Ohjelmistokehityksen teknologioita - *S*eminaarityö

Robot Framework

Tietokannan testaus Robot Frameworkilla

Alex Lindholm

12.5.2024

Sisältö

1 Johdanto/Projektin käynnistys

Seminaarityöni tarkoituksena oli tutustua, johonkin minulle aivan täysin uuteen aiheeseen ja teknologiaan. Päädyin kokeilemaan automatisoitua testausta. Kuten tavoitteessa mainitsinkin, aihe oli minulle siis aivan täysin uusi. Olen vain joskus kuullut sanan automatisoitu testaus, mutta en ole koskaan siihen millään tavalla perehtynyt.

Päädyin toteuttaa automatisoitua testausta Ohjelmistoprojekti 2 ryhmäni PostgreSQL tietokantaan. Hyödynsin tätä, koska tällöin ei minun tarvinnut käyttää ollenkaan aikaa itse tietokannan luontiin.

Ohjelmistoprojekti 2 kurssimme on sovellus, jossa käyttäjä voi hakea liikuntapaikkoja, ja jättää niihin arvosteluja. Tietokanta koostuu siis kahdesta taulusta. appUser, johon tallennetaan käyttäjätietoja ja review, johon tallennetaan arvosteluja. Seminaarityössäni testaan näihin kahteen tauluun lisäämistä, muokkausta ja poistoa.

Työskentelyn vaiheet:

- *Aiheeseen perehtyminen*
- *Teknologioiden valinta*
- *Teknologioiden opiskelu*
- *Testien rakenne*
- *Testien sisältö*
- *Testien suoritus*
- *Raportointi*

1.1 Pohjatyö

Ennen seminaarityön aloitusta, minun oli perehdyttävä testaukseen, sekä Robot Frameworkiin, jotta minulla on tarvittavat taidot työn suorittamiseen.

Robot Frameworkiin tutustumisen suoritin YouTubessa Robot Framework Tutorials nimisellä kanavalla. Kanava on julkaissut kurssin ”Complete Robot Framework Tutorial”, joka antoi minulle tarvittavan ymmärryksen Robot Frameworkista, jotta pystyin aloittaa seminaarin työstämisen.

2 Käytetyt teknologiat

Automatisoidussa testauksessa käytin Robot Framework nimistä teknologiaa. Automatisoitu testaus on minulle aivan uusi aihe ja lähdin internetistä tutkimaan erilaisia mahdollisia tekniikoita, joilla testausta voisi tehdä. Lopulta päädyin Robot Frameworkiin, koska se on yleisesti käytetty, selkeä ja helppo oppia.

2.1 Automatisoidun testauksen työkaluja

Käytä alilukuja ymmärrettävyyden ja luettavuuden takia.

Testauksen toteutukseen käytin PyCharm Community Edition 2024.1 ohjelmaa. Tätä kyseistä ohjelmaa suositeltiin katsomassani YouTube sarjassa, joten päätin ottaa sen käyttöön.

Robot Frameworkin rinnalla käytin robotframework-seleniumlibrarya. Sitä suositeltiin myös katsomassani YouTube sarjassa. Tämän rinnalla käytin myös PyCharmissa IntelliBot nimistä lisäominaisuutta, joka oli pakollinen siihen, että PyCharm tunnisti .robot tiedostot.

Kaiken tämän lisäksi käytin myös jonkin verran pgAdmin4 sovellusta. Sen käyttäminen helpotti tietokannan tutkimista, ja sen varmistamiseksi, että testit todella muuttivat tietokantaa.

3 Toteutus

3.1 Testausprojektin rakenne

Itse seminaarityön toteutus alkoi testiprojektin rakenteen luonnista. Hyvä testi projekti jakautuu useisiin eri kansioihin, jotta rakenne on selkeä ja helposti ymmärrettävä. Itselläni projekti on jaettu 3 pääkansioon. Nämä kansiot ovat Output, Resource ja Tests.

Output kansio sisältää tiedostot, joihin testien tulokset tallennetaan. Nämä tiedostot ovat log.html, output.xml ja report.html. Näistä tiedostoista erityisesti log.html on hyvin tärkeä. Sieltä näkee kaikki testauksen eri vaiheet ja ovatko ne onnistuneet.

Resource kansio koostuu kaikista tiedostoista, jotka haluavat tehdä globaalista nähtäviksi kaikille testitiedostoilla. Itse lisäsin Resource kansioon pelkän tietokantaan yhdistämisen ja yhteyden katkaisemisen. Koin tämän tarpeelliseksi, koska jokaisessa testissä täytyi erikseen aina yhdistää ja katkaista yhteys tietokantaan. Resource tiedoton käyttö nopeutti tätä

paljon, koska nyt pystyin käyttämään vain kahta omaa komentoa tietokantaan yhdistämiseen ja yhteyden katkaisemiseen. Ilman resource tiedostoa olisin jokaiseen tiedostoon joutunut määrittämään tietokantaan liittyviä konfiguraatioita, kuten portin.

Tests kansio taas koostuu kaikista testitiedostoista. Itse jaoin tests kansion vielä kahteen alikansioon. User ja Review. Näiden tarkoituksena on selkeyttää sitä, mitkä testit koskevat tietokannan appUser taulua ja mitkä review taulua.

3.2 Testitiedostot

Jokaisessa testitiedostossa pakollisia kenttiä ovat Settings ja Test Cases.

Settings osia koostuu kaikista mahdollisista kirjastoista, joita projektin suorittamiseen tarvitaan. Esimerkiksi itse käytin *DataBaseLibrary*:a, joka antoi minulle mahdollisuuden kaikkiin erinäisiin komentoihin, jotka on suunniteltu tietokannan testaukseen. Robot Frameworkissa on myös valmiiksi BuiltIn kirjasto, jossa on paljon hyvin perustason komentoja, kuten *LOG*, joka tulostaa aiemmin mainittuun log.html tiedostoon halutun sisällön. Kaikki kirjastot ja niiden tarjoamat toiminnot löytyvät Robot Frameworkin virallisilta nettisivuilta.

Test Cases osio taas koostuu kaikesta testattavasta koodista. Tähänkin on olemassa toinen tapa, jota itse hyödynsin paljon projektissani. On olemassa myös Keywords niminen osio. Tänne voi tallentaa testejä, joita voi sitten Test Cases osiossa kutsua. Tämä rakenne mahdollistaa argumenttien käytön. Argumentit ovat testi tapausten parametreja. Esimerkiksi seminaarityössäni käytän argumentteja arvostelujen muokkauksessa, jolloin test Cases osiossa voi vain kutsua muokkaus keywordia ja antaa sille suoraan uusi arvosana ja teksti. Toinen tapa toteuttaa tämä olisi tehdä Test Case ja muokata siihen aina suoraan arvosana ja teksti.

On olemassa vielä yksi osio. Sen nimi on Variables. Tähän osioon voi tallentaa muuttujia, joita voi kutsua koodissa. Tämän osion käyttö on todella kätevää, jos on jokin testi koodi, jossa pitää käyttää usein samaa arvoa. Jos arvoa tarvitsee muuttaa riittää muuttujan arvon vaihto eikä tarvitse muokata sitä monesta kohdasta koodia. Itse hyödynnän Variables osiota esimerkiksi käyttäjän lisäyksessä, jossa kaikki lisättävän käyttäjän tiedot on tallennettu muuttujiksi.

3.3 Testit

Kuten aiemmin jo mainitsin suoritin lisäys, muokkaus ja poisto testit sekä käyttäjälle sekä arvostelulle. Jokainen testi on suoritettu edellisessä robot testi tiedostossa, jotta jokaista ominaisuutta on helppo testata erikseen.

Käyttäjän lisäyksessä yritetään lisätä tietokantaan käyttäjä. Syötettävät arvot syötetään muuttujista. *Execute SQL String* on komento, jota täytyy käyttää kaikissa SQL komennoissa, joissa halutaan tehdä jonkin laisia muutoksia tietokantaan. Arvostelun lisäys on rakenteeltaan hyvin samanlainen kuin käyttäjänkin lisäys siinä olen myös vain niin sanotusti kovakoodannut osan arvoista niin, että en käytä muuttujia, koska niiden arvojen muuttamista en näe tarpeelliseksi.

Käyttäjän ja arvostelun muokkaus ovat myös hyvin samanlaiset keskenään. Molemmat hyödyntävät Keywords osiota, jotta käyttäjälle ja arvostelulle voidaan antaa muokattavia asioita argumentteina.

Käyttäjän ja arvostelun poisto ovat lähes identtiset. Tämä johtuu siitä, että poisto ominaisuudet toimivat näissä lähes identtisesti.

4 Testien suoritus

Testitiedostojen luomisen jälkeen on aika suorittaa testit. Testin onnistuessa, tulee konsoliin vihreällä tekstillä *PASS*, epäonnistuessa taas punaisella *FAIL*. Suoritettua testiä voi tutkia helposti log tai report.html tiedostoista. Erityisesti jos testi epäonnistuu on hyvin tärkeää, mennä tutkimaan log.html tiedostosta, minkä takia testi epäonnistui. Epäonnistunut testi voi jossain tapauksessa olla merkki onnistuneesta testitiedostosta, mutta toisinaan voi olla, että testin epäonnistuminen voi johtua myös virheellisesti rakennetusta testitiedostosta.

Kun kaikki testit on suoritettu ja testit on rakennettu oikein saadaan selville, että toimiiko testettava asia halutulla tavalla. Esimerkiksi itse sain testausten jälkeen lopputulokseksi, että Ohjelmistoprojekti 2 kurssin tietokanta toimii halutulla tavalla, kaikki testit onnistuivat.

5 Yhteenveto

Suoritin seminaarityöni testauksesta Robot Frameworkilla. Työkalu oli minulle aivan täysin uusi, kuten myös testitiedostojen luominen. Suoritin testit Ohjelmistoprojekti 2 projektin tietokantaan. Opin hyvin paljon uutta, kuten testiprojektin rakenteesta, testi tiedostojen luonnista ja tastausraporteista.

Kaikki testini onnistuivat. Toki suuremmalla ajalla olisi testejä voinut suorittaa myös backend sovelluksen kautta, jotta voidaan varmistaa, että niiden yhteys toimii. Kuitenkin tällä ajalla, ja aiheen ollessa itselleni täysin uusi päädyin siihen lopputulokseen, että minulle on helpompi testata pelkkää tietokantaa.

Kiitos!

Lähdeluettelo

Robot Framework Tutorials ohjevideot:

https://www.youtube.com/playlist?list=PL4GZKvvcjS3u9FP3F9zW4ZbZB8pti2k37

Robot Framework virallinen nettisivu:

https://robotframework.org/