QAutoRun-tulokset

Kuormitustestaus aloitettiin yhdellä virtuaalitietokoneella ja viidellä virtuaalikäyttäjällä ja koneiden määrää alettiin nostamaan jokaisella ajokerralla, kunnes sivuston kipupiste saavutettiin. Tuloksissa käytetään keskiarvoa onnistuneista mitatuista ajoista.

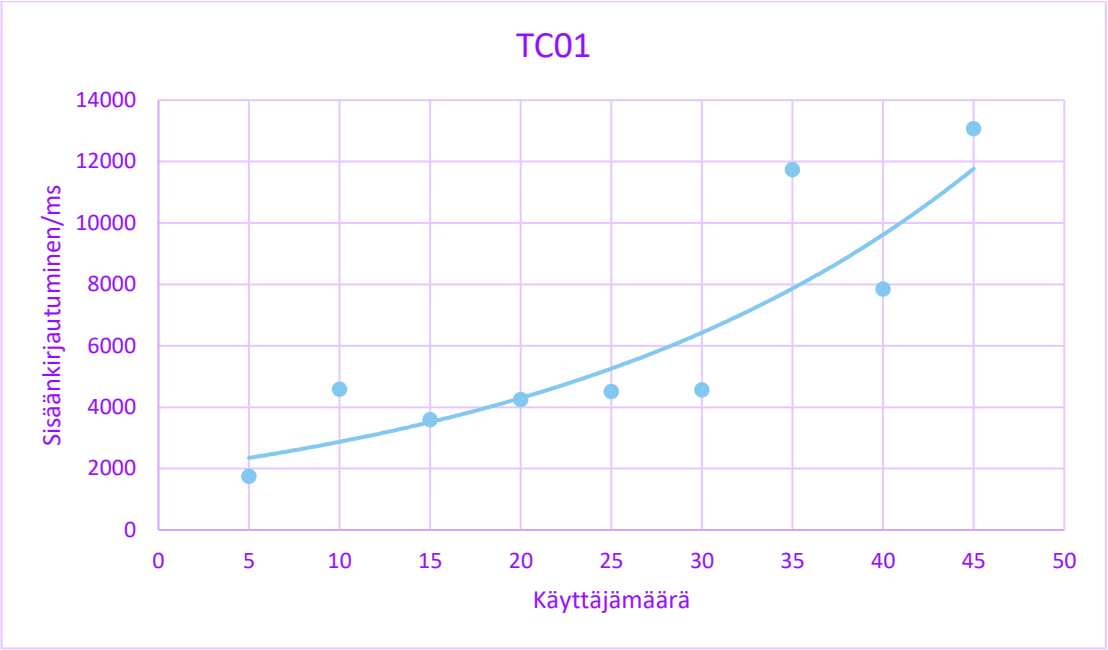
Taulukoiden 2 ja 3 sc ja ic tarkoittavat:

- sc: Tietokoneiden lukumäärä
- ic: Selainten lukumäärä per tietokone

TAULUKKO 2. TC01 kuormitustestauksen tulokset

sc	ic	Yhteensä	Onnistu-neet	Epäon-nistuneet	Kirjautumi-nen/ms	Uloskirjautu-minen/ms
1	5	5	5	0	1752	138
2	5	10	10	0	4586	488
3	5	15	15	0	3595	435
4	5	20	14	7	4261	391
5	5	25	23	2	4520	273
6	5	30	24	7	4571	174
7	5	35	20	15	11 742	117
8	5	40	17	23	7849	119
9	5	45	11	34	13 073	127

Taulukosta havaitaan, että käyttäjämäärän noustessa osa testeistä alkaa epäonnistumaan ja kirjautumiseen kuluu enemmän aikaa. Tuloksien perusteella uloskirjautumiseen käyttäjämäärän nousulla ei ole merkitystä. Kriittinen kipupiste saavutetaan käyttäjämäärän ollessa yli 20, jolloin palvelun käytettävyys alkoi kärsimään, koska testejä alkoi epäonnistumaan.



KUVA 11. Sisäänkirjautumiseen kuluva aika käyttäjämäärän noustessa

Käyttäjämäärän noustessa yli 30 käyttäjään havaitaan, että sisäänkirjautumiseen kuluva aika alkaa kasvamaan huomattavasti.

TAULUKKO 3. TC02 kuormitustestauksen tulokset

sc	ic	Yhteensä	Onnistu-neet	Epäonnis-tuneet	Rajapinta-haku/ms
1	5	5	5	0	1622
2	5	10	10	0	784
3	5	15	15	0	750
4	5	20	15	5	650
5	5	25	23	2	506
6	5	30	14	16	233
7	5	35	17	18	214
8	5	40	18	22	277
9	5	45	10	35	278

Taulukon 3 tuloksista päätellään, että käyttäjämäärän nousu ei vaikuta rajapinta-hakujen nopeuteen. Testien epäonnistumiset liittyivät sisäänkirjautumisproses-siin ja mikäli sisäänkirjautuminen onnistuu, niin tällöin käyttäjämäärällä ei ole vai-kutusta sivuston käytettävyyteen ja rajapinnan nopeuteen.

Tulosten perusteella TC02 käyttäjämäärän kipupiste oli myös yli 20 käyttäjää, jolloin testejä alkoi epäonnistumaan.

6 POHDINTA

Opinnäytetyön tavoitteena oli tutkia ja löytää QAutoCloud-Webpalvelun yhtäaikaisen käyttäjämäärän kipupiste hyödyntämällä RFW-käyttöliittymätestejä suoraan kuormitustestaustyökalulla ja se onnistuttiin löytämään kummassakin testissä käyttäjämäärän ollessa yli 20. Tuloksista päätellään, että kipupisteen ylittyessä osa testeistä alkoi epäonnistumaan ja sisäänkirjautumiseen kuluva aika alkaa kasvamaan huomattavasti, kunnes jollain käyttäjämäärällä ***n*** sivusto ei vastaisi enää ollenkaan. Tulosten perusteella rajapintahakuihin käyttäjämäärän kasvulla ei ole vaikutusta tai se on minimalistinen.

Opinnäytetyön pohjalta saadaan selville perinteisten QAutomate Oy:n pystyttämien virtuaalitietokoneiden, jossa QAutoCloud-Webpalvelu toimii, yhtäaikaisen käyttäjämäärän kipupiste. Tuloksissa pitää kuitenkin ottaa huomioon virtuaalitietokoneiden tekniset tiedot, ja mikäli sivustolla kävisi yhtäaikaisesti enemmän käyttäjiä, pitäisi joko tietokoneiden laitteistotehoa nostaa tai ottaa käyttöön AWS:n tarjoama kuormituksen tasapainottamistoiminnallisuus, jolloin opinnäytetyössä mitatut tulokset eivät enää olisi paikkansapitäviä.

Tuloksia voidaan hyödyntää tulevaisuudessa siten, että mikäli uusi QAutoCloud-Webpalvelu pystytetään ja yhtäaikainen käyttäjämäärä ylittäisi 20 käyttäjää kerralla, tarvitsisi palvelu joko AWS:n tarjoaman kuormituksen tasapainottamistoiminnallisuuden tai tehokkaamman AWS-virtuaalitietokoneen.

LÄHTEET

Molyneaux, Ian. 2014. The art of application performance testing: from strategy to tools. " O'Reilly Media, Inc.".

QAutomate tuotteet. QAutomate Oy. Luettu 24.01.2022
https://qautomate.fi/fi/tuotteet-ja-ratkaisut/

Robot Framework. Luettu 26.05.2022
https://robotframework.org/

Runeson, P. (2006). A survey of unit testing practices. IEEE software, 23(4), 22–29.
DOI: 10.1109/MS.2006.91

Singh, S. K. & Singh, A. 2012. A.Software testing. Vandana Publications

Suorituskyky- ja kuormitustestaus. Sogeti. Luettu 2.12.2021.
https://www.sogeti.fi/palvelut/digitaalinen-laadunvarmistus-testaus/suorituskyky--ja-kuormitustestaus/