LAMK Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

Päiväkirjaopinnäytetyö:

SEO-raportin luonti ohjelman muutostyö

LAHDEN
AMMATTIKORKEAKOULU
Liiketalous
Tietojenkäsittely
Opinnäytetyö AMK
Kevät 2018
Sasu Santaranta

Lahden ammattikorkeakoulu
Tietojenkäsittely
SANTARANTA, SASU:

Päiväkirjaopinnäytetyö:
SEO-raportin luonti ohjelman muutostyö

Tietojenkäsittelyn opinnäytetyö, 46 sivua

Kevät 2018

TIIVISTELMÄ

Opinnäytetyö on päiväkirjaopinnäytetyönä tehty käytännön työ, jonka etenemisen kirjoitan päiväkirjan tapaan. Opinnäytetyössä kuvataan päiväkohtaisesti sekä työvaihekohtaisesti hakukoneoptimointi-ohjelman muutostyön tekemistä. Opinnäytetyö toteutettiin lahtelaiselle henkilöstö sekä liikevaihto yritys kooltaan pienelle IT-alan yritykselle.

Opinnäytetyön kehitystyön aiheena oli muuntaa olemassa olevaa hakukoneoptimointi-raportin luovaa ohjelmaa, tehden siitä käyttäjäystävällisen ja vapaasti tilattavan. Uuden ohjelman oli tarkoitus olla työn tilaajan kotisivuilla sivustolla vierailijoiden tilattavana. Raportin teksti sisältö tuli myös siirtää tietokantaan raportin muokattavuuden helpottavuuden lisäämiseksi. Raportti käännettiin myös PDF-muotoon sen tulostamisen helpottamiseksi.

Tavoitteenani oli syventää omaa osaamistani ohjelmointi kielessä, erityisesti PHP-kielessä, sekä niiden soveltamisessa käytännössä. Halusin myös perehtyä muihin kyseisen alan työtehtäviin, kuten tietokantoihin ja sen hyödyntämiseen ohjelman teossa.

Työvaiheiden päätteeksi analysoin omaa kehitystäni työvaiheen aikana ja mietin vaihtoehtoisia tapoja tehdä työ. Opinnäytetyön kehitystyön tuloksena valmistui tilattava hakukoneoptimointi raportti, joka on asiakkaalle tilattavissa PDF-muodossa.

Asiasanat: hakukoneoptimointi, SEO, API, PHP, muuttuja, tietokanta

Lahti University of Applied Sciences
Degree Programme in Information Technology
SANTARANTA, SASU

A diary-based thesis:
Modifying an SEO -report program

Bachelor's Thesis in Information Technology, 46 pages

Spring 2018

ABSTRACT

This diary-based thesis reports a practical project. The thesis describes the modification of a search engine optimization program. This is based on daily notes and work phases. The thesis was commissioned by a small IT company based in Lahti, Finland.

The objective on this thesis was to modify an existing search engine optimization reporting program so that it would be user friendly and easy to use. The new program was meant to be on the homepage of the commissioner of the thesis for visitors to order. The text of this report was imported to a database so that the report would be easier to modify. The report was also converted into PDF format so it would be easy to print.

My goal was to deepen my knowledge of programming languages, especially PHP, and how to use them. I also wanted to become more familiar with other work tasks regarding business databases and their use.

After every work phase I analysed my own learning during the work phase and thought alternative ways to make the work. The result of this thesis is search engine optimization report the commissioner's clients can order in PDF format.

Keywords: search engine optimization, SEO, API, variable, database

SISÄLLYS

1 JOHDANTO

Opinnäytetyössäni dokumentoin kuinka tein Lahdessa sijaitsevalle DevNet Oy:lle ohjelman, jonka avulla asiakas voi yrityksen kotisivuilta tilata SEO-raportin, eli hakukoneoptimointi raportin valitsemastaan nettisivusta. Hakukoneoptimoinnilla tarkastetaan ja pyritään parantamaan kotisivujen ja sivustojen hakukone näkyvyyttä.

Kirjoitan päiväkirjan muodossa tavoitteeni, haasteeni ja ratkaisuni työtä tehdessäni. Työhön kuuluu myös viikoittainen raportointi, jossa analysoin tekemääni ja kohtaamiani haasteita sekä pohdin, olisiko minulla ollut vaihtoehtoinen tapa tehdä tekemäni. Viikoittaisen raportoinnin korvaan työvaihekohtaisella raportoinnilla, jotta työn ymmärtäminen olisi helpompaa. Keskeisimmät työtehtävät työssäni ovat ohjelman muokkaaminen uuteen tarkoitukseen, SEO-raportin muuntaminen PDF-muotoon sekä tietokannan luonti ja yhdistys raporttiin. Työ vaatii tuntemusta eri ohjelmointi kielistä, erityisesti HTML sekä PHP. Tämän lisäksi vaaditaan tuntemusta tietokannoista ja niiden käytöstä.

Keskeisiä käsitteitä, joita esiintyy työssäni ovat:

SEO = Search Engine Optimization, eli hakukoneoptimointi kuvaa keinoja, joilla verkkosivun näkyvyyttä hakukoneiden hakutuloksissa pyritään lisätä.

API = Application Programming Interface tarkoittaa ohjelmointirajapintaa, jonka avulla eri ohjelmat vaihtavat tietoa keskenään.

PHP = PHP:hypertext preprocessor on erityisesti webympäristöissä käytetty ohjelmointikieli.

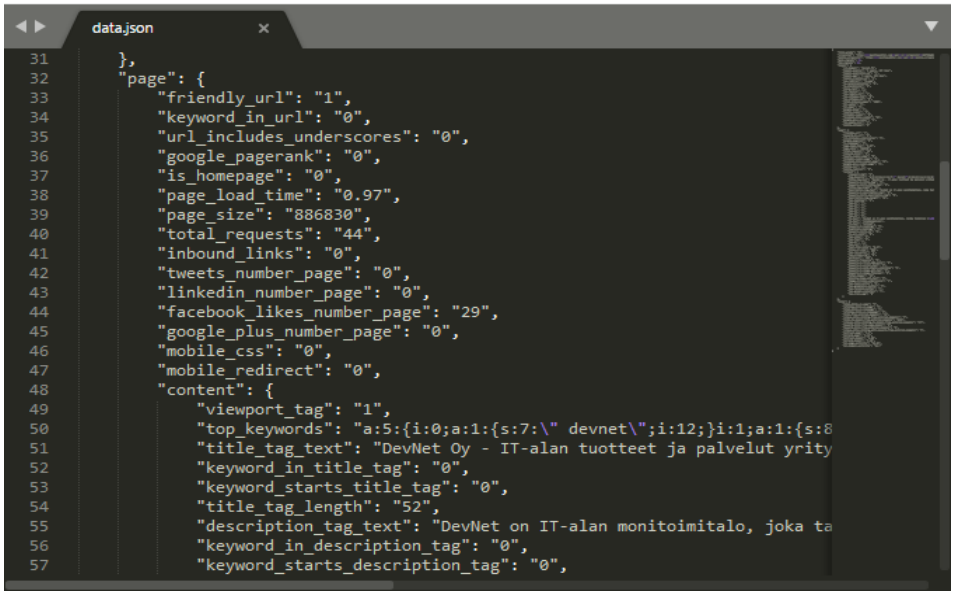
Muuttuja = Muuttujiin säilötään tietoa, joka mahdollistaa pidemmän asian kirjoittamisen lyhyesti muuttujan avulla.

Tietokanta = kokoelma tietoa, jossa säilötään tauluihin, riveihin ja kolumneihin, siten että tietoa on helppo päivittää, muokata ja päästä käsiksi.

1.1 Pohja uudelle ohjelmalle

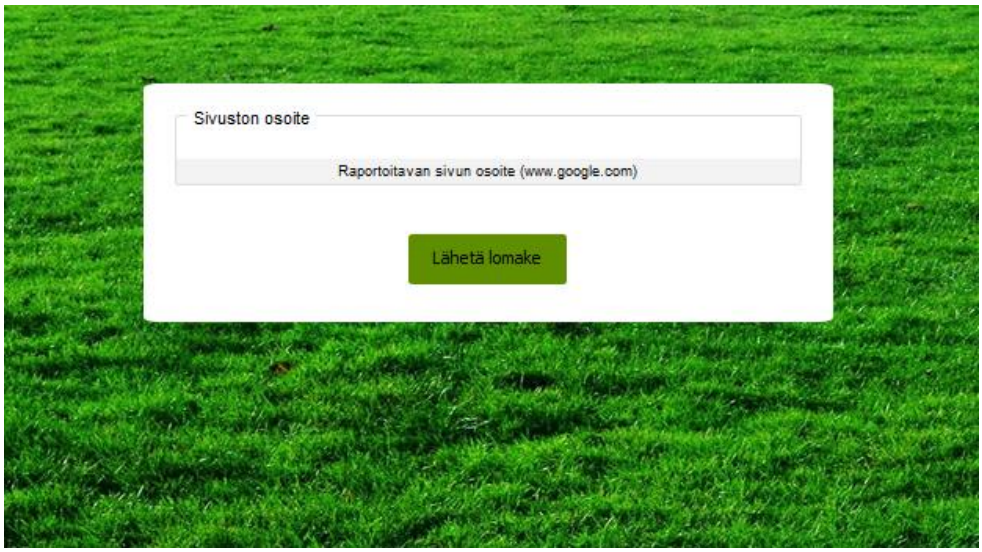
Pohjana työlle käytän syksyllä 2016 harjoittelussa toisen harjoittelijan kanssa yhdessä tekemäämme SEO-raportti ohjelmaa. SEO tulee sanoista ” Search Engine Optimization” eli hakukoneoptimointi. Hakukoneoptimoinnilla tarkoitetaan sivuston näkyvyyttä hakukoneissa kuten googlessa, ja keinoja näkyvyyden parantamiseksi (Gabbert 2017). DevNet Oy:n palveluihin kuuluu SEO-raportin myynti asiakkaille koskien asiakkaiden sivustoa. Raporttien tulokset saatiin selaimeen näkyviin, mutta raporttiin halutut osiot ja tekstit jouduttiin kuitenkin yksitellen kopioimaan DevNet Oy:n omaan raporttipohjaan. Tekemämme SEO-raportti ohjelman tarkoitus oli tehdä SEO-raporttien teko mahdollisimman automaattiseksi. Valmis ohjelma mahdollisti, että täyttämällä raporttiin tulevan sivun nimen, tulosti ohjelma suoraan raportin.

Käytimme työssä APIa eli Application Programming Interfacea, jonka avulla ohjelmat voivat vaihtaa tai pyytää tietoja keskenään (Hughes 2015). API jota käytimme, oli mysiteaudito.com. Valitsimme sen useista eri sivustoista, sillä se antoi kaikki raporttiin halutut osiot. Kyseinen API oli lisäksi ainoa, jossa oli heti käyttövalmis ilmainen demoversio. APIn tehtyä haun sivusta, se luo väliaikaistiedoston, johon se lisää kyseisen nettisivun tiedot ja puutteet (kuva 1). Raportin tulokset haetaan kyseisestä väliaikaistiedosta, ja uuden haun myötä API lisää väliaikaistiedostoon tulokset uudesta hausta. Laskimme myös sivustolle SEO arvosanan perustuen APIn omaan arvonantoon. Laskimme arvon itse, jolloin se perustuin vain raporttiin kuuluviin osiin, sillä APIn oma arvosana perustui kaikkiin APIn hakemiin tietoihin ja suuri osa näistä tiedoista oli omalle raportillemme tarpeetonta. Kaava, jolla laskimme arvosanan, oli ’virheettömät osat / kaikki osat * 100=SEO arvosana’.



Kuva 1. Tuloste tiedoista vastaanotettu nettisivusta

Käytimme ohjelman tekemisessä Sublime Text 3 nimistä tekstieditoria. Pääosin PHP 5, JavaScript, HTML ja CSS ohjelmointikieliä hyödyntäen teimme useita eri tekstitiedostoja. Raportin luonti alkaa ensimmäiseltä sivulta, johon tultaessa ponnahdusikkuna kehotti käyttämään selaimena Google Chromea, koska osa koodin pätkistä ei ollut yhteensopiva Internet Explorerin tai Mozilla Firefoxin kanssa. Myös raportin ulkoasussa ja asettelussa saattoi ilmetä virheitä. Liitimme taustakuvan ja keskelle sivua tekstikentän, johon syötettiin nettisivun URL-osoite, josta raportti halutaan tehdä (kuva 2).

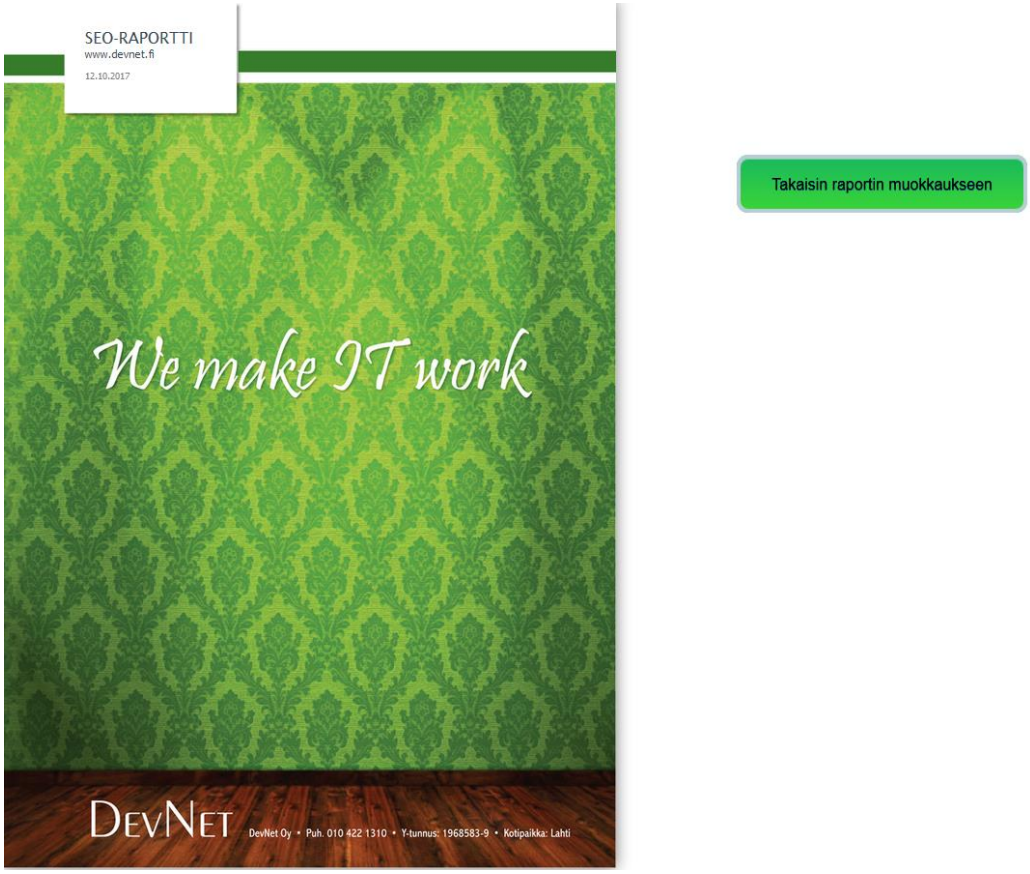


Kuva 2. Raportin aloitussivu

Tämän jälkeen ohjelma haki APIn avulla hakukonetiedot halutusta nettisivusta. Sivu latasi paikallaan tämän jälkeen hetken, että ohjelma ehti päivittää väliaikaistiedostoon tiedot haetusta nettisivusta. Kun ohjelma oli saanut uudet tiedot halutusta sivusta, ohjautuu se sivulle, jossa valitaan raportin tekijä, myyjä ja raporttiin tulevat asiat (Kuva 3). Valittavien osien vasemmalla puolella voidaan valintanappien avulla valita, että halutaanko tiettyä kohtaa raporttiin ollenkaan, sekä kuka on myynyt kyseisen raportin ja tehnyt sen. Valintasivun oikealla puolella taas on osioita, jotka kuuluvat raporttiin, mutta joiden tekstikentän sisältöä on mahdollista muokata raportin laatijan haluamalla tavalla.

Kuva 3. Raporttiin valittavat osuudet ja tekstiosuudet

Kun raporttiin tulevat osiot oli valittu ja muokattu raportintekijän mieleisiksi, painamalla nappia ”Luo raportti” tulostusta vaille valmis raportti ilmestyy seuraavalle sivulle. Raportti on A4 kokoisena sivun vasemmassa laidassa, mikä mahdollistaa sen tulostuksen oikean kokoisena. Raporttipohjan vieressä oikealla on myös ”palaa takaisin” -painike, jonka avulla voi palata raportin muokkaukseen, mikäli raportista halua vielä muokata tai lisätä jotain (kuva 4).



Kuva 4. Valmiin raporttipohjan kansilehti sekä takaisinpaluu -nappi

Valmis raportti on viisisivuinen ja se antaa sivulle arvosanan pisteasteikolla 0-100. Raportti sisältää mallikuvat hyvästä meta-otsikosta ja meta-kuvauksesta, eli hakukoneen antamista tuloksista, sekä haetun nettisivun meta-otsikosta ja meta-kuvauksesta ja kertoo mahdollisista puutteista. Lisäksi raportti kertoo yleisistä puutteista mitä sivustolla on ilmennyt, kuten esimerkiksi otsikoiden puutteesta tai kuvien kuvatekstien puutteista, sekä perustelut puutteille. Raportti myös kertoo useimmin sivulla toistuvat sanat ja muuta. Raportin viimeisellä sivulla on lisäksi kuva sivuston mobiilinäkymästä ja sen mahdollisista puutteista.

2 TYÖN ESITTELY JA LÄHTÖTILANNE

Edellä kuvattu taustoitus on pohjana projektityölleni, jossa olemassa olevaa SEO-raportti ohjelmaa muokataan. Syynä työn muokkaukseen on muutos henkilöstössä, jolloin eniten kyseisiä raportteja myynyt henkilö poistui yrityksestä. Tästä syystä DevNet Oy haluaa muokata raporttia toisenlaiseksi. Tarkoituksena on hyödyntää osaa valmiista ohjelmasta ja muokata sitä. Uusi SEO-raportti tulisi tilattavaksi DevNet Oy:n uusilta kotisivuilta asiakkaille ilmaiseksi. Raportin pystyisi tilaamaan täyttämällä pieneen lomakkeeseen tilaajan sähköpostiosoitteen, johon raportti lähetetään, sekä sivuston, josta kyseinen raportti tehdään. Ohjelman on myös tarkoitus kerätä lomakkeeseen täytetyt tiedot DevNet Oy:n omaan uutiskirjejärjestelmään, jossa tietoja voidaan mahdollisesti hyödyntää esimerkiksi markkinointia varten. Myös raportissa olevien osuuksien tulostusvaihtoehtoja varten luodaan tietokanta, johon tulostusvaihtoehdot lisätään.

2.1 Työtehtävät ja niihin tarvittava osaaminen

Ensimmäinet tehtäväni on muokata palvelimella ohjelman koodia siten, että se soveltuu uuteen käyttöön. Osia koodi tiedostoista voin poistaa kokonaan, mutta joidenkin sisältöä voin hyödyntää. Voin myös sisällyttää muutaman tiedoston kokonaan sellaisenaan. Tällaiset tiedostot ovat kuitenkin pieniä ja vain lähettävät toteutuspyynnön ja toimeksiannon APIlle. Osan tarvittavasta koodista joudun myös itse keksimään ja kokeilemaan. Koodia muokkaan Sublime Text 3 nimisessä tekstieditorissa ja ohjelmointikieliä, joita tarvitsen ovat PHP, CSS sekä HTML. Tämän lisäksi mahdollisesti myös JavaScriptiä ja Ajaxia.

Tämän jälkeen minun tulee luoda pieni lomake, joka kertoo luonnollisesti mitä asia koskee. Lisäksi lomakkeessa täytyy olla kaksi tai kolme tekstikenttää sivuston nimelle, sähköpostille ja mahdollisesti tilaajan nimelle. Minun tulee etsiä koodi, joka lähettää luodun PDF-tiedoston annettuun sähköpostiosoitteeseen siten, että PDF-tiedostoon sisältyy tehty

raporttipohja ja raporttiin kuuluva sisältö. Pyrin hyödyntämään vanhan raportin pohjaa, mikäli mahdollista. Tarvittavat muutokset sisältöön ja raportin ulkoasuun teen kuitenkin vasta työn viimeisenä vaiheena ennen lopullista testausta.

Seuraavana työlistallani on luoda tietokanta raportille. Tietokanta luodaan MySQL-tietokantaan. Tarkoituksena on lisätä kyseiseen tietokantaan kaikki raporttiin kuuluvat osiot, kuten h1 eli otsikot. Kaikille osioille lisätään tulostusvaihtoehdot perustuen APIn antamaan tulokseen nettisivusta. Käyttäen h1 otsikoita esimerkkinä, mikäli APIn antama tulos h1 otsikoille on 0, ovat sivuston h1 otsikot puutteellisia tai niitä ei ole. Täten tietokannan antama vastaus raporttiin olisi luokkaa ”Sivuston h1 otsikot ovat puutteellisia tai ne puuttuvat kokonaan”. Lisäksi loppuun lisättäisiin info, miksi h1 otsikoita olisi hyvä käyttää ”Otsikoiden avulla hakukone ymmärtää sivuston hierarkian paremmin”. Tietokanta luodaan, jotta osioiden ja niiden vastausten muokkaus olisi mahdollisimman helppoa. Aikaisemmin osioiden tulostusten vaihtamiseksi on pitänyt kirjautua palvelimelle ja vaihtaa ne suoraan koodiin.

Tämän jälkeen lisään DevNet Oy:n uutiskirjejärjestelmään osion SEO-raportti ja yhdistän sen ohjelmaan koodilla. Tällöin raportin tilatessa ohjelma lähettää ja kirjaa uutiskirjejärjestelmään tiedot henkilöstä lomakkeeseen täytettyjen tietojen perusteella. Tiedot jotka ohjelma saa ovat sähköposti, sivuston nimi sekä mahdollisesti tilaajan nimi. Saatuja tietoja voidaan mahdollisesti käyttää hyväksi markkinointitarkoituksessa. Tästä syystä lomakkeessa kysytään mahdollisesti tilaajan nimeä.

Tässä vaiheessa työstä on enää jäljellä raportin ja sen ulkoasun viimeistely, tarkempi testaus ja mahdollinen hienosäätö. Myös mahdollisesti pienten muokkausten teko, mikäli niille on ilmaantunut tarve testauksessa. Valmis työ on tarkoitus lisätä DevNet Oy:n tuleville uusille kotisivuille niiden valmistuttua. Valitettavasti uudet kotisivut eivät ole ehtineet valmistua tämän työn valmistuttua. Valmis ohjelma kuitenkin testataan toimivaksi ja tullaan lopulta ottamaan käyttöön, kun kotisivut ovat

valmiit. Muut mahdolliset ohjelmointikielet ja työkalut sekä lähteet, joista koodeja etsin, lisään niiden ilmaannuttua.

2.2 Tietopohja työntekoon

Käydessämme DevNet Oy:n kanssa työnkuvausta läpi, sekä itse tutkiessani sen läpivientiä, uskon työn olevan samaa haastavuustasoa kuin harjoittelussa tekemäni työ. Työ ei siis tule olemaan itsestäänselvyys ja se tuo haastetta. Vanhasta oppimani perusteella minulla on kuitenkin hyvä käsitys siitä, mitä työn valmistuminen vaatii. Koen, että pystyn suoriutumaan työstä sen vaatimalla tasolla. Uskon, että työstä on minulle paljon hyötyä ammatillisessa kehittymisessä. Jo harjoitteluaikanani opin käyttämään useita ohjelmointikieliä ja toteuttamaan eri ohjelmia ja kotisivuja. Kyseinen työ kartuttaa osaamistani vielä lisää asioissa, joihin en ole perehtynyt kovin paljoa, kuten esimerkiksi tietokantoihin ja uutiskirjejärjestelmään. Tästä johtuen on oman kehittymiseni kannalta hienoa, että työssäni pääsen perehtymään enemmän asioihin, jotka ovat vieraampia.

Työtehtävien valmiiksi saamiseksi minun tulee käyttää eri ohjelmointikieliä, kuten HTML, CSS ja erityisesti PHP. Nämä ohjelmointikielet tulivat minulle tutuiksi harjoitteluaikanani. Vaikka kielet ovatkin minulle tuttuja, tulen kuitenkin tarvitsemaan niitä eri tarkoituksissa kuin harjoitteluaikanani. Etenkin etsiessäni keinoa PDF-raportin luomiseen, tulen todennäköisesti tarvitsemaan PHP-kieltä paljon. Minun pitää siis perehtyä vielä tarkemmin kielen toimintaa ja käyttöön. PDF-raportin luonti tulee varmaan olemaan suurin este, joka minun tulee selvittää. Toinen suuri tehtävä tässä työssä on tietokannan luonti, josta minulla on hyvin vähän kokemusta. Siksi minun on perehdyttyvä tarkemmin tietokantojen toimintaan, luontiin ja käyttöön. Tulen myös yhdistämään työn DevNet Oy:n uutiskirjejärjestelmään. En harjoitteluaikanani toiminut uutiskirjejärjestelmän kanssa, joten minulla ei ole entuudestaan tuntemusta kyseisestä järjestelmästä. Minulle kuitenkin kerrottiin tämän

olevan yksinkertainen asia, ja että minua ohjeistettaisiin tämän vaiheen tullessa vastaan.

2.3 Sidosryhmät sekä vuorovaikutus

Tehtävällä työllä on muutamia vaikuttavia sidosryhmiä, jotka ovat sekä sisäisiä että ulkoisia. Sisäisesti vaikuttavia sidosryhmiä ovat DevNet Oy:n johto sekä työntekijät. DevNet Oy:n johto on antanut kyseisen toimeksiannon ja he määrittävät mitä sen tulee sisältää. Työntekijät puolestaan mahdollisesti avustavat ja kertovat, miten jokin asia tulisi tehdä, tai mistä voin tietoa löytää. Kyseiset sidosryhmät vaikuttavat niin työn aikana kuin työn valmistuttua, jolloin he hyödyntävät ja käyttävät työtä ja mahdollisesti käyttävät kerättyjä tietoja hyväkseen markkinoinnissa. Ulkopuolisia sidosryhmiä ovat yksityis- sekä yritysasiakkaat, jotka hyödyntävät ohjelmaa ja sen tarjoamaa palvelua. Myös ulkopuolinen sidosryhmä on mysiteauditor, jonka APIa ohjelma hyödyntää. Sidosryhmiä kuvaa kaavio figure 1.

Sisäisistä sidosryhmistä johdon mielipiteet ovat työn kannalta olennaisia, sillä he ovat työn tilanneet ja työn pitää vastata heidän tarpeitaan. Ulkoisista sidosryhmistä puolestaan työn pitää olla kattava ja antaa paras mahdollinen kuvaus asiakkaan sivustosta, jotta palvelusta olisi heille hyötyä. Asiakkaiden mahdolliset mielipiteet tulevat kuitenkin ilmi vasta työn valmistuttua, joten se vaatisi muutoksia työhön jälkikäteen.

Mahdolliset vuorovaikutustilanteet, joita työtovereiden kanssa käyn ovat, että konsultoin heidän kanssaan, jonkin ongelman ratkaisemiseksi. Johdon kanssa käytävä vuorovaikutus on neuvotteluita ja näkemysten vaihtamisia siitä, kuinka työ toteutetaan ja mitä se pitää sisällään.

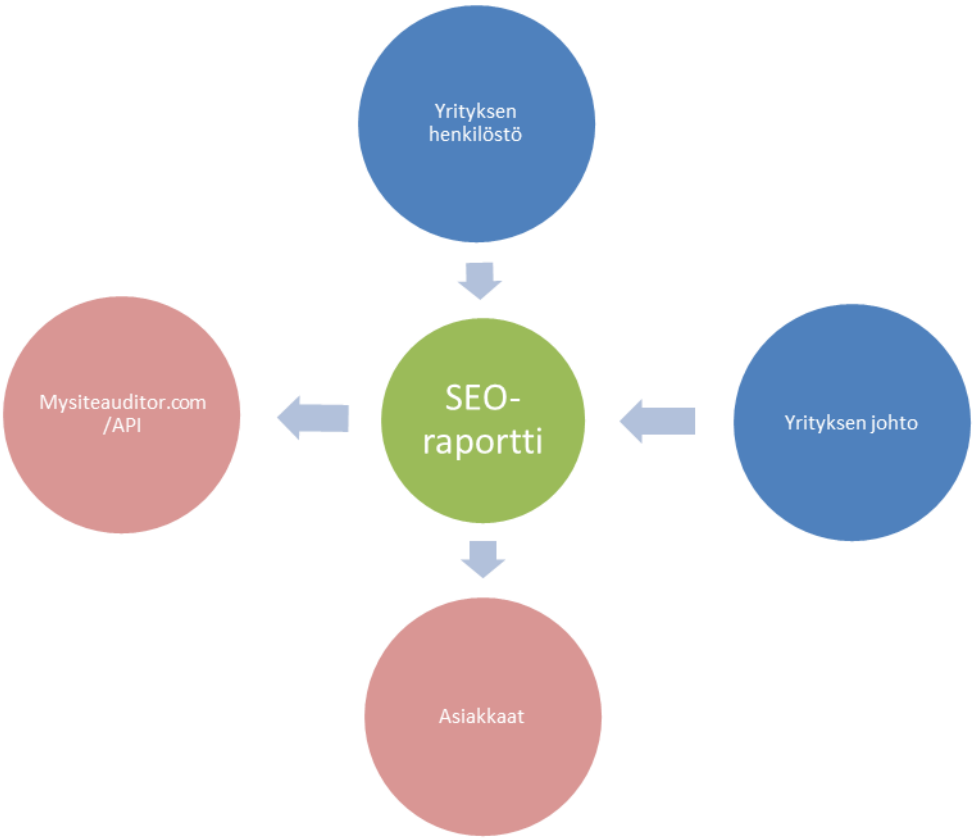


Figure 1 SEO-raportti työn sidosryhmät. Sisäiset sidosryhmät merkitty sinisellä ja ulkoiset punaisella

2.4 Työn tavoitteet

Harjoittelukollegani kanssa teimme vanhan SEO-raportti ohjelman, jotta prosessi olisi mahdollisimman automaattinen, sillä myydyn raportin tekeminen käsin vei valtavan määrän vaivaa ja työaikaa. Raportti oli viisisivuinen ja kaikki tekstit ja kuvat piti yksitellen kopioida selaimella tehdystä raportista. SEO-raportti ohjelman valmistuttua prosessi oli automaattinen ja hyödyllinen yritykselle. Kuitenkin työsuhteen päätyttyä myyjän kanssa, joka myi raportteja eniten, raporttien myynti väheni. Tästä syystä DevNet Oy haluaa muokata ohjelmaa, jotta siitä saataisiin mahdollisimman suuri hyöty yritykselle. Uuden ohjelman tultua DevNet Oy:n kotisivuille, voivat sivulla vierailijat tilata raportin itselleen. Raportti voi saada tilaajan haluamaan oman kotisivunsa parantamista ja ottaa yhteyttä DevNet Oy:hyn asian parantamiseksi. Tästä seuraa positiivista julkisuutta,

etenkin mikäli raportin tilaajat kertovat tästä mahdollisuudesta eteenpäin. Tämä on myös oiva lisäpalvelu, joka erottaa DevNet Oy:n kilpailijoista. DevNet Oy myös kerää raportin tilaajien sähköpostiosoitteet sekä tilatun sivun nimen. Näitä on mahdollista käyttää markkinointitarkoitukseen myöhemmin, sekä tarjota tilaajalle palveluita sivuston kehittämiseen.

Henkilökohtaiset tavoitteeni työn tekemiseen on oman osaamiseni kartuttaminen ja uusien asioiden oppiminen, sekä opittujen asioiden soveltaminen työssä. Haluan tutustua uusiin ohjelmiin ja järjestelmiin, joista olisi minulle hyötyä tulevaisuudessa. Etenkin tietokannoista haluan oppia, sillä en ole perehtynyt niihin kovinkaan paljoa. Toivon myös, että työstäni on oikeasti hyötyä DevNet Oy:lle.

3 OHJELMAN MUOKKAUS

Tässä kappaleessa käyn läpi prosessin, jolla muokkaan olemassa olevaa ohjelmaa uuteen tarkoitukseen. Erittelen mitä kaikkea ohjelman tiedostoista voidaan hyödyntää uudessa ohjelmassa. Mitä pitää muokata uuteen tarkoitukseen ja kuinka paljon tiedostojen sisällöstä poistetaan ja korvataan uudella. Kirjoitan työn etenemisen päiväkirjamuodossa päivittäin raportoimalla ja kuinka saavutan päivälle asettamani tavoitteet. Aina työvaiheen päätteeksi teen myös työvaihekuvauksen, jossa käyn läpi työn etenemisen kuluneen työvaiheen aikana.

3.1 Ohjelman karsiminen

Käytän WinSCP ohjelmaa kirjautuakseni DevNet Oy:n palvelimelle, jossa siirrän vanhan ohjelman kaikkine tiedostoineen uuteen tiedostosijaintiin. Uudessa sijainnissa aloitan ohjelman teon karsimalla vanhoista tiedostoista pois kaiken, mitä en tarvitse työhön. Mikäli uskon, että jostakin voi olla hyötyä työssä myöhemmin, mutta ei kyseisellä hetkellä, kommentoin kyseisen osan koodista piiloon. Tällöin se on tiedoston sisällä, mutta ei vaikuta ohjelmaan. Tiedostoja muokkaan Sublime Text 3 nimisellä tekstieditorilla.

Etenkin raportin sisältöön kuuluvat vastaukset ja raportin ulkoasu tulee pysymään muuttumattomana, mutta tiedostoihin lisätään koodia mahdollistamaan halutut vaatimukset. Kuten esimerkiksi nykyiset vastausvaihtoehdot ovat suoraan ohjelman koodissa, ja tämä muokataan siten, että vastausvaihtoehdot haetaan tietokannasta. Tämän seurauksena nykyisin ohjelmassa oleva raportin kohtien valintasivu tulee tarpeettomaksi, mutta koodissa oleva ulkoisesti näkymätön osa määrittää tulosten perusteella halutut vastaukset itse raportissa. Näkyvän valintasivun poistuttua myös useat CSS tiedostot, joilla muokataan sivun ja kenttien asettelua, jäävät suureksi osaksi tarpeettomiksi ja voidaan poistaa.

3.1.1 23.10.2017

Päivän aikana tavoitteeni on saada tunnukset DevNet Oy:n palvelimelle ja tämän jälkeen saada siirrettyä tiedostot uuteen tiedostosijaintiin, jossa voin alkaa vapaasti muokkaamaan ohjelmaa. WinSCP-ohjelman avulla kirjaudun DevNet Oy:n palvelimelle, missä luon uuden kansion ohjelmalle ja kopioin vanhan ohjelman sinne. Tämän jälkeen voin aloittaa ohjelman ensimmäisten alkuvaiheen tiedostojen muokkaamisen. Näitä alkuvaiheen tiedostoja ovat muun muassa aloitussivu, callback-tiedosto sekä lataussivu. Aloitussivu on ohjelman ensimmäinen sivu, jossa annetaan nettisivun osoite, jota raportti koskee. Callback-tiedosto lähettää hakupyynnön APIin ja hakee kyseisestä nettiosoitteesta tiedot lisäten ne väliaikaistiedostoon. Lataussivu taas on näkyvä väliaikainen sivu, johon ohjelma pysähtyy odottamaan haettujen tulosten päivittymistä väliaikaistiedostoon.

Luotuani uuden tiedostosijainnin ja siirrettyäni tiedostot kyseiseen sijaintiin törmäsin ongelmaan, jonka seurauksena APIin lähettämäni haku ei palannut takaisin, eikä täten päivittänyt raporttia uudella haulla. Ongelman sain korjattua huomattuani, että väliaikaistiedosto oli saanut väärät oikeudet uudessa tiedostosijainnissa. Annoin tiedostolle oikeudet lukea, kirjoittaa ja suorittaa. Muutoksen tehtyäni alkoi ohjelma toimia taas.

Tämän jälkeen aloin tutkia aloitussivua ja sen karsimista. Uudessa ohjelmassa ohjelma sijaitsee DevNet Oy:n kotisivuilla, joten poistin aloitussivulta taustakuvan. Lisäsin myös toisen täytettävän tekstikentän, johon täytetään vastaanottajan sähköposti. Muokkasin myös CSS-tiedostoilla tekstikenttien asettelua, koska uudessa ohjelmassa aloitussivu on vain pieni osa nettisivua. Callback-tiedosto pitää sisällään vain haku pyynnön APIn ja haettujen tietojen lisäämisen väliaikaistiedostoon, joten en tehnyt muita muokkauksia kuin, että päivitin API-avaimen callback-tiedostoon sekä aloitussivulle. Myös uudesta tiedostosijainnista johtuen muokkasin aloitussivulta callback kutsuun oikean tiedostosijainnin, jotta kun hakua lähetetään kohti callback -tiedostoa, se vastaan ottaa sen.

Saavutin päiväkohtaisen tavoitteeni saadessani tunnukset ja siirrettyäni tiedostot uuteen sijaintiin. Ehdin myös tutkimaan ohjelman alkuvaiheen tiedostoja ja tekemään muutoksia niihin. Osaamiseni ei erityisesti kehittynyt päivän aikana, mutta kohdatessani ongelman tiedoston oikeuksissa tajusin, kuinka jotkut tiedostot tarvitsevat eri oikeuksia toimiakseen tietyllä tavalla.

3.1.2 13.2.2018

Aion käydä läpi kaikki tiedostot, joita ohjelmaa varten on tehty. Koska uudessa ohjelmassa on huomattavasti vähemmän sisältöä, voin etenkin CSS-kielellä kirjoitettuja tiedostoja poistaa ja karsia niiden sisältöä. Osa tiedostoista on suurempia kokonaisuuksia, jotka vaikuttavat muun muassa raportin ulkoasuun tai ominaisuuksiin. Muutamat tiedostot ovat kuitenkin yleisesti käytössä eri vaiheissa ja sisältävät valtaosan koko ohjelman sivujen asetteluista. Poistan tai piilotan koodista osat, jotka tulevat tarpeettomiksi. Samoin käyn myös läpi JavaScript tiedostot.

Aloitin tiedostojen läpikäymisen JavaScript tiedostoista. Tiedostot olivat vain muutamia valmiiksi ladattuja kokonaisuuksia. Kahden “bootstrap” tiedoston annoin olla koskemattomana. Bootstrapin avulla voidaan sivustosta tehdä helposti eri laitteisiin mukautuva. Bootstrapin avulla näyttöpääte jaetaan maksimissaan 12 eri palstaan. Sivuston voi jakaa eri lohkoihin mukautumaan niin pieniin kuin isoihin näyttöihin. Tietokoneella katsottuna voi sivulla olla kolme eri tekstipalstaa, ja puhelimella katsottuna sivulla teksti olisi allekkain yhdellä palstalla. Bootstrap on valmis ohjelma, joka on ladattu w3schools.com sivustolta. Myös CSS-tiedostoista osa oli bootstrappia varten ja jäi näin ollen koskemattomaksi. (w3schools 2017.)

Jquery.browser.js nimisen tiedoston poistin kokonaan ohjelmasta. Tiedoston tarkoitus oli tunnistaa, mitä selainta raportin laatija käyttää ja tehdä popup-ilmoitus, jossa tekijälle kerrotaan mikä selain on kyseessä ja mahdollisesti pyydetään vaihtamaan selainta Google Chromeen. Kyseinen tiedosto oli tehty, koska vanhassa ohjelmassa raporttia tulostettaessa

joissakin selaimissa oli ongelmia tulostuksen kanssa. Esimerkiksi Firefoxilla tulostuksen esikatselussa raportin kaikkien sivujen tulosteet olivat päällekkäin ensimmäisellä sivulla. Kyseisen ominaisuuden poistin kuitenkin, koska ohjelma tulee DevNet Oy:n kotisivuille vapaasti tilattavaksi, jolloin popup-ilmoitus olisi vain häiriönä. Lisäksi raportti lähetetään PDF-tiedostona tilaajalle, joten tulostus ei ole välttämätön.

En päässyt päivätavoitteeseeni, sillä kävin läpi vain JavaScript tiedostot. CSS-tiedostot jäivät vielä koskematta ja jäävät seuraavaan kertaan. JavaScript tiedostot sain kuitenkin käytyä läpi ja poistettua uudessa ohjelmassa tarpeettomiksi jäävät tiedostot. Osaamiseni ei varsinaisesti kehittynyt, sillä kävin läpi jo tekemiäni asioita. Koska kyseisten asioiden tekemisestä on kulunut jo aikaa, muistin paremmin mitä tietyt asiat tekevät JavaScript-kielessä ja kuinka kieltä käytetään.

3.1.3 14.2.2018

Tämän päivän aikana aion jatkaa CSS-tiedostojen läpikäymistä. JavaScript tiedostoja ei ollut kuin mainitut kolme. Kaikkiaan kahdeksan CSS-tiedostoa koski bootstrappia, jotka jäävät koskemattomiksi. Jäljelle jäävistä kahta käytetään vain pienen ominaisuuden tekoon, kuten ympyrän kaaren jota käytetään raportin SEO -arvosanan vieressä kuvainnollistamassa, kuinka hyvä SEO -arvosana sivustolla on. Kolme CSS-tiedostoa ovat suurempia kokonaisuuksia, johon on kaikki sivujen asettelut tallennettu. Näiden läpikäynti on päivän tavoitteeni.

Yksi näistä suurista kokonaisuuksista oli selkeästi isompi ja koski raportin ulkoasua. Alkuperäisessä ohjelmassa oli raportin vieressä nappi, jonka avulla saattoi palata edelliselle sivulle muokkaamaan raporttiin valittavia kohtia. Nappi ja sen tyyliasettelut poistetiin, koska uudessa ohjelmassa raportin vastaukset haetaan tietokannasta ja kohtien valinta päätetään ennakkoon, eikä raporttia tehdessä.

Toinen laaja CSS-tiedosto koski raportin sisällön valintasivua. Tämän pystyin poistamaan heti, sillä valinta sivu on ainoa, joka haki asettelut ja

tyylit kyseisestä tiedostosta ja sivu tulee poistumaan ohjelmasta kokonaan. En kuitenkaan poistanut kyseistä tiedostoa heti, sillä valintasivua pidän voimassa siihen asti, että ohjelma on valmis vastausten siirtämiseksi tietokantaan. Ohjelman toimivuus on helpompi testata, kun CSS-asettelut ovat voimassa.

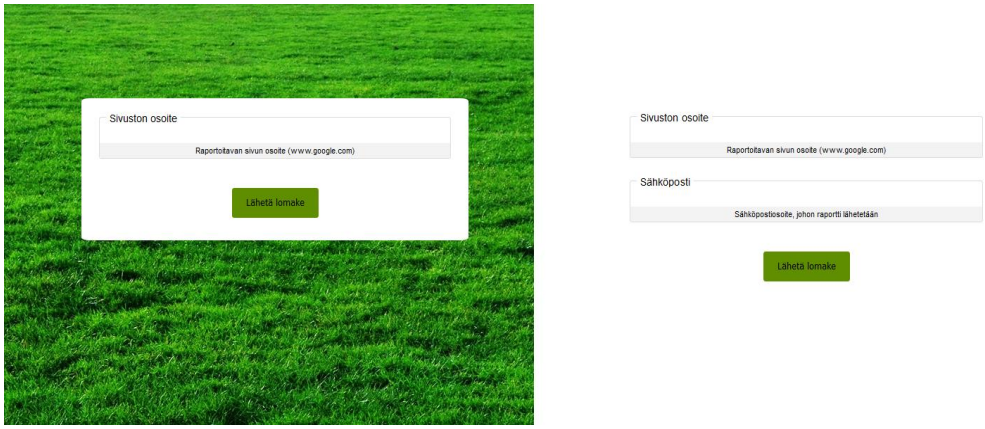
Viimeinen laajempi CSS-tiedosto koski ensimmäistä sivua ja väliaikaista lataussivua. Osa molempien sivujen asetteluista määritetään samannimisillä tunnisteilla, jolloin molemmilla sivuilla voidaan samanaikaisesti määrittää asetteluja. Koska lataussivu jää pois ohjelmasta, poistin tiedostosta kaiken mikä ei vaikuta aloitussivuun mitenkään. Ensimmäisen sivun asetteluita en muuttanut vielä, koska sivu tulee vielä muuttumaan, sillä nyt sivu on koko ruudun kokoinen, johon tekstikentät on keskitetty keskelle. Ohjelman lopullinen sijainti tulee olevaan DevNet Oy:n kotisivuilla todennäköisesti sivussa palstalla, joten tekstin, tekstikenttien ja nappien koko, väri ja sijainti tullaan vielä todennäköisesti muuttamaan.

Päivätavoitteeni saavutin ja sain käytyä loputkin CSS-tiedostot läpi ja poistettua sen mitä en ohjelmaan tarvitse. Osan koodista jätin vielä tallelle ohjelman testaamisen helpottamiseksi. Koen, että osaamiseni kehittyi päivän aikana, sillä kävin suuria tiedosto kokonaisuuksia läpi ja jouduin miettimään, mikä on olennaista ohjelmassa. Tämän myötä tieto- ja taitotasoni kehittyi lähelle samaa tasoa kuin harjoitteluaikanani, jolloin olin tekemisessä CSS-kielen kanssa päivittäin.

3.1.4 14.3.2018

Päivän aikana käyn läpi PHP-tiedostoja samalla tavoin kuin CSS- ja JavaScript-tiedostojen kanssa. Osa tiedostoista pyörii taustalla, eivätkä vaikuta näkyvään osaan sivulla. Joitain tiedostoja ja koodin osuuksia joudun jättämään vielä jäljelle testaamisen helpottamiseksi, mutta pyrin poistamaan heti kaiken, mitä en tarvitse ohjelmassa tai sen testauksessa.

Aloitin PHP-tiedostojen tutkimisen ohjelman aloitussivusta. Poistin sivulta taustakuvan sekä ylimääräiset tekstit ja lisäsin tekstikentän sähköpostille, johon raportti lähetetään (Kuva 5). Vaihdoin myös APIn avaimen, jolla API tunnistaa käyttäjän ja hakee tiedot haetusta sivusta. Lisäksi päivitin tiedon, mistä tiedostosijainnista ohjelma löytää callback -tiedoston, jonka avulla API tunnistaa käyttäjän, hakee tiedot halutusta sivusta sekä päivittää tiedot väliaikaistiedostoon.



Kuva 5. Aloitussivu ennen ja jälkeen muokkausten

Seuraavana tiedostona poistin kokonaan loading.php nimisen lataussivun. Sivun tarkoitus oli toimia väliaikaissivuna, kun API hakee uuden sivun tietoja. Sivulla pyöri vihreä latauspallo joka merkitsi, että ohjelma päivittää sivua, vaikka todellisuudessa kyseisen gif-animaation tarkoitus oli pitää käyttäjä sivulla ja antaa ohjelman siirtyä rauhassa eteenpäin tiedot saatuaan. Kyseiselle sivulle ei enää ollut tarvetta, sillä koko ohjelma tulee pysymään jatkossa paikallaan ja tietojen haku ja valinta tapahtuu taustalla.

Kolmantena tiedostona tutkin raportin osien valintasivua. Sivu ei tule olemaan lopullisessa ohjelmassa, mutta en poistanut sitä vielä, koska ohjelman toimivuuden ja testauksen kannalta sivu on vielä tärkeä. Sivun voin poistaa vasta kun vastaukset on lisätty tietokantaan.

Viimeisenä näkyvänä sivuna on sivu nimeltä action.php, johon raportti tulostautuu. Sivulle en vielä tehnyt isoja muutoksia, koska vastaukset lisätään tietokantaan, jolloin tulostettavat vastaukset voidaan poistaa myös raporttisivulta. Aiemmin poistin CSS-asettelut napista, jolla palattiin

takaisin raportin osien valintaan, ja nyt poistin napin myös näkyvistä kokonaan. Lisäksi poistin raportista kohdat, joilla määritettiin raportin tekijä sekä raportin myyjä. Muut muutokset raporttisivuun teen tietokannan luonnin yhteydessä. Loput PHP-tiedostot toimivat taustalla, eivätkä vaatineet muutoksia, kuin tiedostosijaintien nimien vaihdossa ja tiedon etenemisen määrityksessä.

Saavutin päivälle asettamani tavoitteet käydessäni läpi kaikki PHP-tiedostot. Tekemistä ei tässä osuudessa ollut paljoa, sillä osan tiedostoista jätin vielä jäljelle testauksen helpottamiseksi. Sain kuitenkin poistettua turhaksi jääviä asioita hyvin ja valmisteltua ohjelman valmiiksi PDF-muunnosta varten.

3.1.5 Työvaiheanalyysi

Ensimmäinen työvaihe venyi todella pitkäksi erinäisistä asioista johtuen. Pyrin keskittymään muihin samanaikaisesti käynnissä oleviin kursseihin ja hoitamaan ne pois alta ensisijaisesti. Myös palvelimelle pääsyssä oli ongelmia salasanojen päivittyessä, joka hidasti työn etenemistä ja vaati DevNet Oy:n toimistolle paikanpäälle menemistä asian korjaamiseksi. Henkilökohtaiset menot hidastivat myös työn etenemistä ja yhdessä palvelimelle kirjautumisongelmien kanssa estivät etätyöskentelyä jonkin verran.

Osaamiseni kehitys ei ollut erityisen suurta. Tämä johtuu erityisesti siitä, että en tehnyt paljoakaan uutta ohjelmaan, vaan poistin ja kävin läpi olemassa olevaa. Tehdyt asiat kuitenkin palautuivat mieleen mitä enemmän ohjelmaa kävin läpi. Vaikka uutta en oppinutkaan, on asioiden toiminnan muistaminen todella tärkeää ohjelman seuraaviin vaiheisiin edetessä. Tiedostojen läpikäynti ja tutkiminen palautti mieleeni, kuinka kyseisiä ohjelmointikieliä käytetään ja kuinka ne toimivat.

Työvaiheessa ei ilmennyt suuria ongelmia, mutta suurimpana varmaankin tiedoston oikeuksissa ilmennyt, joka esti ohjelman toiminnan. Ongelman korjasin muuttamalla tiedoston oikeudet huomatessani eriäväisyyden

tiedostojen oikeuksissa, vertaillessani tiedostoja vanhassa sijainnissa sekä tiedostoja, jotka kopioin uuteen sijaintiin. Työvaiheen olennaisin osa oli uudessa tiedostosijainnissa tiedostojen läpikäynti ja niiden tarkastaminen sekä epäolennaisen poistaminen. Vaikka työvaiheessa kesti todella kauan, tein sen oikeastaan ainoalla mahdollisella tavalla. Koska työ vaati tiedostojen tutkimista, oli hyödyllistä käyttää Sublime Text 3 nimistä tekstieditoria, koska se tunnistaa eri ohjelmointikielet ja värjää ne eri värillä. Sublime Text 3 myös värjää saman koodikielen eri osia eri väreillä ohjelman ymmärtämisen helpottamiseksi. Ohjelma myös näyttää missä jokin tunniste, kuten tekstikenttä tai PHP tekstiosuus, on aloitettu ja suljettu, tai mikäli sitä ei suljettu ollenkaan. Kyseinen tekstieditori helpottaa huomattavasti koodin hahmottamista ja ymmärtämistä. En siis usko, että nopeuden lisäksi minulla olisi ollut parempaa tapaa hoitaa työvaihe.

3.2 Raportin muuntaminen PDF muotoon

Työn seuraava vaihe on raportin muuntaminen PDF-muotoon asiakkaalle. Tämä toteutetaan hyödyntämällä PDF-kirjastoa tai APIa, jonka avulla raportti muunnetaan PDF-muotoon palvelimelle. Kun PDF-tiedosto on muunnettu, ohjataan raportin tilaaja juuri muunnetun PDF-tiedoston nettiosoitteeseen eli URL:iin. Tällöin selaimeen aukeaa kyseinen raportti PDF-muodossa.

Tämän osion tekeminen ohjelmaa varten on kaiken haastavin ja aikaa vievin, koska sopivan PDF-kirjaston tai APIn löytäminen ja toimivaksi testaaminen voi olla hyvinkin aikaa vievää ja haastavaa. Ratkaisun löytämisestä haastavan tekee alan kehittyminen koko ajan, jolloin ohjelmat ovat kehittyneet tai taantuneet. Tämä hankaloittaa parhaan ratkaisun löytämistä tällä hetkellä, koska tutkiessa eri artikkeleita tai foorumeja koskien ongelmaani, tulisi tiedon olla melko ajantasaista. Yhdistän päiväkirjadokumentoinnin useamman päivän ajaksi, sillä ratkaisua etsiessä päivät ovat pitkälti samanlaiset testatessani jotain APIa tai PDF-kirjastoa. Käyn kuitenkin jokaisen testaamani APIn sekä PDF-kirjaston erikseen läpi ja määritän foorumit sekä artikkelit, joista ratkaisua etsin.

3.2.1 5.-6.4.2018 & 10.-12.4.2018 & 17.-18.4.2018

Seuraavien päivien päivätavoitteenani on löytää ratkaisu, jolla saan muunnettua raportin PDF-muotoon. Päivien aikana itse työn eteneminen saattaa olla hyvinkin hidasta, mikäli toimivaa ratkaisua ei löydy.

Tutkin erilaisia PDF -kirjastoja ja APIa etsien toimivaa ratkaisua. Osa lupaavilta näyttävistä vaihtoehdoista olivat maksullisia, kuten “pdfcrowd” ja “phptopdf”, jotka hyödynsivät APIa. Toiset taas olivat valtavia tiedostokokonaisuuksia ja täten turhan vaikea käyttöisiä sekä hitaita, kuten “wkhtmltopdf”. Tutkin myös “TCPDF” ja “mPDF” nimisiä ohjelmia.

Wkhtmltopdf vaikutti lupaavalta ja ohjelman kotisivuilla väitettiin ohjelman muuntavan HTML- tai PHP-tiedoston PDF-muotoon. Ohjelman käyttöohjeista kuitenkin paljastui, että luotu tiedosto piti syöttää tietokoneella ohjelmaan, joka muodostaa PDF-tiedoston. Myös ohjelman dokumentaatio oli todella huonoa, joten en voinut tarkistaa olisiko ohjelma ollut kuitenkin käyttökelpoinen. (wkhtmltopdf 2018.)

MPDF:n kotisivut olivat myös todella sekavat ja epäilyttävän oloiset. GitHubista löytyi eniten tietoa ohjelman käytöstä ja sieltä ohjelma oli myös ladattavissa. Kuten wkhtmltopdf ohjelman kanssa, oli ohjelman dokumentaatio kehnoa ja epäselkeää. Kyseisen ohjelman lopulta hylkäsin tarkastelusta aika nopeasti. (github 2018.)

TCPDF:stä oli ohjelman kotisivuilla puolestaan kattavasti tietoa ohjelman käytöstä. Myös esimerkkejä oli paljon sekä esimerkki kuva, kuinka tietty osa koodia vaikuttaa raportin ulkoasuun. Otin ohjelman testaukseen, jolloin huomasin ohjelman olevan todella suuri. Ohjelma piti sisällään todella paljon tiedostoja ja niiden käyttö sekä tarpeellisuus eivät minulle selvinneet. Minulle myös paljastui, että ohjelmalla piti luoda tiedosto. Tämä olisi tarkoittanut koko raporttisivun uudelleen kirjoitusta, eikä takeita toimivuudesta ollut. Tästä syystä hylkäsin myös tämän vaihtoehdon. (tcpdf 2018.)

Tutkin stackoverflow.com nimistä foorumia, joissa käsitellään ohjelmointia ja ongelmia, joihin käyttäjät törmäävät, pyytäen apua ongelmiin (stackoverflow 2018). Toinen sivusto, jossa ihmiset vastaavat ohjelmoinnin kysymyksiin oli quora.com. Yksi näistä kysymyksistä oli saanut kattavan vastauksen henkilöltä Milenko Vlajnic, jossa hän käy läpi eri PDF-kirjastoja, joilla muunnoksen voisi tehdä ja vertailee niiden onnistumista muunnettavan tiedoston monimutkaisuuteen (Vlajnic 2016).

En päässyt päivien aikana päivätavoitteeseeni, sillä en löytänyt keinoa PDF-muunnoksen tekemiseen. Tutkin paljon PHP:ta ja sen toimintaa ja opin ainakin tunnistamaan eri PDF-muunnos työkalujen toimintaa. Tästä saattaa olla hyötyä ratkaisua etsiessäni, tai mahdollisesti tulevaisuudessa, mikäli kohtaa vastaavanlaisen ongelman.

3.2.2 19.4.2018

Jatkan keinon etsimistä, jonka avulla saan muunnettua raportin PDF-muotoon. Tutkin samoja foorumeita, sekä artikkeleita etsien mahdollisimman uutta ilmoitusta, jossa pohditaan samantapaista ongelmaa kuin omani. Täten saan tietää, mikä PDF-kirjasto tai API on tuottanut onnistuneita tuloksia viime aikoina, eikä edellyttäisi raportin täysin uudeksi kirjoittamista. Pyrin etsimään ratkaisua, jolla voin muuntaa raportin PDF-muotoon perustuen raportin tiedostoon tai URL:iin. Tämä mahdollistaisi sen, että itse raporttia ei tarvitsisi muokata muunnosta tehdessä.

Löysin sivulta phptopdf.com APIn, joka vaikutti yksinkertaiselta käyttää ja tulostaisi PDF-tiedostona raportin perustuen sivun URL:iin. Sivusto on maksullinen, mutta siinä on ilmainen kokeilujakso rajallisella määrällä raportin tulostuksia. Määrän ylittyessä voin tehdä uuden tunnuksen APIn sivuilla, jolloin saan uudet tulostus kerrat. Latasin ohjelman ja siirsin sen DevNet Oy:n palvelimelle testatakseni sitä. Aluksi yritin saada ohjelman tallentamaan palvelimelle PDF-tiedoston raportista, johon käyttäjä olisi sitten ohjattu. Tässä en kuitenkaan onnistunut, joten tyydyin yrittämään

muuntamaan erillisen tiedoston PDF-muotoon tiedoston nettiosoitteeseen siirryttäessä. Tässä onnistuin lopulta huomatessani, että minun täytyi antaa tiedostolle oikeudet kirjoittaa, lukea ja suorittaa. Ratkaisu oli sama kuin mihin törmäsin siirtäessäni ohjelmaa uuteen tiedostosijaintiin.

APIn mukana tulleessa tiedostossa oli valmiina koodattuna ohjelman asetukset ja eteneminen. Minun piti luoda uusi tiedosto ja yhdistää se APIn tiedostoon ja pystyin lisäämään asetukset tulevalle sivulle. Asetuksilla määritin hakevani nettiosoitteesta kohteen PDF-tiedoston näkymälle sekä osoitteen, josta raportti tulostetaan. Lisäksi muunsin hieman PDF-sivun kokoa, sillä kun selaimessa ollut raportti muuntui PDF-muotoon, sen koko muuttui hieman ja osa sivun footer-osasta eli alatunnisteesta siirtyi seuraavan sivun alkuun.

Tämän jälkeen koitin saada aloitussivun raportin siirtymään PDF-sivun osoitteeseen, kun käyttäjä oli hakenut sivua ja kyseisen sivun tiedot oli päivitetty. Tämän pyrin tekemään samalla tavalla kuin vanhassa raporttiohjelmassa, jossa sivun osoitteen annettu ohjelma siirtyi lataussivulle, jossa se odotti, kunnes tiedot oli päivitetty. En onnistunut tekemään siirtymää samalla tavalla, joten tyydyin laittamaan sivun “uneen” sleep-komennolla neljäksikymmeneksi sekunniksi, antaen APIlle aikaa hakea sivun tiedot ja päivittää ne, jonka jälkeen ohjelma eteni PDF näkymään raportista. Huomasin kuitenkin, ettei raporttiin tullut mukaan vastauksia SEO hakutuloksiin ilman, että ohjelma kulki valintasivun kautta. Koska valintasivu poistuu käytöstä, on minun seuraavaksi keksittävä ratkaisu saada raportin tiedot sisältymään itse raporttiin.

Toinen hyvin samankaltainen API, jota tutkin juuri ennen löytynyttä phptopdf APIa oli pdfcrowd. APIssa oli myös ilmainen kokeilujakso, jonka jälkeen se siirtyy maksulliseksi. Dokumentaatioon ei kuitenkaan kunnolla päässyt käsiksi ennen kuin oli luonut tunnukset. Tätä APIa en käytä ohjelmassa, mutta tiedän ainakin varavaihtoehdon olevan valmiina, mikäli phptopdf APIn kanssa tapahtuu jotain odottamatonta, jonka seurauksena APIa ei voi käyttää. (pdfcrowd 2018.)

Saavutin kuitenkin lopulta päivätavoitteeni löytämällä toimivan ratkaisun tarpeisiini muuntaessa raporttia PDF-muotoon. Sain testattua ohjelman toimivuuden sekä etenemään raporttiin. Raportin sisältö ei kuitenkaan tulostunut ilman etenemistä valintasivun kautta. Tämän ongelman korjaamista jatkan seuraavana päivänä. Osaamiseni kehittyi paljon päivän aikana. Entuudestaan tiesin, kuinka API toimii ja käyttäytyy. Sain kuitenkin yhdistää ja tarkkailla tiedon etenemisen aina raporttiin asti. Muokkasin myös hieman PDF tulostuksen ulkoasua vastaamaan haluttua raporttia, joka oli minulle uutta. Ohjelman kotisivujen dokumentaatiosta oli apua työtä tehdessäni ja opin sen avulla APIn toimintaa.

3.2.3 Työvaihe analyysi

Toinen työvaihe sujui huomattavasti ensimmäistä nopeammin saatuani muut kouluasiat alta pois, sekä kiireen lisääntyessä työn valmistumiseen. En ollut ennen tekemisissä PDF-tiedostojen muuntamisen kanssa, joten asia oli kovin uutta minulle.

Kävin läpi useita eri foorumeita sekä artikkeleita etsiessäni ratkaisua. Tästä opin, että PDF muunnoksen tekemiseen on olemassa useita eri vaihtoehtoja. Opin myös, että mahdollisimman uuden artikkelin tai foorumi ketjun tutkiminen tuo yleensä varmimmin toimivan ratkaisun, johtuen teknologian ja sovellusten kehittymisestä. Esimerkiksi tutkimaani mPDF-ohjelmaa oli kehuttu paljon aiempina vuosina, mutta viime vuosina ei ohjelman toiminta ole ollut yhtä varmaa. Voin mahdollisesti hyödyntää oppimaani samankaltaisissa tehtävissä vastaavaisuudessa. Ratkaisun löytäminen PDF-muunnokseen oli hankala, koska SEO-ohjelma määritti tietyt ehdot, johon ratkaisun piti sopia. Näitä ehtoja oli muun muassa se, että tulostettavat tiedot haetaan APIn avulla ja tiedot tulostetaan muuttujien avulla toisesta tiedostosta. Tämän vuoksi osa ohjelmista eivät olleet kelvollisia, koska ne vaativat PDF-sivun kirjoittamista aivan alusta. Koska raportti -sivussa on yli tuhat riviä koodia, olisi raportin uudelleen kirjoitus vienyt todella kauan, eikä takeita toimivuudesta olisi ollenkaan. Myös virheellisen kohdan etsiminen olisi lähes mahdotonta. Toinen haaste

ohjelman löytämisessä oli, että tulostettavat kohdat vaihtuvat jokaiseen raporttiin. Tämän vuoksi päädyin ratkaisuun, jossa käytän toista APIa muuntaessa raporttia. API pystyi hakemaan tiedot verkkotunnuksen eli URL:in perusteella tai tiedoston perusteella. Päädyin valitsemaan URL:in, koska se on varmempi toimivuudellaan, koska se tulostaa raportin ulkoasun perusteella samanlaisen PDF-sivun.

Opin työvaiheen aikana tunnistamaan ohjelman vaatimukset ratkaisua etsiessä. Edellisessä kappaleessa mainitsemani haasteet PDF-muunnos tavan valinnassa tunnistin SEO-raportti ohjelman määrittelemäksi vaatimukseksi työvaiheen onnistumiseksi. Opittuani tämän pystyin rajaamaan vaihtoehtoja ratkaisun löytämiseksi ja lopulta löysinkin toimivan ratkaisun. Toteutus ei täysin vastaa täysin alun perin suunniteltua, sillä alkuperäisessä suunnitelmassa raportti oli tarkoitus lähettää tilaajan sähköpostiin ja tällä ratkaisulla raportti vain aukeaa selaimeen PDF-muodossa. Minulle on kuitenkin annettu vapaat kädet työtä tehdessä, jonka myötä voin poiketa aiemmin suunnitellusta, pyrkien samalla toteuttaa työtä parhaaksi katsomallani tavalla. Suurempia ongelmia en kokenut löydettyäni toimivan ratkaisun.

Vaihtoehtoisesti raportin muunnoksen olisi voinut tehdä muilla PDF-kirjastoilla, jotka hylkäsin. Tähän kyseiseen työhön keinoni muuntaa PDF-raportti APIn tulostamalla raportti perustuen sivun URL:iin on kuitenkin yksinkertaisin ratkaisu edellä mainitsemistani ehdoista johtuen, jotka kyseinen työ antaa. Muut ratkaisut olisivat vaatineet enemmän työtä, mutta olisivat olleet toiminnassaan varmempia, sillä ne eivät ole riippuvaisia mistään toisesta ohjelmasta, vaan vain omasta sisällöstään. Mikäli samankaltaisia töitä tulee itselleni vastaan jatkossa, osaan tunnistaa vaatimukset, jotka kyseinen työ antaa, sekä valita alustavasti parhaimman vaihtoehdon työn toteutukseen.

3.3 Tulostuksen korjaus

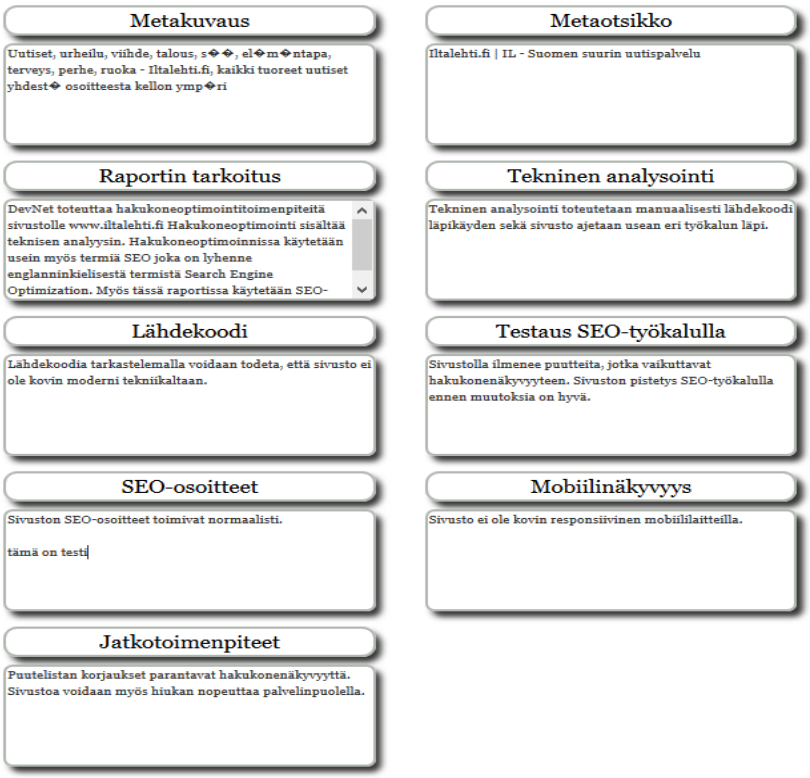
Saatuani raportin PDF-muotoon alan korjata ilmennyttä ongelmaa raportin sisällön tulostuksessa. Seuraava tavoitteeni on korjata ongelma ja siistiä ohjelma kokonaan. Poistan myös turhaksi jäävät sivut, joita olen pitänyt testauksen apuna, kuten valintasivun. Tavoite on saada raportti tulostumaan sekä HTML-sivulle että PDF-sivulle. Raportin sisällön tulos kuitenkin tulee olemaa osittain väliaikainen ratkaisu, koska raportissa tulostettavat vastaukset tullaan lisäämään tämän osion jälkeen tietokantaan.

3.3.1 24.4.2018

Aion käydä läpi tiedon kulun valintasivun sekä raporttisivun välillä ja katsoa, kuinka tietojen haku väliaikaistiedostosta on linkitetty sivuihin. Tarkastan mahdollisuuksia, kuinka saada raportin tiedot helpoiten tulostumaan. Tutkin onko mahdollista säilyttää valintasivu ohjelman taustalla saaden tiedot raporttiin käymättä kyseisellä sivulla. Varmempi keino on kuitenkin käydä läpi tiedon kulun linkitys ja hakea tiedot ja tulostaa ne vasta itse raportissa.

Käydessäni läpi tiedon kulkua huomasin, että kaikki raportin vastaukset haettiin vasta raporttisivulla. Valintasivulla oli kuitenkin yhdeksän osiota, joihin käyttäjä voi itse kirjoittaa vastauksia. Nämä osiot luotiin valintasivulla eikä raporttisivulla kuten muut osiot. Kohtia oli muun muassa “raportin tarkoitus”, “tekninen analysointi” sekä “jatkotoimenpiteen” (kuva 6). Kaikissa kohdissa on valmis tuloste, joista osa haetaan sivun tiedoista, kuten sivuston nimi. Tämän lisäksi kyseisiä kohtia voi muokata ja lisätä

haluamiansa lisäyksiä.



Kuva 6. Vapaasti muokattavat vastaukset raportissa

Raportin vastaukset, jotka haetaan vasta raporttisivulla, oli haettu väliaikaistiedostosta seuraavalla tavalla ja tallennettu muuttujaksi:

```
$metakuvaus1 = $json['page']['content']['description_tag_text'];
```

Tällä tavoin oli haettu kaikki vastaukset, joilla mitataan hakukonenäkyvyyttä. Tämän jälkeen kyseiset muuttujat, kuten esimerkki tapaus ”$metakuvaus1”, muunnettiin $_SESSION muuttujiksi, jotta muuttujilla pystyi vaikuttamaan toiseen sivuun. Muunnosta tehtäessä olemassa oleva muuttuja oli muutettu muotoon:

```
$metaotsikko1 = $_SESSION['title1'];
```

Raporttisivu ei pystynyt hakemaan tietoja valintasivulta $_SESSION muuttujien avulla, koska valintasivua ei enää käytetä. Raportin tietojen pitäisi tulostua käyttämällä alkuperäisiä muuttujia, kuten esimerkiksi $metaotsikko1.

Raportin osat, joiden tekstikenttiä pystyi muokkaamaan, oli teksti lisätty HTML -kirjoituksella. Tästä syystä kopioimalla kyseiset vastaustekstit ja lisäämällä ne raporttisivulle, pitäisi niiden näkyä normaalisti. Osassa tekstejä tulostettiin kuittenkin PHP:n avulla muuttujia, joten tekstin oikein tulostus pitää tarkastaa muutoksia tehdessäni.

Päiväkohtaisen tavoitteeni saavutin selvittäessäni syyn tulostus ongelmaani. Vielä en kuitenkaan tehnyt muutoksia, vaan jatkan asiasta seuraavalla kerralla. Ratkaisun toimivuutta en vielä osaa sanoa, mutta pidän onnistumista varmana, sillä testasin yhden hakukohdan tulostusta ja tämä toimi.

3.3.2 25.4.2018

Tavoitteenani on muokata kaikki raportin tulostus kohdat, jotta tulostus toimisi sekä raporttisivulle sekä PDFsivulla ilman valintasivua. Tämän jälkeen valintasivun voi poistaa täysin. Testattuani eilen onnistuneesti yhdellä muuttujalla, uskon tulostuksen toimivan muutettuani kaikki muuttujat takaisin alkuperäisiin. Muokattavat tekstikentät kirjoitan vain raporttiin omiin kohtiinsa HTML-koodilla, jolloin teksti tulostuu. Kyseisiä tekstikenttiä ei kuitenkaan voi tämän jälkeen itse vapaasti muokata.

Muokattuani kaikki kohdat huomasin, että se vastasi täydellisesti raporttia, joka tulostettaisiin valintasivun kautta tehtynä. Raportti sivun muuttujat oli haettu väliaikaistiedostosta ja nimitetty muuttajaksi kuten edellä kävinkin läpi. Korjattuani tulostettavan osuuden muuttujan nimeksi oikean muuttujan, alkoi raportin tulostus toimia. Useassa kohdassa riitti, että vaihtoi muuttujan nimen. Valtaosa tulostettavista kohdista oli tulostettu raporttiin komennolla:

```
<? php if($_SESSION['desc'] === 'true') {echo $metadesc;}?>
```

Tällöin tulostus tehtiin $_SESSION muuttujalla, mutta tulostuksen sain toimimaan, muutettuani tulostus tavan muotoon:

<? php echo $metadesc; ?>

Tämän saman tein kaikille tulostettaville muuttujille, jonka jälkeen tulostus toimi moitteetta.

Tämän jälkeen muokkasin vapaasti valittavien kenttien tulostuksen. Tämä oli tehty kirjoittamalla suoraan tekstisisältönä haluttu lause raporttiin. Koittaessani tulostaa kyseisiä kohtia muuttujalla, ei vastaus tulostunut. Päätin siis kopioida saman tekstin kuin valintasivulla on oletuksena ja lisätä sen teksti tulosteena HTML-koodilla suoraan sivulle. Ratkaisu oli toimiva ja sain raportin tulostettua kokonaisuudessaan. Aiemmin vapaasti muokattavia raportin kohtia ei enää pysty muokkaamaan raporttia tehdessä, mutta tästä ei ole haittaa, koska vastaukset ollaan kuitenkin lisäämässä tietokantaan.

Odottamaton ongelma tulostuksessa tuli vastaan pyrkiessäni tulostaa sivustolla toistuvien sanojen listaa. API kerää viisi yleisintä sanaa, jotka toistuvat ja kerää ne luetteloksi merkittynä pienellä pallolla •. Itse raportti sivulla merkki toimi, mutta PDF-muotoon muunnettaessa API, joka tekee muunnoksen ei tunnistanut merkkiä. Päädyin poistamaan listan edestä tuon merkin, jolloin toistuvat sanat tulevat allekkain ilman listauksen merkkiä.

Päivän lopuksi poistin loading.php sekä selection.php sivut, jotka olivat jääneet tarpeettomiksi jo aiemmin. Nyt kun raportti muuntuu PDF-muotoon ja tulostus tulee oikeanlaisena ei sivuilla ole enää tarvetta edes testauksessa. Raportti on nyt lopullisessa muodossa, lukuun ottamatta mahdollisia muutoksia, joita mahdollisesti vielä edessä.

Saavutin päivälle asettamani tavoitteet saadessani tulostuksen sivulle toimimaan. Poistin myös turhaksi jääneet sivut, kuten suunnittelin. Seuraava vaihe ohjelman etenemisessä on tietokannan luonti, raportin vastaus tekstien lisääminen tietokantaan sekä tietokannan yhdistäminen ohjelmaan, jolloin vastaukset saadaan haettua.

3.3.3 Työvaihe analyysi

Kyseisen työvaiheen sain suoritettua onnistuneesti hyvin nopeasti. Pää syy tähän oli, että työvaihe oli korjaustyötä juuri ilmenneeseen ongelmaan.

Koen oppineeni tätä työtä tehdessäni, sillä välittömästi ongelman huomatessani, tajusin välittömästi syyn. Osasin tunnistaa viaksi muuttujilla tehtävän tiedonsiirron tiedostosta toiseen. Oletin aluksi kuitenkin virheellisesti, että muuttujat haetaan valintasivun yhteydessä ja lähetetään raporttisivulle tämän jälkeen. Viaksi onnistuin löytämään kuitenkin muuttujien muuntamisen toiseen muuttuja tyyppiin. Muuttujat, jotka haettiin raporttisivulla, oli muunnettu session- sekä post-muuttujiksi tiedon välittämiseksi tiedostojen välillä. Keksittyäni tämän tajusin, ettei raporttisivu saa tietoja takaisin käymättä valintasivulla ja täten ei tulostanut tekstejä.

Kohtasin yhden ongelman työvaiheen aikana. Yksi raportissa tulostettava kohta tulosti luettelona toistuvimmat sanat haetusta nettisivusta. Luettelon kohdat oli merkitty erikoismerkkeillä ja PDF muokkauksen tekevä API ei tunnistanut kyseistä merkkiä ja tulosti luettelon kysymysmerkeillä. Luettelon olisin voinut saada toimimaan selvittäessäni, mitä merkistöä käytetään APIssa, ja muokata erikoismerkit oikean merkistön vaatimiksi. Päädyin kuitenkin tyytymään poistamaan nämä erikoismerkit kokonaan, jolloin toistuvat sanat tulostuvat luettelona ilman etumerkkejä.

Koen työvaiheen aikana ongelmanratkaisu kykyni kehittyneen. Osaan mielestäni tunnistaa virheet helpommin ja keksiä ratkaisut niihin. Myös aina ongelman ratkaistessani koen osaamiseni kehittyvän ja oppivani miten asia toimii.

Tämä työvaihe sisälsi pääosin tekstieditorissa koodin muokkausta, joten eri tapoja tehdä työvaihe ei juuri ole, muuten kuin vaihtamalla tekstieditoria. Eri vaihtoehtoja työvaiheen ongelmien suorittamiseen on kuitenkin. Raportin tulostamisen kuitenkin tein yksinkertaisimmalla mahdollisella tavalla tulostaen alkuperäiset muuttujat, joilla tieto on ensin haettu, ja mihin vastaus on tallennettu. Vaihtoehtoisia tapoja en etsinyt

ollenkaan, koska tapani oli yksinkertaisin ja varmin, koska muuttujien avulla APIlla haetaan sivun tulokset väliaikaistiedostosta ja vastausvaihtoehdot oli jo tallennettu muuttujiksi. Tästä syystä oli loogisinta tulostaa raportin tekstikentät muuttujilla, jotka oli luotu jo valmiiksi. Erikoismerkkien tulostus ongelmassa menin kuitenkin kohdasta, josta aita oli matalin, eli tyydyin vain poistamaan vialliset merkit. Kyseiset merkit saisi kuitenkin tulostettua, selvitettyä PDF APIn käyttämän merkistön ja hakemalla erikoismerkin tulostustavan kyseisellä merkistöllä. Tämä ei välttämättä toimi, mikäli kyseinen merkistö ei ole yhteen sopiva PHP-tiedoston merkistön kanssa. Tässä tapauksessa PDF APIn merkistöä on muutettava itse. Tämän myötä luettelo tulostuisi pienillä palloilla, joka viittaa kyseessä olevan luettelo. Tämä olisi kaunein tapa raportin kannalta, mutta tyydyin tekemään nopeimman mahdollisen tavan. Jatkossa samankaltaisessa tilanteessa, mikäli en ole tiukassa aikamääreessä, tulen todennäköisesti tekemään pidemmällä tavalla, eli vaihtaisin tiedostoissa käytettäviä merkistöjä tai asetuksia, saadakseni luettelomerkit myös tulostettua. Tämä takaisi hieman hienomman lopputuloksen.

4 TIETOKANTA

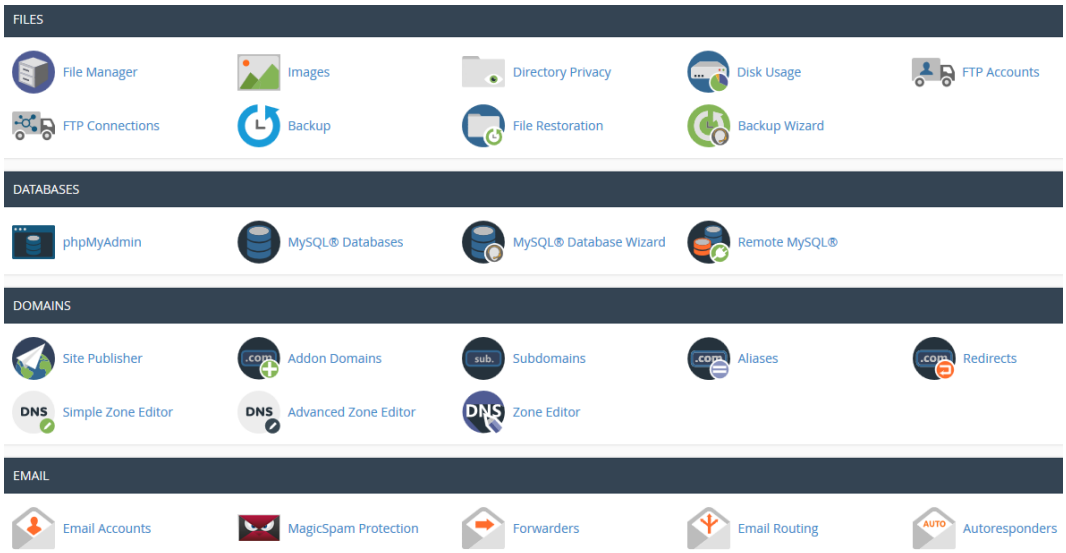
Tietokannan tarkoitus on koota tietoa, jolloin siihen on helppo päästä käsiksi, sekä tieto on helposti muokattavissa tai päivitettävissä (Rouse 2017). Tähän mennessä ohjelman kiinteiden vastausten muuttaminen on vaatinut palvelimelle kirjautumista ja vastauksen muokkaamista suoraan koodista. Vastausten siirtäminen tietokantaan ei vaikuta itse raporttiin, mutta raportin muokattavuus helpottuu.

Koska ohjelma on tarkoitus siirtää lopulta DevNet Oy:n kotisivuille ja kotisivut eivät sijaitse samalla palvelimella kuin ohjelma tällä hetkellä, niin työ siirretään oikeaan paikkaan tässä vaiheessa. DevNet Oy:n kotisivut ovat cPanelissa, joka on sovellus, joka tarjoaa hosting-palveluita. CPaneliin voi yhdistää suoraan MySQL nimiseen tietokantaan. CPanel ylläpitää sivustoa ja omaa työpöydän, joka mahdollistaa yhdistämisen eri verkkotunnuksiin, sähköposteihin ja tietokantoihin muun muassa. (Kinsta 2018.) Tämä yksinkertaistaa asiaa, sillä näin ohjelma ja tietokanta sijaitsevat samassa paikassa samojen tunnusten takana. Molempien muokkaus on myös näin helpompaa. Tiedostot siirretään ja tietokannat luodaan kuitenkin eri verkkotunnukseen eli domain:iin kuin DevNet Oy:n kotisivut. Ohjelma voidaan myöhemmin reitittää DevNet Oy:n kotisivuille näkyviin. Tällä tavoin ehkäistään, että mikään ei mene vahingossakaan vikaan käytössä olevien kotisivujen kanssa.

Tämä osa projektia jakautuu kahteen alalukuun. Tietokannan luontiin ja vastausten lisäämiseen tietokantaan sekä vastausten yhdistäminen raporttiin tietokannasta. Ensimmäistä vaihetta varten menen käymään paikan päällä DevNet Oy:ssä, saadakseni tunnukset tietokantaan. Minulle selitetään ja mahdollisesti näytetään myös pikaisesti, kuinka luonti tapahtuu sillä en ole itse tietokantoja luonut tai hakenut niistä tietoa. Tulen tutkimaan ja käyttämään tässä työnvaiheessa lähteenä w3scools.com sivustoa. W3scools on testaus, opettelu ja harjoittelu ympäristö, jossa on tietoa eri koodikielistä ja tietokannoista ja ohjeita mitä niillä voi tehdä (w3scools 2018.)

4.1 Tietokannanluonti ja tietojen lisääminen tietokantaan

Käytyäni paikan päällä DevNet Oy:n toimistolla minulle luotiin tunnukset cPaneliin sekä domain eli verkkotunnus, johon ohjelman siirrän. Koska cPanelin työpöytä on selkeä (Kuva 7), on helppo huomata, missä ja kuinka tiedostot siirretään. Vaihtoehtoina on myös eri tietokantoja, joita käyttää. Tutkin tietokannat ja niiden käytettävyyden selvittääkseni, mitä lopulta käytän.



Kuva 7 cPanelin työpöytä

4.1.1 7.5.2018

Päiväkohtainen tavoitteeni on siirtää ohjelma cPaneliin onnistuneesti. Tutustun myös kahteen tietokanta vaihtoehtoon, phpMyAdmin:iin sekä MySQL:ään. Muut vaihtoehdot ovat MySQL tietokantoja lisäominaisuuksin. Selvitettyäni tietokannan, johon aion ohjelman yhdistää, alan lisäämään raportin tekstikenttiä tietokantaan. Pyrin saamaan tietokannan kokonaisuutena valmiiksi jo päivän aikana, mutta viimeistään seuraavan päivän aikana.

Kirjauduttuani cPaneliin, menin tiedostot valikkoon ja painamalla lataa tiedostoja, pääsin siirtämään ohjelman tiedostot palvelimelta cPaneliin. CSS- sekä JavaScript-tiedostot ja ohjelmassa käytetyt kuvat olivat

kansioissa ja tiedostoja ei voinut siirtää tällä tavoin. Otettuani tiedostot kansioista pois pystyin lisäämään tiedostot. Tein kuitenkin cPaneliin uudet kansiot näille tiedostoille ja lisäsin ne niihin, jotta tiedostojen tiedostopolut pysyvät oikeina, ja tieto osaa liikkua oikeisiin tiedostoihin. Tämä myös selkeyttää tiedostojen hallintaa, kun tiedostot ovat järjestyksessä kansioissa. Testasin myös, että ohjelma toimii halutulla tavalla uudessa sijainnissa. Huomasin tällöin ilokseni ja hämmästykseksi, että PDF-muunnoksen tekevä API lisäsi PDF-tiedoston tiedostojen joukkoon. Tämä oli yllätys, koska vanhassa palvelimessa tämä ei onnistunut. Huomasin kuitenkin samalla, että pdf.php muodossa oleva PDF-raportti ei toimi. APIn luoma PDF-tiedosto vastaa täysin selaimeen auennutta raporttia. Ainoana erona se, että tiedosto on PDF-formaatissa. Tämä sopii paremmin tehtävänanon kannaltakin, joten yhdistin raportin aukeamaan PDF-tiedostoon, jonka API luo PDF-muunnoksen yhteydessä.

Tutustuttuani cPaneliin dokumentointiin tietokannoista huomasin, että minun tulee käyttää MySQL tietokantaa tai MySQL database Wizardia tietokannan luonnissa. Jälkimmäistä suositeltiin käyttämään ensimmäistä tietokantaa luodessa ja tällä tavoin tietokannnan sain myös luotua. Tämän jälkeen phpmyadmin:iin piti ilmaantua luomani tietokanta. (MySQL 2018.) Tietokanta lopulta ilmestyi ensin kirjauduttuani cPanelista ulos ja takasin sisään. Yrittäessäni miettiä sopivaa tapaa rankentaa tietokanta, aloin ensin luomaan yksittäin rivejä tulostettaville kysymyksille ja näihin vastauksia. Tajusin kuitenkin olevan yksinkertaisempi ja parempi tapa. Tietokanta taulun luotuani voin lisätä SQL komennoilla kaksi tekstikenttää, johon tulostettava aihe ja vastaus lisätään. Lisäsin myös rivin ID ja annoin tälle perusavain oikeudet. Perusavain identifioi rivillä olevan tiedon ja antoi oikeuden muokata tekstikenttiä. (w3schools 2018.)

En valitettavasti päässyt tavoitteeseeni saada tietokantaa valmiiksi päivän aikana. Tutkiessani MySQL dokumentaatiota sekä w3schoolsin materiaalia tietokannoista, kului suuri osa työajastani. Opin kuitenkin tietokannan rakentamisesta enemmän, kuten käyttämäni tavan luoda itse tietokanta tauluun tarvitsemani määrän kenttiä. Täten löysin selkeän

tavan, jolla tehdä tietokanta, joten seuraavan päivän aikana saan todennäköisesti tietokannan valmiiksi.

4.1.2 8.5.2018

Edellisen päivän tavoitteessani epäonnistuttuani, on tämän päivän tavoitteeni jatkaa edellispäivän tavoitetta ja saada tietokanta rakennettua valmiiksi. Tietokannan pitäisi rakentua nopeasti, sillä kuten edellä kerroin, löysin yksinkertaisen tavan koota tulostettavat tekstikentät ja eri tulostus vaihtoehdot kokonaisuudeksi. Pyrin myös selvittämään keinoa yhdistää tietokannan sisältö ohjelman raporttiin.

Jatkoin tietokannan tekoa siitä mihin edellispäivänä jäin. Luotuani ensimmäisen niin sanotun tunnisterivin, pystyin lisäämään raportissa tulostettavia rivejä. Jokaisella luodulla rivillä on ID-tunniste ja selitys, jotta tietokannan muokkaaja tietää, mitä kyseinen osa tarkoittaa. Ja viimeinen rivin tieto on raporttiin tulostettava osuus. Raporttiin tulostettavilla kohdilla on yhdestä neljään tulostus vaihtoehtoa. Esimerkiksi tietokannan ensimmäiset kaksi riviä on nimetty nimillä “friendlyurl1” ja “friendlyurl2”. Tulostettava osuus on ensimmäisessä “Sivuston SEO-osoitteet toimivat normaalisti”. Toinen on vastaavasti “Sivuston SEO-osoitteet ovat puutteellisia”. Yhdellä tietokannan rivillä on siis selitys tulosteeseen sekä tulostettava teksti. Raporttiin tulostettava vastaus riippuu sivuston tiedoista. Kokonaisuudessa loin tietokantaan 33 riviä (kuva 8). Muutamalla raportin tulostus kohdalla on jopa neljä vaihtoehtoa antaen eri tulostuksen. Näitä on muun muassa meta-otsikon ja -kuvauksen analysointi sekä sivun latausnopeuden arviointi. Nämä ovat jo yli kolmannes kirjoittamistani tietokannan riveistä. Osa tietokannan riveistä tulostetaan raporttiin vain, mikäli kyseinen asia puuttuu tutkittavasta sivusta.

	ID	tuloste	vaihtoehto
Muokkaa Kopioi Poista	1	friendlyurl1	Sivuston SEO-osoitteet toimivat normaalisti.
Muokkaa Kopioi Poista	2	friendlyurl2	Sivuston SEO-osoitteet ovat puutteellisia.
Muokkaa Kopioi Poista	3	mobiili1	Sivusto on responsiivinen mobiililaitteilla.
Muokkaa Kopioi Poista	4	mobiili2	Sivusto ei ole kovin responsiivinen mobiililaittei...
Muokkaa Kopioi Poista	5	metakuvaushuono	Sivuston meta-kuvaus eli meta-description on täysi...
Muokkaa Kopioi Poista	6	metakuvauspitka	Sivuston meta-kuvaus löytyy mutta se on liian pitk...
Muokkaa Kopioi Poista	7	metakuvauslyhyt	Sivuston meta-kuvaus löytyy mutta se on hiukan puu...
Muokkaa Kopioi Poista	8	metakuvaushyva	Sivuston meta-kuvauksessa ei ole puutteita.
Muokkaa Kopioi Poista	9	metaotsikkohuono	Sivustolle ei ole asetettu meta-otsikkoa eli meta-...
Muokkaa Kopioi Poista	10	metaotsikkopitka	Sivustolle on asetettu meta-otsikko mutta se on li...
Muokkaa Kopioi Poista	11	metaotsikkolyhyt	Sivustolle on asetettu meta-otsikko mutta se on pu...

Kuva 8 Ote tietokantaan lisäämistäni riveistä ja vastauksista

Samasta w3schoolsin tietokanta materiaalista, josta löysin keinon luoda tämä tietokanta, löysin myös keinon, jolla yhdistää tietokanta raporttiin. Samassa materiaalissa oli myös esitelty muutama eri tulostustapa tietokannan sisällölle. Kyseisen tavan pitäisi olla toimiva ja testaan sitä heti seuraavalla kerralla.

Saavutin päivälle asettamani tavoitteet viimeistellessäni tietokannan. Tietokannan tekemisessä ei ollut suurempia ongelmia. Joissakin tulostettavissa kentissä oli tulostettava muuttuja keskellä virkettä. Näissä tapauksissa muokkasin lauseen rakennetta siten, etten lisännyt muuttujaa tietokantaan, koska epäilin ettei se olisi toiminut. Siirsin muuttujan tulostettavan osion loppuun, jolloin muuttuja tulee raporttiin heti tietokannasta haetun tulostuksen jälkeen. Ehdin myös tutkia keinoa yhdistää tietokanta raporttiin ja tapaa tulostaa vastaukset tietokannasta.

4.2 Tietokannan yhdistys raporttiin

Vaikka olen nyt luonut tietokannan ja lisännyt raportin sisällön ja teksti vastaukset sinne, ei raportti ole missään yhteydessä tietokantaan. Tämä pitää toteuttaa PHP-kielellä. Käytän hyväkseni w3schoolsista löytämääni ohjetta koskien tietojen lisäystä tietokantaan ja tietokannan tietojen hakemista verkkosivulle. Yhdistettyäni tietokannan ja raportin, voin alkaa hakemaan tietokannasta tulosteita ja korvaamaan tämän hetkisen teksti osuuden koodilla, joka hakee saman tiedon tietokannasta.

4.2.1 9.5.2018

Päivän tavoitteeni on kokeilla edellispäivänä w3scoolsista löytämääni tapaa yhdistää tietokanta raporttiin. Mikäli keino on toimiva, aion päivän aikana saada raportin yhdistämisen lisäksi saada kaikki raportin kohdat tulostumaan tietokannasta.

Aluksi yhdistäessäni tietokantaa raporttiin, oli minun yhdistämistä varten ilmoitettava tietokantani nimi sekä tunnukseni ja serverini. Tallensin tiedot muuttujiin ja yhdistin seuraavalla tavalla raporttisivun tietokantaan:

```
$conn = new mysqli ($servername, $username, $password, $dbname);
```

Tämän jälkeen tulostin tietokantani taulukkona raporttisivulla. Tämä taulukko ei normaalisti näy raportissa, mutta tulostin taulukon näkymään raporttisivun alaosaan voidakseni seurata, millä arvoilla haen mitäkin kohtaa tietokannassa. Samalla lisäsin tulokset muuttujaan $row. Tällä muuttujalla haen taulukosta tulosteet ja tulostan ne oikeissa kohdissa. Testasin tulostusta tietokannan ensimmäisiin kohtiin ja huomasin sen tulostuvan oikealla tavalla. Esimerkiksi mikäli sivuston SEO-osoitteet olivat puutteellisia, haettiin vastaus tietokannasta tavalla $row [1][2]. Ensimmäinen luku vastaa ensimmäistä riviä arrayssa, joka etsii oikean rivin tietokannasta. Toinen luku taas etsii riviltä tulostettavan tulosteen (kuva 9). (w3scools 2018.)

```
Array
(
    [0] => Array
        (
            [0] => 1
            [1] => friendlyurl1
            [2] => Sivuston SEO-osoitteet toimivat normaalisti.
        )

    [1] => Array
        (
            [0] => 2
            [1] => friendlyurl2
            [2] => Sivuston SEO-osoitteet ovat puutteellisia.
        )

    [2] => Array
        (
            [0] => 3
            [1] => mobiili1
            [2] => Sivusto on responsiivinen mobiililaitteilla.
        )

    [3] => Array
        (
            [0] => 4
            [1] => mobiili2
            [2] => Sivusto ei ole kovin responsiivinen mobiililaitteilla.
        )
```

Kuva 9 neljä ensimmäistä tietokannan riviä taulukko muodossa

Tällä tavoin hain kaikkiin tietokannan kohtiin tulostuksen. Kohtasin kuitenkin ongelman, sillä ohjelma meni rikki kohdissa, joissa tulostuksen jälkeen tuli muuttuja. Ongelman sain korjattua kuitenkin laitettuani muuttujan tulostumaan eri PHP tekstikenttänä, vaikkakin samalla rivillä. Tämän jälkeen sain yhdistettyä tietokannan kaikki tulosteet raporttiin siten, että raportti tulostuu oikein. Huomasin kuitenkin, että raportti ei tulosta ä ja ö kirjaimia vaan tulostaa ne  -merkkeinä. Tiesin, että vika johtuu tietokannassa käytössä olevasta merkistöstä. Ongelman sain korjattua vaihdettua tietokannan merkistöksi UTF-8_generalin, joka tukee ä- ja ö-merkkejä (Laaksonen 2011). Oikean merkistön löydettyäni korjasin tietokannan vastaukset ja raportin tulostus toimi onnistuneesti.

Saavutin päivälle asettamani tavoitteet saadessani tietokannan yhdistettyä raporttiin. Sain myös korjattua ilmenneet ongelmat, jolloin tietokanta toimii moitteettomasti. Edellisellä kerralla löytämästäni w3schoolsin ohjesivusta oli paljon hyötyä minulle. Sain yhdistettyä sivun tietokantaa suhteellisen helposti ohjeiden avulla ottaen kuitenkin huomioon, etten ole toiminut

tietokantojen kanssa juuri ollenkaan ennen. Tämän myötä sain tietokanta osan valmiiksi.

4.2.2 Työvaihe analyysi

Tämän työvaihe analyysin kirjoitan tietokannoista kokonaisuudessaan, enkä jaa sitä kahteen osaan alalukuihin. Tämän teen koska tietokanta on yksi kokonaisuus, johon kuuluu luonti, tietojen lisäys ja raporttiin tulostus.

Tämän työn alkupuolella kerroin, kuinka pidin PDF-muunnoksen tekemistä haastavimpana asiana. Tietokanta ei vaativuudeltaan ollut yhtä haastava, mutta sen tekoa jännitin kaikista eniten. Tämä johtuu siitä, että en ole koskaan ennen luonut tietokantaa, lisännyt omia rivejä sinne tai tulostanut tietokannasta mitään. Kokemukseni tietokantojen kanssa rajoittui koulun kurssiin, jonka aikana valmiista tietokannoista SQL komentojen avulla haettiin tietoa. Asia oli siis minulle täysin uutta ja opin kaikesta tekemästäni.

Tutkiessani cPanelin ja MySQL:n dokumentointia ja w3schoolsin MySQL osiota, opin kuinka tietokanta tulee luoda. Opin myös, kuinka tietokantaan lisätään tietokantataulu ja siihen rivejä, joihin raportin sisältö lisätään. Opin myös, kuinka täyttää nämä taulut, rivit ja kentät sekä yhdistää se haluamaani sivuun, ja tulostaa tietokannan rivien vastaukset raporttiin. Suoriuduin työvaiheesta mielestäni hyvin, ottaen huomioon tämän olleen ensimmäinen kerta, kun teen tietokantaa. Kokonaisuudessa työvaihe oli opettavaisin kaikista asioista ja työvaiheista, joita työtä tehdessäni vastaan minulle tuli, juuri siitä syystä johtuen, että kaikki oli minulle uutta.

Ongelmia minulle vastaan tuli muutamia. Koittaessani luoda tietokantatauluun rivejä yksinkertaisesti “lisää rivi” komennolla, vaati rivi tietoja, joita en ymmärtänyt ja olivat tarpeettomia itselleni. Löysin ratkaisun kuitenkin w3schoolsin MySQL ohjeistuksesta, jossa neuvotaan, kuinka lisätä rivejä tietokantaa. Ohjeen avulla lisäsin SQL komennolla tietokantaan otsikko rivin, jossa on tunniste, tulosteen otsikko ja tulostettava vastaus. Tämän avulla sain lisättyä tietokantaan rivin

jokaiselle raportissa tulostettavalle tekstiosuudelle tai -vaihtoehdolle. Toinen vastaan tullut ongelma tuli yhdistettyäni tietokannan raporttiin. Tulostus toimi muuten hyvin paitsi, että ä- ja ö-kirjaimet eivät tulostuneet. Tiesin ongelman johtuvan samasta asiasta kuin PDF-raportin tulostuksessa oli luettelomerkkien kanssa. Tietokannan merkistö ei siis tukenut kyseisiä merkkejä ja tästä syystä korvasi ne. Ongelman ratkaisemiseksi perehdyin tarkemmin tietokannan eri asetuksiin. Ongelman korjaamiseksi vaihdoin kahteen paikkaan käytettäväksi merkistöksi UTF-8_general_ci:n. Tämän lisäksi raporttisivun koodiin myös kohdan, jonka avulla tietokannasta saatu tieto on UTF-8 merkistöä.

Tietokannan sisällön tulostamisessa olisi mahdollisesti ollut muitakin vaihtoehtoisia tapoja. Tapani kuitenkin tulostaa tietyltä riviltä tietty vastaus on kuitenkin paras ja yksin kertaisin. Tavalla valitsen taulukosta esimerkiksi rivin 1 [1] ja kentän 2 [2]. Näin tulostan taulukon ensimmäisen rivin toisen tekstikentän (kuva 9). Tietokannan olisin kuitenkin voinut rakentaa vaihtoehtoisesti toisella tavalla, joka olisi ollut selkeämpi. Nyt minulla on luotu jokaiselle tekstille oma rivinsä. Tämä tarkoittaa, että mikäli raportin kohdalla on neljä tulostus vaihtoehtoa, niin jokainen vaihtoehto on omalla rivillään. Tämän myötä tulostus rivejä on 33 kappaletta ja tulostettava kenttä on aina kenttä numero kaksi. Vaihtoehtoisesti olisin voinut luoda riville kolme tekstikenttää enemmän, jolloin minulla olisi ollut yhdellä rivillä tulostettavan rivin tunniste, tulostettava kohdan nimi sekä tulostettavat tekstivaihtoehdot. Tässä tapauksessa tietokantaan olisi tullut vähemmän rivejä ja lopputulos olisi ollut selkeämpi. Näiden vaihtoehtojen erot ovat seuraavanlaiset: Tällä hetkellä neljän tulostus vaihtoehdon tulosteet haetaan tavalla [1][2], [2][2], [3][2], [4][2]. Vaihtoehtoisella tavalla tulostus haettaisiin tavalla [1][1], [1][2], [1][3], [1][4]. Tämä vaihtoehtoinen tapa olisi ollut paljon selkeämpi ja aion käyttää sitä tulevaisuudessa, mikäli olen tekemisissä tietokantojen kanssa.

5 LOPPUTULOS

Raportin tilaajan tilatessa raportin oli tarkoitus kerätä tilaajan sähköpostiosoite ja tilattavan sivun nimi DevNet Oy:n uutiskirjejärjestelmään ”DevNet Newsletter”. Tämän tarkoituksena oli, että DevNet saa tiedon kaikista sivuista, josta raportti on tehty ja tilaajien sähköpostit. Täten DevNet Oy voisi käyttää kerättyjä tietoja myöhemmin markkinointi tarkoitukseen. Päätimme kuitenkin DevNet Oy:n kanssa, että emme sisällytä uutiskirjejärjestelmää vielä tähän työhön. Päädyimme tähän tulokseen aikataulusta johtuen. Työn valmistumisen takaraja alkoi tulla vastaa ja DevNet Oy:n kiireellisyydestä johtuen emme saaneet sovittua tapaamista, jotta olisimme voineet yhdistämisen tehdä. Ja koska minulla ei ole aikaisempaa kokemusta heidän omasta uutiskirjejärjestelmästään, en olisi tähän yksin pystynyt.

Ohjelma tulostaa tällä hetkellä asiakkaalle PDF-muotoon raportin tilaajan haluamasta sivusta. Alkuperäinen suunnitelma oli tallentaa PDF-tiedosto ja lähettää tilaajalle raportti sähköpostiin. Päädyin kuitenkin vain ohjaamaan tilaajan suoraan PDF-muotoiseen raporttiin sivusta. Täten ei tilaajan tarvitse kirjautua erikseen sähköpostiin, vaan raportti on suoraan katsottavissa, ladattavissa tai tulostettavissa tilaajan oman halun mukaan. Tämä oli työn päätarkoitus. Ohjelma on myös siirretty uuteen sijaintiin cPaneliin, jossa raportin tiedot haetaan samassa yhteydessä olevasta tietokannasta. Tämä vaihe toteutui täysin suunnitellusti. Uutiskirjejärjestelmään yhdistäminen olisi ollut alkuperäisen työsuunnitelman viimeinen vaihe. Tätä ei kuitenkaan sisällytetä ohjelmaan johtuen edellisessä kappaleessa mainituista aikataulullisista ongelmista.

Työ on kokonaisuudessaan käyttöönottoa vaille valmis. Käyttöönottoa varten työ olisi vielä reititettävä cPanel sijainnista näkymään DevNet Oy:n kotisivuille heidän haluamaansa paikkaan. Käyttöönoton yhteydessä heidän täyttyy vain määritellä, minne ohjelma sijoitetaan, ja mahdollisesti muokata tilauslomakkeen asettelua sekä infotekstejä lomakkeen

yhteydessä. Myös työstä nyt pois jätetty uutiskirjeeseen yhdistäminen on mahdollista tehdä myöhemmin.

6 POHDINTA

Työn aikana tavoitteeni oli tutustua siihen, mitä kaikkea ohjelmia ja työkaluja IT-alan projektissa hyödynnetään ja täten täydentää osaamistani. Olennaisia asioita joita työ vaati, olivat muun muassa eri ohjelmointikielten tuntemus, kuten CSS, PHP ja HTML. Tämän lisäksi tietokantojen ymmärtäminen ja luonti oli tärkeä osa työtä. Myös ongelmanratkaisu oli työn suorittamisessa tärkeää, sillä selvitin itse asioiden toimivuuden ja tarvittaessa kysyin konsultaatioapua DevNet Oy:stä. Minulla oli tietopohjaa valmiiksi muun muassa käytettävistä ohjelmointikielistä. Tiesin kuitenkin jo työtä aloittaessani, että joudun soveltamaan osaamistani ja etsimään ratkaisuja työn ongelmiin tekemällä ja kokeilemalla.

Ennen työn aloitusta, vastaavanlainen kokemukseni rajoittui juuri alkuperäisen SEO-raportin tekoon harjoitteluni aikana. Tämän tein yhdessä toisen harjoittelijan kanssa, joten koko vanha ohjelma ei ollut minun kirjoittamaani. Tästä syystä aluksi tutkin tiedostot tarkkaan läpi ymmärtääkseni, mitä kaikkea ohjelma tekee ja missäkin vaiheessa mitäkin tapahtuu. Olin toki käynyt ohjelman sisällön läpi, mutta koska kaikki ei ollut itse kirjoittamaani, piti minun ymmärtää koko ohjelman toiminta perusteellisesti ennen, kun aloitin ohjelman muokkaamisen.

Kehityin työn tekemisen aikana mielestäni oikein hyvin, sillä aiempi kokemukseni samankaltaisista töistä, oli harjoittelujaksoltani. Tästä oli kuitenkin ehtinyt kulua jo melkein vuosi työtä aloittaessani ja joitakin asioita joutui aluksi palauttelemaan mieleen. Opin etsimään vaihtoehtoisia keinoja työn toteuttamisessa, kuten esimerkiksi etsiessäni keinoa muuntaa SEO-raportti PDF-muotoon. Myös analysoidessani tekemiäni työvaiheita, pyrin keksimään vaihtoehtoisia tapoja tai keinoja, jolla tekemäni voisi mahdollisesti toteuttaa paremmin tai toisenlaisessa tilanteessa. Kehittymiseni käy myös ilmi päiväkirjaraportoinnistani, sillä alussa esitellen työvaiheet ja sen ongelmat yleisessä mielessä ja päivän tapahtumista raportoidessani osasin löytää ongelman, keksiä siihen ratkaisun ja

toteuttaa sen. Ongelmanratkaisukykyni kehitys sekä ohjelmointikielien osaamiseni korostui yllättävien ongelmien ilmentyessä. Näitä ongelmia oli muun muassa raportin tulostuksen puuttuminen vastauksia haettaessa muuttujilla, joka esti raportin sisällön tulostuksen ilman valintasivun läpikulkua. Tämä ongelma ilmeni sekä avatessa raporttia selainversiossa että PDF-versiossa. Muita suurempia ongelmia, joita työn edetessä ilmaantui, oli merkistöjen kanssa, jolloin raportti ei tulostunut oikein. Samaan ongelmaan törmäsin tietokannan ja PDF-muunnoksen kanssa.

Taitoni tietokantoihin liittyen kehittyivät mielestäni kaikista eniten tässä opinnäytetyöprosessissa. Tämä siksi, etten ollut aikaisemmin ollut tekemisissä tietokantojen kanssa. Koulun kursseilla kävimme läpi tietojen ja rivien hakemisen tietokannoista, mutta emme niiden luontia. Minulle oli luvattu tarjota apua, mikäli en itse asiaa osaisi. Luin eri dokumentaatioita tietokannoista ja sain käsityksen, kuinka tietokantojen kanssa toimitaan. Lopulta onnistuinkin luomaan tietokantaan vastaukset tavalla, joka toimi työn kannalta. Lähtökohdan huomioon ottaen kehitykseni oli mielestäni suurta, sillä koen nyt osaavani luoda yksinkertaisia tietokantoja.

Kirjoitettuani ensin päivittäin päiväkirjanomaisesti päiväni tavoitteet ja tekemisen, tein myös kyseisestä työvaiheesta työvaiheanalyysiin. Tähän kokosin työvaiheen aikana oppimani ja kohtaamiani ongelmia. Etsin myös vaihtoehtoisia tapoja tehdä kyseiset asiat. Tämä auttoi tehtyjen asioiden sisäistämisessä, sillä jouduin paneutumaan uudelleen jo tehtyyn asiaan. Lisäksi minun oli mietittävä, kuinka tekemäni toimii ja olisiko minulla ollut vaihtoehtoista ratkaisua.

Päiväkirjamuotoisella opinnäytetyöllä oli mielestäni sekä etunsa että haittansa. Haittapuolena oli, että jouduin samaan aikaan kirjoittamaan opinnäytettä ja tekemään SEO-raportti -ohjelmaa. Opinnäytetyötä kirjoittaessani saatoin alkaa miettiä ohjelmaa, jolloin keskittymiseni herpaantui. Vaihtoehtoisesti, jos tein ohjelmaa pidempään kerralla, oli minun tutkittava ohjelmaa ja jälkikäteen katsottava, mitä kaikkea tein työn etenemiseksi. Päiväkirjaopinnäytetyöllä oli kuitenkin myös etunsa.

Kirjoittaessa päiväkirjaraportteja, jouduin paneutumaan enemmän tekemääni. Asioita oli pohdittava tarkemmin, kun pyrin ilmaisemaan asiat yksityiskohtaisesti ja mahdollisimman selkeästi. Tämä auttoi tehdyn asian ymmärtämisessä ja edisti oppimaani. Erityisenä haastena päiväkirjamaisessa opinnäytetyössä, ja erityisesti tässä työssä, oli lähteiden käyttäminen. Koska aiheeni on käytännön työ ja kirjoitan työtä koodina, on valtaosa lähteistä käyttämieni koodikielien ja ohjelmien kotisivuja ja ohjelmointi foorumeita ja yleisiä ohje - ja harjoitusmateriaalisivuja. Teoriaa, käsitteitä ja ohjelmia selittäessäni lähteiden löytämisessä ei ollut ongelmia, mutta itse ohjelman tekemisessä ajoittain oli.

Kokonaisuudessa sekä päiväkirjamuotoinen opinnäytetyö ja tekemäni ohjelma opettivat minua paljon. Jouduin perehtymään uusiin asioihin ja selvittämään niiden toimintaa. Jouduin myös ratkaisemaan ongelmia, jotka ilmaantuivat työn edetessä. Työn alussa tavoitteenani oli syventää osaamistani ohjelmointikielissä sekä oppia muista alan työtehtävistä. Mielestäni saavutin työlle asettamani tavoitteet.

LÄHTEET

CPanel. 2018. MySQL Databases. [Viitattu 7.5.18]. Saatavissa: https://documentation.cpanel.net/display/68Docs/MySQL+Databases

Gabbert, E. 2017. What Is SEO Content? A Guide to Creating Content for SEO. [Verkkodokumentti]. [Viitattu 7.10.17]. Saatavissa: https://www.wordstream.com/blog/ws/2012/01/17/seo-content-beginners-guide

Github. 2018. PHP library generating PDF files from UTF-8 encoded HTML. [Viitattu 12.4.18] https://github.com/mpdf/mpdf

Hughes, M. 2015. What Are APIs, And How Are Open APIs Changing The Internet. [Verkkodokumentti]. [Viitattu 10.10.17]. Saatavissa: http://www.makeuseof.com/tag/api-good-technology-explained/

Kinsta. 2018. What is cPanel? The Control Dashboard Explained For Beginners. [Viitattu 7.5.18]. Saatavissa: https://kinsta.com/knowledgebase/what-is-cpanel/

Laaksonen, A. 2011. PHP-ohjelmointi: Osa 17 - Merkistöt . [Viitattu 9.5.18]. Saatavissa: https://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=php_17#koodauksenvalinta

Mysiteauditor. 2017. [Viitattu 1.10.17]. Saatavissa: http://mysiteauditor.com/

MySQL. 2018. MySQL Documentation. [Viitattu 7.5.18]. Saatavissa: https://dev.mysql.com/doc/refman/5.6/en/data-types.html

Pdfcrowd. 2018. [Viitattu 6.4.18]. Saatavissa: https://pdfcrowd.com/

Quora. 2016. What is the best way to generate a PDF file from HTML and CSS?. [Viitattu 5.4.18]. Saatavissa: https://www.quora.com/What-is-the-best-way-to-generate-a-PDF-file-from-HTML-and-CSS

Rousa, M. 2017. database (DB). [Verkkodokumentti]. [Viitattu 10.10.17]. Saatavissa: https://searchsqlserver.techtarget.com/definition/database

Stackoverflow. 2016. Best way to create a PDF with PHP. [Viitattu 12.4.18]. Saatavissa: https://stackoverflow.com/questions/2132015/best-way-to-create-a-pdf-with-php

Stackoverflow. 2014. Convert php file to pdf file by using mPDF. [Viitattu 10.4.18]. Saatavissa: https://stackoverflow.com/search?q=convert+php+file+to+pdf

Sublime text3. 2017. [Viitattu 17.10.17]. Saatavissa: https://www.sublimetext.com/3

w3scools. 2017. Bootstrap 3 Tutorial. [Viitattu 13.2.17]. Saatavissa: https://www.w3schools.com/bootstrap/default.asp

w3scools. 2018. PHP Select Data From MySQL. [Viitattu 8.5.18]. Saatavissa: https://www.w3schools.com/php/php_mysql_select.asp

w3schools. PHP Create a MySQL Database. [Viitattu 8.5.18]. Saatavissa: https://www.w3schools.com/php/php_mysql_create.asp

w3schools. 2018. PHP Create MySQL Tables. [Viitattu 8.5.18]. Saatavissa: https://www.w3schools.com/php/php_mysql_create_table.asp

Winscp. 2017. [Viitattu 17.10.17]. Saatavissa: https://winscp.net/eng/index.php

Wkhtmltopdf. 2018. [Viitattu 10.4.18]. Saatavissa: https://wkhtmltopdf.org/

Tcpdf. 2018. [Viitattu 10.4.18]. Saatavissa: https://tcpdf.org/