Haaga-Helia
ammattikorkeakoulu Oy

Ohjelmistokehittäjänä omassa yrityksessä

Niko Tikkanen

Opinnäytetyö

Tietojenkäsittelyn koulutusohjelma

2019

Haaga-Helia ammattikorkeakoulu Oy

Tiivistelmä

Päiväys

Tekijä(t)
Niko Tikkanen

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Opinnäytetyön otsikko
IT-yrittäjän päiväkirja

Sivu- ja liitesivumäärä
60

Opinnäytetyön otsikko englanniksi
As a software developer in own company

Opinnäytetyö on toteutettu päiväkirjamuotoisena. Opinnäytetyö kattaa kymmenen viikon ajalta kuvailua päivittäisestä työstäni yrittäjänä IT projektin parissa. Merkinnöissä asetetaan päivittäisiä tavoitteita ja analysoidaan niiden toteutusta. Analyysit keskittyvät viikon aikana tehtyyn työhön ja sen toteutukseen.

Kirjoittaja työskentelee ohjelmistokehittäjänä yksityisyrittäjänä. Työtehtäviin kuuluu ohjelmiston suunnittelu, toteutus ja siihen liittyvien projektien parissa työskentely.

Asetin tavoitteeksi ennen opinnäytetyön alkua jatkokehittää jo aloitetun tuotteen toteutusta ja oppia lisää ohjelmistokehityksen teknologioista ja palvelinympäristöistä. Opinnäytetyön aikana saavutin tavoitteeni ja kehityin niin ohjelmistokehityksessä käytettävien teknologioiden kanssa, kuin palvelinympäristössäkin. Oman kehityksen lisäksi olen myös kuvannut ongelmia, joita olen kohdannut työssäni, sekä miten ne ovat ratkenneet.

Asiasanat
yrittäjä, ohjelmistokehitys, web-ohjelmointi

Sisällys

1 Johdanto

Päiväkirjatyyppisen opinnäytetyön kesto on kymmenen viikkoa, joka koostuu päivittäisistä raporteista, analyyseistä viikon lopussa sekä pohdin oman osaamisen kehittymistä ja työtehtäviäni sekä tarkastelen viikon aiheita kirjallisuuteen ja lähteisiin viitaten. Opinnäytetyön suurin osuus, eli päiväkirjatyyppisen osuuden kirjoittaminen alkoi 18.2.2019 ja päättyi 25.4.2019.

Opinnäytetyö keskittyy projektiin, jonka kanssa työskentelen pelkästään tällä hetkellä. Kyseessä on internet markkinoinnissa käytettävä tuote, joka auttaa yrityksiä löytämään verkkosivujen ongelmakohdat ja parantamaan tulosten perusteella verkkosivuja tuottavammaksi. Tuotetta varten on perustettu osakeyhtiö Britanniaan ja yrityksessä on toinen henkilö minun lisäkseni, joka vastaa rahoituksesta, markkinoinnista ja visuaalisesta suunnittelusta. Vastaan tuotteen suunnittelusta ja toteutuksesta. Toimin fullstack kehittäjänä tuotteen parissa.

Työympäristöni on kehitysvaiheessa Windows 10 pohjainen ympäristö, päivittäisessä käytössäni minulla on PHP ja JavaScript ohjelmointikieli. Lisäksi apuna toimii niihin tehdyt kirjastot ja frameworkit. PHP ohjelmointikieli, joka mahdollistaa esimerkiksi MVC mallisen web-kehityksen. PHP:n apuna minulla on käytössä Laravel framework, joka auttaa kehitysnopeudessa, sekä hoitaa suurimman osan verkkosivuille tarvittavista koodeista, kuten osoite reitityksen, ulkoasun hallinnan ja tarvittavat komponentit turvalliseen käyttäjän autentikoimiseen. Kehityksessä käytettävään ohjelmistoon kuuluu tekstieditori Sublime Text 3, Git versionhallintatyökalu, NPM ja Composer paketinhallintaohjelmat, Elasticsearch hakukone sekä MySQL tietokanta.

Työtehtävissäni vaaditaan laajaa ohjelmointiosaamista niin front- kuin back-end ympäristöistä. Lisäksi vaaditaan laajaa osaamista tietokannoista ja ohjelmiston suunnittelusta lopulliseen toteutukseen asti. Lisäksi kehityksessä tarvitaan ongelmanratkaisukykyä.

Front-end puolen toteutus vaatii tuntemusta JavaScript ohjelmointikielestä. JavaScriptin lisäksi pääasiassa käytössä on HTML ja CSS. Lisäksi apukirjastoina on käytössä jQuery, VUE ja Semantic UI. Ymmärrys JavaScriptin rakenteesta ja niiden toiminnallisuuksista on oltava vahvaa, jotta kehitys tapahtuu nopealla tahdilla.

Back-end puoli on toteutettu lähes kokonaan PHP ohjelmointikieltä käyttäen. PHP ohjelmointikieleen voi tutustua esimerkiksi Branko Ajzelen kirjoittamassa kirjassa *Mastering PHP 7*, jossa käsitellään PHP kirjoittamista syntaksista lähtien, jopa edistyneisiin toimin-

toihin ja ohjelmistokoodin testaukseen. Back-end puoli vaatii lisäksi API rajapintojen tuntemusta, sekä Laravel frameworkin tuntemusta. Laravel frameworkin tuntemus on lähes yhtä tärkeää, kuin tuntea PHP ohjelmointikieli. Jotta frameworkilla voi kehittää tehokkaasti ja turvallisesti, tulee frameworkin eri komponentit tuntea laajasti.

Suurimpana kehityksessä käytettävänä työkaluna toimii Git versionhallinta, jotta päivitykset saa tehokkaasti talteen ulkopuoliselle palvelimelle, lisäksi olisi erittäin huono ratkaisu pitää ohjelmiston koodi pelkästään omalla tietokoneella, koska kovalevyn hajotessa olisi riski menettää kaikki tehty työ.

Tietokantana projektissa toimii MySQL tietokanta, joka hoitaa kaiken tärkeän, kuten laskutuksen, asiakasrekisterin, käyttäjätiedot ja asiakasportaalin asetukset. Elasticsearch hakukone toimii tietokantana kaikelle asiakkaan sivuilta kerätylle datalle. Elasticsearch päätyi ratkaisuksi ylivoimaisen nopeutensa takia. Se on tällä hetkellä maailman suosituin hakukone yritysympäristössä.

Kehitysympäristön käyttö vaatii tuntemusta komentorivin kanssa työskentelystä. Komentorivin kautta saadaan käynnistettyä kehitysvaiheessa oleva tuote, sekä aputoiminnot, jotka auttavat kehittämisen nopeuttamisessa, kuten SCSS. Lisäksi viralliselle palvelimelle pääsy vaatii erityisesti tuntemusta Linux kehitysympäristöstä, jotta siellä voi tehdä ylläpitotoimenpiteitä.

1.1 Keskeiset ammattikäsitteet

API

Ohjelmistorajapinta, jonka avulla ohjelmistot lähettävät ja vastaanottavat tietoa keskenään.

CRUD

CRUD (Create, read, update delete) eli luokka, jonka avulla voidaan lisätä, lukea, päivittää tai poistaa dataa esimerkiksi tietokannasta.

CSS

Sisältää ulkoasumäärittelyyn vaaditut tiedot verkkosivuilla.

Composer

Paketinhallintaohjelmisto PHP kirjastoille.

Controller

Kontrolleri toimii välikätenä tietokannan ja ulkoasun kanssa. Lisäksi kontrolleri käyttää model komponentteja apunaan.

Docker

Palvelimella ajettavia eristettyä palveluita, jotka käyttävät palvelimen tehoa mutta palvelimen muut asetukset eivät sotke niiden toimintaa.

Dovecot

Ohjelmisto, joka sisältää sähköpostipalvelimen palvelut, jotka hallitsevat miten ja missä viestit säilytetään.

Elasticsearch

Palvelimelle asennettava hakukone, jota voidaan käyttää ohjelmistokehityksessä varastoimaan tietoa.

Framework

Kehikko, joka sisältää tietyn kehitystavan ohjelmistokielelle. Framework sisältää yleensä paljon apuluokkia kehityksen nopeuttamiseksi.

Front-end ja back-end

Ohjelmistokehityksen kaksi eri puolta. Front-end eli selaimessa näkyvä ja ajettava koodi. Back-end kattaa kaiken muun, näkymätön osuus käyttäjälle. Back-end puoli hoitaa esimerkiksi tietokantayhteydet ja ohjelman logiikan.

Fullstack

Fullstack kehittäjä työskentelee molempien, front-end ja back-end puolien parissa.

HTML

Merkkauskieli, joka kertoo missä järjestyksessä nettisivun ulkoasun osat ovat.

iRedMail

Sähköpostiohjelmisto, joka pitää sisällään Dovecot ja Postfix sähköpostipalvelut.

JavaScript

Erittäin suosittu ohjelmistokieli, jota ajetaan yleensä selaimessa.

JSON

Tiedostomuoto, jota käytetään esimerkiksi ohjelmistojen välillä.

jQuery

JavaScript kehityksessä käytetty apukirjasto, joka auttaa esimerkiksi HTML sivujen rakenteiden muokkauksessa.

Kubernetes

Kubernetes on ohjelmisto, joka ohjaa virtuaalisesti ajettavia ohjelmistoja ja hoitaa niiden käyttöönoton, skaalauksen ja hallinnan automaattisesti.

Laravel

PHP kielelle tehty erittäin suosittu frameworkki.

Luokka

Luokka, MVC mallissa model, joka voi olla esimerkiksi yksi iso eristetty tietorakenne ohjelmistossa tai osuus, joka hoitaa yhden ison tehtävän. Luokka sisältää yleensä metodeja.

Metodi

Metodin tarkoitus on suorittaa yksi tietty tehtävä ohjelmistossa.

MVC

MVC (Model, view, controller) on ohjelmistokehityksessä käytetty malli, jossa ohjelmiston eri osat ovat jaettuna eri osioihin.

MySQL

Relaatiotietokanta, jota käytetään SQL (Structured Query Language) kutsuilla.

NPM

Paketinhallintaohjelmisto JavaScript pohjaisille kirjastoille.

Optin

Käyttäjän suostumuksella tietokantaan tallennettava tieto, esimerkiksi sähköpostiosoite markkinointitarkoitukseen.

PHP

Web-palvelinympäristössä ajettava ohjelmistokieli.

Postfix

Sähköpostin välitysohjelma. Hoitaa yhteydet, kun sähköpostia lähetetään tai saapuu palvelimelle.

SCSS

Merkkauskieli, päivitys CSS kielelle. SCSS koodi auttaa kirjoittamaan koodia nopeammin kuin perinteinen CSS. SCSS käännetään automatisoidulla ohjelmalla CSS koodiksi.

SSL

Salaa verkkoliikenteen ja näin suojaa verkkosivua hakkereilta.

Sublime Text 3

Erittäin nopea ohjelmistokehitykseen tarkoitettu tekstieditori, jonka toimintoja voi laajentaa tekstieditorin omalla paketinhallintajärjestelmällä.

VUE

Apukirjasto verkkosivujen kehitykseen, avustaa paljon muuttuvan HTML sisällön toteutuksessa.

2 Lähtötilanteen kuvaus

2.1 Oman nykyisen työn analyysi

Työhöni kuuluu erilasia työtehtäviä ohjelmistokehityksen parissa. Tällä hetkellä ohjelmistokehitys tapahtuu ainoastaan *Monolyth* projektin parissa, internet markkinointiin liittyvä tuote, jota olemme kehittäneet yhteistyökumppanin kanssa jo 2018 kesäkuusta lähtien. Toimin fullstack kehittäjänä, eli työkuvaani kuuluu niin front- kuin back-end puolen työstäminen. Lisäksi tehtäviini kuuluu ohjelmistosuunnittelu, eli suunnittelen tuotteen eri osat alusta loppuun asti, joihin kuuluu tietokannan, ohjelmiston logiikan ja käyttöliittymän suunnittelua.

Tärkeimpiin työtehtäviini kuuluu:

- Ohjelmiston suunnittelu.
    - Toiminnallisuuksien suunnittelu.
    - Tietokannan suunnittelu.
    - Käyttöliittymän suunnittelu.
- Ominaisuuksien ohjelmointi.
    - Back-end puolella tapahtuva ohjelmointi.
        - Tuotteen taustalla tapahtuvan logiikan ohjelmointi.
        - API rajapinnan kehitys.
        - Tietokantakutsut ohjelmistokoodissa.
    - Font-end puolella tapahtuva ohjelmointi.
        - Käyttöliittymän toteutus.
        - Käyttöliittymää tukevien aputoimintojen ohjelmointi.
- Bugien löytäminen ja korjaus.
- Uusien teknologioiden opettelu, jotta tuotetta voidaan laajentaa.

Olen yhteyksissä yhteistyökumppaniin Skype viestintäsovelluksen kautta lähes päivittäin ja keskustelemme tuotteen kehityksestä ja uusista toiminnoista. Käytännössä sovimme aikataulun, milloin kukin toiminto on valmis. Yritän keskittyä yhteen toimintoon kerrallaan. Välillä toimintojen priorisointi voi muuttua, jolloin minun pitää lopettaa tietyn toiminon toteuttaminen ja siirtyä toiseen. Projekti on erittäin joustava minua kohtaan, pystyn luomaan toiminnot alusta loppuun asti ja suunnittelemaan ne juuri siten, miten näen sen parhaaksi mahdolliseksi keinoksi.

Vastuullani on koko tuotteen tekninen toteutus. Saan apua käyttöliittymän visuaalisessa suunnittelussa yhteistyökumppanilta, lisäksi sovimme hänen kanssaan mitä ominaisuuksia tuotteessa tulee olla.

Olen toiminut yksityisyrittäjänä yli kuusi vuotta, joten koen kokemukseni olevan jo vahvasti ammattilaisen tasolla. Tunnen PHP ja JavaScript kielen rakenteet erittäin hyvin ja pystyn tuottamaan koodia itsenäisesti ilman ulkopuolista apua. Ohjelmoinnin opettelun aloitin itsenäisesti jo yli kahdeksan vuotta sitten. Opinnäytetyön alkaessa minulle oli tuttua jo lähes kaikki tekniikat mitä käytin kymmenen viikon seurannan aikana.

Opinnäytetyön aikana minulla oli vaikeuksia keskittyä omaan oppimiseen, kuten jo mainittu, niin kokemusta on useamman vuoden edestä, uuden oppiminen keskittyi lähinnä Kubernetes palvelinympäristöön ja Google Cloud alustaan. Opinnäytetyön aikana opittu Kubernetes osaaminen tuntuu vielä olevan perustasolla, olen joutunut kysymään ulkopuolisilta henkilöiltä apua tietyissä ongelmatilanteissa. Mutta pääasiassa olen selvinnyt jo aikaisemmalla kokemuksella ja ongelmanratkaisu on ollut osa arkipäivääni jo pitkään, joten olen tottunut haasteisiin. Kuitenkin oppimista riittää vielä varsinkin palvelinympäristön kanssa. Ongelmiin ei ole voinut varautua etukäteen, ne ovat tulleet täysin yllätyksenä minulle tähän mennessä.

Koen, että olen siinä vaiheessa ohjelmistokehittäjän uraa, jossa voin opastaa aloittelevia ohjelmoijia työtehtävissä, mikäli sellainen tilanne tulee vastaan.

Haasteena yrittäjänä työskennellessä koen erityisesti työtaakan, vaikka tykkäänkin työskennellä yksin vapaamassa ympäristössä. Se tuo silti ison taakan kehittäjän harteille. Olen yksin vastuussa siitä, jääkö ohjelmistoon haavoittuvuus, joka voisi riskeeraa arkaluonteisen tiedon vuotamisen ulkopuolisille tekijöille. Lisäksi joudun suunnitella kaiken yksin, miettiä hankalan logiikan toteutusta ja ennen kaikkea, tehdä tämän kaiken yksin turvautuen vain omaan osaamiseeni. Samalla tämä kaikki näkyy välillä stressinä ja motivaatio ja oma jaksaminen on ollut tiukalla.

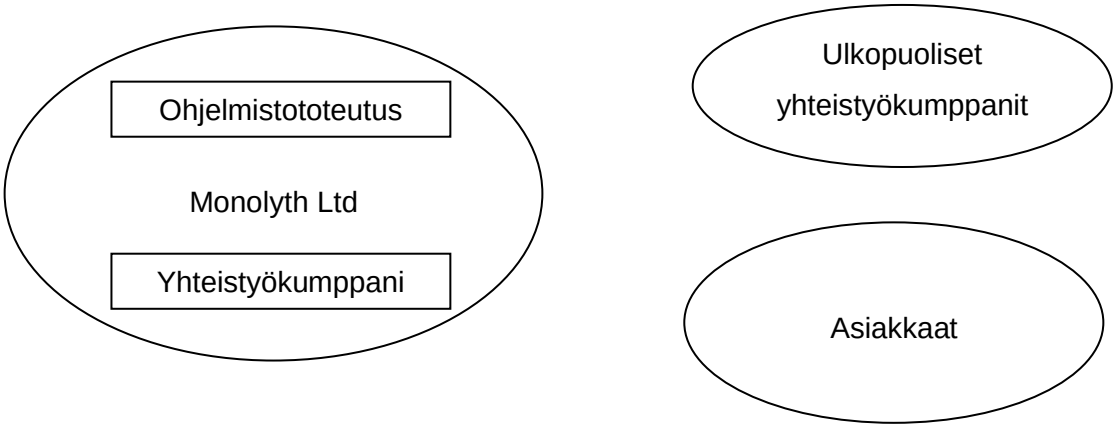
2.2 Sidosryhmät työpaikalla

Opinnäytetyön aikana työskentelen pelkästään Monolyth projektin parissa ja jätän kuvaamatta muut sidosryhmät, joiden kanssa olen ollut tekemisissä aikaisemmin. Tilanne perustuu pelkästään nykyiseen tilaani IT-yrittäjänä.

Monolyth yrityksessä on tällä hetkellä vain kaksi työntekijää. Ohjelmistokehittäjänä minä ja yhteistyökumppani, joka vastaa myynnistä, markkinoinnista, visuaalisesta suunnittelusta ja muiden yhteistyökumppaneiden kanssa tehdyistä sopimuksista.

Olen jatkuvasti yhteydessä kumppaniini, mutta en ole tekemisissä muiden yhteistyökumppaneiden kanssa. Kehitämme yhdessä tuotteen ominaisuuksia ideapohjalta, mitä voin tehdä, mitä en ja missä ajassa kehitys tapahtuu. Kaikki kommunikaatio on tapahtunut Skype viestintäsovelluksen avulla.

Ulkopuolisiin sidosryhmiin kuuluu kolme muuta yritystä. Yhden yrityksen kautta hoituu meidän kaikki sähköpostimarkkinointi ja lähtevän sähköpostin palvelut. Toisen yrityksen kautta hoidan ulkopuolisiin palveluihin kohdistuvat integrointitarpeet. Kolmas yritys on Google Cloud partneri ja olemme saaneet ilmaiseksi Google Cloudin kaikki palvelut rajattomasti usealle vuodelle.



Kuva 1 Monolyth Ltd sidosryhmät

2.3 Vuorovaikutustaidot työpaikalla

Olen vuorovaikutuksessa lähes päivittäin yhteistyökumppanin kanssa. keskustelemme tuotteen kehitykseen liittyvistä asioista lähinnä ja missä vaiheessa kehitystä ollaan menossa. Olemme yhteydessä pääasiassa Skypen kautta, mutta olemme juuri perustaneet Slack chattiryhmän, jonne liitämme lähitulevaisuudessa ulkopuoliset markkinointikumppanit. Suurimpia haasteita teknisen puolen toteutuksesta on yhteistyökumppanin kanssa ollut päätökset tuotteen ominaisuuksien päivityksistä ja muutoksista. On ollut tilanteita, jossa olen käyttänyt viikon jonkin ominaisuuden kehittämiseen, mutta olemmekin päättäneet myöhemmin, että olisiko toinen ratkaisu ollut parempi. Näin ollen on mennyt työtunteja hukkaan. En ole itse vaikutuksessa ulkopuolisiin markkinointikumppaneihin, olen vain tekemisissä heidän yrityksen tuotteiden kanssa ja integroinnissa osaksi meidän tuotetta.

3 Päiväkirjaraportointi

3.1 Seurantaviikko 1

Maanantai 18.2.2019

Ensimmäinen raportointipäiväni alkoi keskeneräisistä tehtävistäni Monolyth projektin parissa. Tehtävänä on luoda kampanjoiden sisäinen tiimihallinta, mikäli asiakas haluaa jakaa markkinointikampanjan työkavereiden kesken. Olin saanut valmiiksi jo aikaisemmin näkymän, jossa listataan kaikki asiakkaan hallitsemat tiimit, seuraavaksi oli vuorossa tehdä UI näkymä, eli käyttöliittymä tiimin hallintaan. Hallintanäkymässä pääsee lisäämään uusia henkilöitä, asettaa hänen oikeutensa, sekä muokkaamaan ja poistamaan jo lisättyjä henkilöitä.

Ennen kuin aloin työstämään käyttöliittymää, tein nopeasti uuden taulun tietokantaan, johon lisätään tiimin työntekijät roolin kanssa. Lisäsin taulun, jossa oli seuraavat arvot: *team_id, user_id, role* (read, write, admin). *Team_id* kolumni viittaa kyseiseen luotuun tiimiin, *user_id* viittaa henkilöön, joka on kutsuttu liittymään sekä *role* kolumni, mikä kertoo kuinka laajat oikeudet käyttäjällä, on kampanjan hallinnassa. Lisäksi tein taulun, joka seuraa onko käyttäjä liittynyt tiimiin, taulu sisältää myös aikaleiman, jolloin käyttäjä on lisätty, sekä sähköposti on lähetetty käyttäjälle tiimin liittymiseksi.

Tiimin hallintanäkymässä ensimmäinen työ oli lisätä lomake uuden käyttäjän lisäämiseksi. Asiakas voi lisätä tiimiinsä käyttäjän syöttämällä hänen sähköpostiosoitteensa. Lisättävän käyttäjän tulee olla jo rekisteröitynä palveluun ennakkoon, jotta lomake toimii haluamallani tavalla. Halusin tehdä autocomplete-lomakkeen, eli lomakkeen, joka syöttää ehdotuksia käyttäjistä haetulla sähköpostiosoitteella, jotta tiimin hallitsija näkee suoraan, onko sähköpostia olemassa järjestelmässä. Näin vältytään ärsyttävältä virheeltä, jossa tiimin hallinnoija saa virheilmoituksen ”käyttäjää ei löydy”.

Kun olin tehnyt lomakkeen, tein kontrolleriin metodin, joka validoi ja tallentaa syötetyt tiedot. Kun validointi eli tarkastus, jossa katsotaan käyttäjän syöttämät tiedot oikeaksi, oli valmis, kontrolleri ottaa yhteyttä tietokantaan ja lisä varmistaa, että kyseinen lisättävä henkilö todella löytyy järjestelmästä. Pelkkä käyttöliittymässä oleva varmistus ei takaa sitä, että käyttäjä syöttäisi oikeaa tietoa, näin vältytään mahdollisilta tietoturva-aukoilta. Kun kontrolleri on tehnyt vaadittavat tarkastukset, ottaa se yhteyttä tietokantaan vielä kerran ja lisää tiedot kantaan, jonka jälkeen sivulle lähetetään päivityskutsu ja käyttäjä palaa ilmoituksen kera, joka ilmoittaa, että käyttäjä on onnistuneesti lisätty tiimiin, mutta hänen

tulee vahvistaa sähköpostista liittyminen. Sähköpostin lähetykset teen vasta myöhemmin, koska minulla ei ole tällä hetkellä valmiina palvelinta, josta voisin lähettää testi sähköposteja.

Kun käyttäjän lisäys tiimiin oli valmis, tein kutsun, joka hakee kaikki tiimin käyttäjät ja lähettää ne listattavaksi käyttöliittymälle.

Tiistai 19.2.2019

Päivä alkoi tuttuun tapaan projektin käynnistämisellä, jatkoin siitä mihin jäin edellisenä päivänä, eli tiimihallintaan. Sain valmiiksi edellisenä päivänä listan, joka näyttää tiimin jäsenet ja statuksen, onko käyttäjä liittynyt tiimiin. Seuraava oli vuorossa tehdä listaan jokaisen käyttäjän kohdalle nappulat, jonka avulla voi poistaa käyttäjän tiimistä tai lähettää sähköpostikutsun uudelleen, mikäli käyttäjä ei ole vielä liittynyt tiimiin.

Lisäsin nappulat ja tein uuden JavaScript luokan, jota kutsutaan aina kun sivu ladataan. Olen tehnyt jokaiselle sivulle, jossa vaaditaan JavaScript toimintoja, oman luokan, jotta vältytään liiallisen JavaScript koodin kirjoittamiselta HTML sivulla. Lisäsin luokkaan metodin, jota kutsutaan, kun sivu on ladannut kokonaan. Metodin sisälle rekisteröin funktiot, jotka kuuntelevat, kun nappulaa painetaan, tästä taas jatkuu kutsu eteenpäin back-end puolelle, jossa varsinainen toiminto tapahtuu.

Kun käyttäjän poisto oli valmis tein vielä kolmen nappulan ryhmän listaan, jossa voi vaihtaa tietyn käyttäjän roolia. Kopioin ja muokkasin funktiota, jotka olin juuri tehnyt JavaScript luokkaan, jotta ne toimivat juuri tehdyille roolin muokkaus toiminnolle. Lisäksi tein back-end puolelle roolin muokkaus metodin, joka toteuttaa halutut toiminnot.

Tässä välissä siirryin nopeasti toisen projektin pariin, hallinnoin monia eri nettisivuja, jotka tuottavat tuloni mainoksien kautta, valitettavasti en voi kertoa kyseisestä sivusta muuta, kuin sen, että olen ylläpitänyt Suomalaista tuotearvosteluihin keskittynyttä sivustoa jo yli kuusi vuotta. On kulunut pari viikkoa siitä, kun aloitin kokeilun, jossa testaan mitkä mainokset toimivat tällä sivulla. En halua laittaa mainoksia liikaa sivuille, pelästyttääkseni kävijöitä pois, joten kokeilen pari viikkoa aina kerallaan mitkä mainokset toimivat parhaiten. Kirjauduin Tradetracker palveluun ja kävin etsimässä upotuskoodit uusille bannerimainoksille ja korvasin vanhat mainokset. Siirryin takaisin Monolyth projektin pariin.

Päivän toinen homma olisi saada tiimien hallinnassa näkyviin omat kutsut eli asiakas pääsisi näkemään itse, jos hänet on kutsuttu osalliseksi työskentelemään tiimin tai kampanjan kanssa. Tällekin sivulle olen jo valmiiksi tehnyt osoite reitityksen ja tyhjän sivun, jonka kanssa on helppo alkaa työskentelemään.

Aloin muokkaamaan sivun kontrollerin metodia. Tein tietokantahaun, joka kutsuu *team_invites* taulusta kaikki rivit, jotka liittyvät oman käyttäjän ID-numeroon ja ohjasin tiedot käyttöliittymä puolelle. Tein listan, joka näyttää juuri haetut tiedot sekä lisäsin nappulat ilman toiminnallisuutta valmiiksi huomiselle.

Keskiviikko 20.2.2019

Päiväni alkoi sillä, että tein Sublime Text 3 editoriin muutamat pikakomennot, jotka tekevät pieniä koodinpätkiä nopeuttaakseni omaa työskentelyä. Sublime Text 3 on erittäin nopea tekstieditori, joka on tehty Windows, Linux ja OS X käyttöjärjestelmille. Sublime Text 3 editoria voi laajentaa pikakomennoilla ja lisäosilla käyttämällä Python ohjelmointikieltä tai Sublimen omaa ohjelmointirajapintaa. Tein pikakomennot PHP Laravel frameworkille lähetetyn tiedon validoimiselle, sekä pikakomennot muutamien luokkien kutsumiselle. Luon päivässä n. yhdestä viiteen eri luokkia, joten välillä on turhauttavaa kirjoitella samaa koodia uusiksi, siksi pikakomennot auttavat tällaisen koodin tuottamisessa.

Päivän tavoite olisi saada visuaaliset näkymät kutsuttuihin tiimeihin valmiiksi, tiimiin liittyminen, tiimikutsun peruutus, tiimikutsun hylkääminen, sekä näkymä omista tiimeistä, johon käyttäjä on liittynyt.

Jatkoin edellisenä päivänä aloitettua näkymää tiimeistä, johon käyttäjä on kutsuttu. Tein eilen nappulat valmiiksi, jonka avulla voi painaa joko vihreätä tai punaista nappulaa, jolla hyväksytään tai hylätään kutsu. Tein uuden JavaScript luokan, johon kopion pohjan edellisestä luokasta, jonka kanssa työskentelin. Luokassa on kaikki samat toiminnot, jota tulen tarvitsemaan, joten minun tarvitsi vain muuttaa mitä funktiota kutsutaan ja mihin osoitteeseen ajax kutsut lähetetään. Tein tässä vaiheessa uuden kontrollerin ajax kutsuja varten. Tein kontrolleriin metodit tiimikutsun hyväksymiselle tai hyväksymiselle, yhdistin reitityksen kontrolleriin ja näin palvelin on valmis ottamaan kutsuja. Tämä prosessi on aika lailla itseensä toistavaa, jo siihen mitä olen aikaisemmin kuvaillut päiväkirjassa. Kun olin saanut tämän tehtävän valmiiksi, tein vielä uuden listanäkymän tiimeistä, joihin käyttäjä on liittynyt.

Vielä olisi hommana tehdä mahdollisuus lähteä liitytystä tiimistä pois. Mutta jätän sen huomisen päivän hommaksi.

Tässä vaiheessa kävin tarkastamassa, onko eiliset vaihdetut mainokset alkanut tuottamaan toisen projektin sivuilla ja päätin, että päivän työt on tehty.
Torstai 21.2.2019

Torstai aamu alkoi Skype palaverilla projektin toisen omistajan kanssa, keskustelimme noin tunnin ajan, miten optin lomakkeiden teko toteutuu käyttäjän puolella.

Optin lomakkeet ovat yksi palveluistamme, jossa asiakas voi kerätä sivustonsa kävijöitten sähköposteja markkinointitarkoituksiin. Palvelumme mahdollistaa visuaalisen lomakkeiden luonnin, integraation sähköpostimarkkinointi alustoihin. Yksi myyntivalteistamme on koneoppiminen yhdistettynä lomakkeiden optimointiin. Olen kirjoittanut järjestelmän, joka optimoi lomaketta automaattisesti asiakkaan sivustolle sopivaksi. Koneoppimiseen on yhdistettynä A/B testaus eri lomakkeen elementeille, kuten taustaväri, tekstin väri, lomakkeen teema, nappuloiden sijoitus ja väri, otsikon teksti ja lomakkeen muu teksti. Lisäksi lomakkeet voidaan integroida suosituimpiin sähköpostimarkkinointialustoihin.

Ongelmana on se, että alustaa voisi tulla käyttämään yksityishenkilöt ilman teknistä osaamista, sekä isot yritykset, jotka haluavat mahdollisimman kustomoitavia ratkaisuja. Päätimme, että tehdään visuaalinen avustaja uusien ominaisuuksien luonnissa, jossa voi erikseen päättää haluaako ottaa edistykselliset kustomoinnit käyttöön.

Keskustelimme samalla Social Proof-toiminnoista. Social Proof on yksi palvelumme markkinointiominaisuus, joka on pieni popup ilmoitus mikä näyttää käyttäjille livenä mitä sivulla tapahtuu. Toiminto voidaan asettaa näyttämään ilmoituksia esimerkiksi, kun käyttäjä rekisteröityy sivulle, tilaa tuotteen tai liittyy sähköpostilistalle. Muut samaan aikaan sivulla vierailevat saavat ilmoituksen sivun alakulmaan, esimerkiksi: ”Mikko M. tilasi tuotteen X, Suomi”, ”Mikko M. liittyi sivustomme jäseneksi”. Näin ollen muut kävijät saavat sosiaalista vaikutusta siitä, että muutkin käyttävät sivua ja rohkaisee käyttäjää liittymään myös sivulle. Päätimme jo alusta lähtien, että toiminto on aito ja sitä ei voi manipuloida lähettämään virheellistä tietoa. Emme halua, että asiakkaat markkinoivat sivuja kyseenalaisin keinoin esimerkiksi tekemällä ilmoitukset väärennetyllä datalla.

Palaverin jälkeen avasin Photoshop kuvanmuokkausohjelman ja tein uuden 1920x1080 kokoisen kuvan ja aloin suunnittelemaan visuaalista avustajaa. Päivä meni loppujen lopuksi Photoshop ohjelman parissa ja suunnittelin lomakkeita uusien toimintojen luomi-

seen. Tein yhden avustajan pohjan, jota voidaan käyttää pohjana kaikissa toiminnoissa. Tällä hetkellä kaikkien toimintojen lomakkeet ovat erittäin pelkistettyjä ja ei sovellu kuin omaan testaukseen.

Tänään piti tehdä toiminto tiimijärjestelmään, jotta käyttäjä voi lähteä pois jo liitytystä tiimistä, mutta päätin, että teen sen seuraavana päivänä vasta. Itse koodia en kirjoittanut tänään ollenkaan.

Viikkoanalyysi

Viikon aikana projekti edistyi suunnitellusti, sain tehtyä lähes koko tiimihallintajärjestelmän. Tekemättä jäi toiminto, jolla käyttäjä voi poistua liitytystä tiimistä, sekä back-end puolella toimiva logiikka, joka tarkastaa käyttäjän oikeudet, mikäli hän selaa projektia, johon hänet on kutsuttu. On tosin hiukan aikaista tehdä tuota logiikkaa, kun jaettua projektinäkymää ei ole vielä tehty.

Viikko oli hyvin normaali minulle, en joutunut opetella uusia koodikirjastoja tai työskentelyvälineitä, jotka olisivat auttaneet minua saavuttamaan viikon tavoitteet. Joten uuden oppiminen jäi pois kokonaan tältä viikolta.

Olen tyytyväinen, että sain tällä viikolla jo pitkään lojuneen asian pois alta lähes kokonaan, vielä tiimistä pois lähtö toiminto ja pääsen työskentelemään hiukan mielenkiintoisempien asioiden pariin. Tämä viikko oli lähes kokonaan vanhan koodipohjan kopiointia ja muokkausta, hiukan tylsää työtä. Ensi viikolla pääsen jo toteuttamaan uutta visuaalista avustajaa, jota voidaan käyttää jokaiseen kampanjan toiminnon luomiseen.

Viikolla kyllä huomaisi miksi on tärkeää suunnitella ja luoda selkeä koodipohja. JavaScript-luokat, joiden kanssa työskentelen, on suunniteltu alusta lähtien uudelleenkäyttö mielessä. Apumetodeja on paljon, eikä ne kasva kovin suureksi. Näin koodi on helposti luettavaa ja muokattavaa. Jos kaikki olisi kirjoitettu vain yhteen putkeen, viikon työmäärä olisi ollut huomattavasti suurempi.

3.2 Seurantaviikko 2

Maanantai 25.2.2019

Maanantaina jatkoin työskentelyä tiimihallinnan parissa, halusin tänään saada valmiiksi toiminnon, jolla käyttäjä voi lähteä liitytystä tiimistä, sekä aloittaa ulkoasupohjan visuaaliselle avustajalle, pohjautuen Photoshop suunnitelmaan.

Tiimihallinnassa oli jo näkymä listalle, jossa näkyy käyttäjän kaikki liitytyt tiimit, tein uuden nappulan kyseisen tiimin kohdalle sekä uuden JavaScript luokan. Kopioin taas vanhan pohjan luokasta. Tein luokan *onLoad* metodiin triggerin, joka kutsuu *removeFromTeam* funktiota, kun tiiminäkymälistassa painetaan, poistu tiimistä nappulaa. Tein funktioon vahvistuskutsun, joka kysyy käyttäjältä, haluaako hän varmasti poistua tiimistä ja myönteisen vastauksen saadessa, lähtee AJAX kutsu osoitteeseen, joka hoitaa poistamisen tiimistä. Viimeistelin toiminnon samalla, jota AJAX kutsuu, kun tiimistä poistutaan. Tässä vaiheessa tiimihallinta oli sellaisessa vaiheessa, että on hyvä jättää hienosäätö ja viimeistely myöhemmälle.

Aloin työstämään visuaalista avustajaa, joka auttaa vaihe vaiheelta käyttäjää luomaan optin lomakkeen, eli lomakkeen, jota käytetään sähköpostien keräämiseen markkinointitarkoituksessa nettisivulla. Tein uuden template tiedoston, jonka sivu lataa, kun mennään luomaan uutta lomaketta. Tässä vaiheessa on hyvä huomauttaa, että front-end käyttää jQuery kirjaston lisäksi Semantic UI front-end kirjastoa. Semantic UI sisältää erittäin kustomoitavan grid-järjestelmän, jolla sivuston sisältöä voi asetella erityisen yksityiskohtaisesti. Semantic UI framework mahdollistaa sivun jakamisen 16. osaan, kun taas luultavasti maailman käytetyin kilpailija kirjasto Bootstrap mahdollistaa vain 12. osaan.

Jaoin sivun kahteen osaan, toiselle puolelle tulee näkymä, jossa voi asettaa optin lomakkeen tiedot. Toiselle puolelle tulee esikatselunäkymä miltä optin lomake näyttää asetetuilla asetuksilla.

Semantic UI mahdollistaa välilehdillä erotellun sisällön. Loin viisi välilehteä, *1. details*, *2. layout, 3. theme*, *4. customization* ja *5. conversion*.

Ensimmäisessä välilehdessä annetaan optin lomakkeelle nimi ja valitaan lomakkeen typpi. Popup lomake, kokonäytön popup lomake tai sivuston ylä- tai alalaitaan kiinnittyvä palkki, joka sisältää lomakkeen. Layout välilehdessä käyttäjä voi valita millaisen asettelun lomake sisältää. *Theme* välilehdessä voi valita millaisen valmiiksi tehdyn teeman haluaa. Kustomointivälilehdessä taas on laajemmat asetukset teksteille, väreille ja muille asetuksille. Viimeinen välilehti pitää sisältää tiedot mitä tapahtuu, kun käyttäjä syöttää sähköpostin lomakkeeseen, saako hän viestin vai ohjataanko hänet tiettyyn verkko-osoitteeseen sen jälkeen.

Tein tuttuun tapaan uuden JavaScript-luokan optin lomakkeiden tekoa varten, joka tulee hoitamaan kaikki dynaamiset osat kyseisellä sivulla. Tein apumetodin *initEvents*, jota *onLoad* päämetodi kutsuu, kun sivu ladataan. Kyseinen metodi tulee lataamaan kaikki triggerit ja muut funktiot, jota kutsutaan aina, kun jotain visuaalisesti muuttuvaa toimintoa kutsutaan, kuten välilehden vaihtoa, teeman valitseminen ym. Laitoin aluksi metodiin pätkän koodia, joka laittaa välilehdet toimivaksi, lisäksi HTML puolella tein nappulat jokaisen välilehden alalaitaan, jolla pääsee seuraavalle välilehdelle ja rekisteröin JavaScript-luokkaan triggerin, joka kuuntelee, kun välilehteä halutaan vaihtaa.

Tiistai 26.2.2019

Tämän päivän suunnitelmana on jatkaa eilen aloitettua ohjattua visuaalista avustajaa optin lomakkeiden luontiin.

Huomasin jo eilen, että visuaalisesta avustaja tulee isompi työ, mitä olin alun perin ajatellut. Siihen tulee todella paljon muuttuvia osia. Esimerkiksi, kun lomakkeen ulkoasua vaihdetaan, teemat ja kustomoitavat asetukset tulevat muuttumaan. Lisäksi kun teemaa vaihdetaan, tulee kustomoitavien asetuksien sisältö vaihtumaan, sekä kun kustomoitavaa yksittäistä asetusta vaihdetaan, tulisi se päivittyä vasemmalla puolella olevaan esikatselu näkymään.

Aloin suunnittelemaan logiikkaa tämän kaiken pohjalle. Päädyin ratkaisuun, jossa kaikki esitäytetty data tullaan lataamaan yhdestä taulukosta, ulkoasun teemaan mukaan. Ensimmäisenä ladataan esitäytetty HTML-pohja valitulle ulkoasulle, jonka jälkeen ladataan asetukset teeman mukaan. Koska optin lomakkeet tukevat automaattista koneoppimista, niin se tulee ottaa huomioon jo vaiheessa, jossa esitäytettyä dataa ladataan.

```
themes: {
    newsletter: {
        header_text: [
            'Subscribe',
            'Subscribe to newsletter',
            'Join to mailing list'
        ],
        content_text: [
            'Subscribe now and get exclusive deals straight into your inbox.',
            'Join to mailing list to get news from our website.'
        ],
        input_placeholder: [
            'Your email',
            'Enter your email',
            'Enter email address',
            'Enter your email address'
        ],
        button_text: [
            'Subscribe',
            'Subscribe now',
            'Submit',
            'Join'
        ]
    },
    //...
}
```

Kuva 2. Lomakkeen esitäytetty data teeman mukaisesti.

Tässä vaiheessa tein ensimmäiseen välilehteen lomakkeen, jossa voi antaa nimen lomakkeelle, sekä valita ulkoasun kolmesta eri vaihtoehdosta: *Popup*, *fullscreen* ja *floating bar*.

Lisäksi tein JavaScript funktion, joka hoitaa logiikan, kun optin ulkoasu valitaan. Tein luokkaan muuttujan, joka sisältää kyseisen optin lomakkeen valitut asetukset, kuva 2 tyylisesti. Kun ulkoasu valitaan, varastoidaan teeman nimi kyseiseen muuttujaan. Tein funktiosta uudelleenkäytettävän ulkoasun ja teeman valitsemiselle.

Sain tänään optin lomakkeen luontiin jo paljon logiikkaa valmiiksi, lisäksi tein vielä toisen välilehden valmiiksi, jossa voi valita lomakkeen ulkoasun.

Keskiviikko 27.2.2019

Päivän tavoitteena on yrittää saada valmiiksi tällä viikolla aloitettu visuaalinen avustaja. Koska sivulla on niin paljon muuttuvia osia, esimerkiksi kun ensimmäisellä välilehdellä valitaan asetus, toisella välilehdellä sisältö muuttuu, niin ajattelin lisätä VueJS-kirjaston avuksi hoitamaan kaiken visuaalisesti muuttuvan sisällön. Käytännössä tämä tarkoittaa sitä, että VueJS-kirjastoon rekisteröidään elementti, joka tässä tapauksessa tulee kuuntelemaan tiettyä muuttujaa ja sen avulla päättää näytettävän sisällön.

VueJS on frameworkki, joka auttaa rakentamaan käyttöliittymiä, kirjasto tehtiin vuonna 2014 Evan You:n toimesta. Evan työskenteli Googlella paljon AngularJS kirjaston parissa, joka on samantyyppinen kirjasto kuin VueJS. Hän sai ideaksi rakentaa kevyen ja nopean kirjaston niistä Angularin osista, joista hän tykkäsi paljon. (First Week of Launching Vue.js)

Tällä hetkellä toinen välilehti, jossa valitaan lomakkeen ulkoasu, on staattinen HTML sivu, korvasin sen Vuen avulla dynaamiseksi sisällöksi, joka asetetaan näkyviin, kun sivu ladataan. Tein JavaScript luokkaan muuttujan, joka sisältää listana kaiken datan, joka syötetään toiselle välilehdelle. Kolmas välilehti, jossa valitaan lomakkeen teema, on lähes identtinen välilehti, kopion juuri tehdyt koodit ja muokkasin ne toimimaan kolmannessa välilehdessä.

Sain tehtyä tänään hyvin lomakkeen luontia eteenpäin, vielä puuttuu lomakkeen kustomoinnin välilehti ja viimeistely. Jatkan siitä huomenna.

Torstai 28.2.2019

Jatkan tänään siitä mihin eilen jäin. Seuraavana olisi vuorossa lomakkeen sisällön kustomointi.

Tein uuden Vue komponentin, joka kuuntelee muuttujaa siitä, mitä sisältöä kustomointi välilehdessä näytetään. Ajattelin, että on helpompi tehdä tämäkin Vuen avulla, koska kyseisen järjestelmän ylläpito jQueryn tai natiivin JavaScript koodin avulla tulisi olemaan työläs homma.

Tein välilehden valmiiksi testidatalla, jonka jälkeen aloin tekemään logiikkaa siihen, mitä tapahtuu, kun lomakkeen ulkoasua tai teemaa vaihdetaan. Ulkoasun ja teeman vaihdolle olin tehnyt jo aikaisemmin metodit, joita on helppo laajentaa. Tein uuden metodin, jota kutsutaan aina kun jokin asetus millä vain välilehdellä muuttuu. Metodi hoitaa lomakkeen asetusten muutokset ja samalla laitoin metodin päivittämään sisällön esikatselunäkymään.

Päivän aikana kuulin, että pääprioriteetti olisi seuraavaksi saada sivuston uusi ulkoasu valmiiksi, jotta yhteistyökumppani voisi aloittaa markkinoinnin betaa varten. Tavoitteena olisi saada beta kahden kuukauden kuluessa. Markkinointi olisi tarkoitus aloittaa kuukausi ennen betaa. Laitan lomakkeen luonnin tauolle siksi aikaa, kunnes saan etusivun valmiiksi. Jatkan siitä ensiviikolla.

Viikkoanalyysi

Viikon aikana koodia syntyi yli 1800 riviä pelkästään JavaScript luokkaan, joka hoitaa optin lomakkeen luonnin. Määrä on runsaasti enemmän mitä alun perin ajattelin. Huomasin kyllä jo viikon alussa, että urakasta tulee todella iso, muuttuvien osien takia. Toki on aina parempi, että ohjelmistokoodin voisi kirjoittaa mahdollisimman tiiviiksi, mutta sekin riippuu tilanteesta hyvin paljon. Jos on laaja ominaisuus, niin on myös ymmärrettävää, että koodiakin voi syntyä tuhansia rivejä.

Viikon aikana otin pitkästä aikaan VueJS kirjaston käyttöön, jota en ole käyttänyt projekteissa moneen kuukauteen, samalla pääsin tutustumaan uusimpiin ominaisuuksiin ja muutoksiin.

Viikon alussa oli ongelma siinä, miten tulisin näyttämään optin lomakkeen esikatselunäkymässä. Tavoite oli se, että se näkyisi samanlaisena kuin oikealla nettisivulla. Päädyin ratkaisuun, jossa loin uuden tyhjän sivun, jota käytetään pelkästään esikatseluun.

Esikatselunäkymään laitoin *iframe* koodin, joka lataa kyseisen tyhjän sivun ja aina kun asetusta muutetaan, laitetaan JavaScript luokka asettamaan *iframe* sivun sisällöksi lomakkeen renderöity HTML ja CSS tyylit. Normaalisti tämä ei olisi mahdollista, turvallisuuden vuoksi mutta, koska *iframe* lataa sivun, joka on osa samaa sivustoa, niin yläsivulla on rajoittamaton pääsy *iframe* sivun sisältöön. *Iframe* mahdollistaa samalla realistisen esikatselun lomakkeesta.

Harmikseni lomakkeen luonnin sivu jäi hiukan kesken. Viimeinen välilehti, jossa viimeistellään mitä tapahtuu, kun käyttäjä syöttää sähköpostiosoitteen, jäi tekemättä. Ensiviikolla siirryn hetkeksi toisiin työtehtäviin, tavoitteena olisi saada sivuston uusi ulkoasu kuntoon ja lisäksi viraalinen betaan liittyminen.

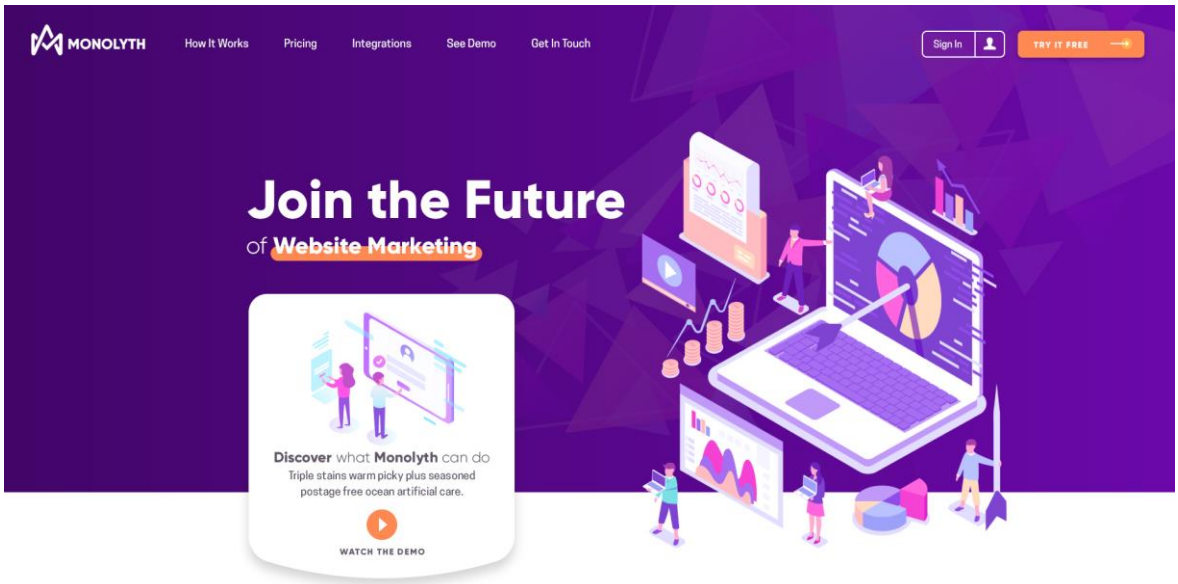
3.3 Seurantaviikko 3

Maanantai 4.3.2019

Työpäivän alussa:

Tämän päivän suunnitelmana on saada aloitettua sivuston uusi ulkoasu. Sain viimeviikolla *Photoshop* tiedoston kumppanilta, joka hoitaa visuaalisen suunnittelun tässä projektissa. Suunnitelmana on tehdä vastaava sivu toimivaksi tämän *Photoshop* suunnitelman pohjalta. Yhteistyökumppani oli ennen verkkosivujen suunnittelijana britti yrityksessä, jossa hän on oppinut käyttämään *Photoshop* kuvanmuokkaus ohjelmaa, ennen kuin siirtyi yrittäjäksi.

Päätimme noin kuukausi sitten, että nykyisestä sivuston ulkoasusta puuttui se jokin, emme olleet täysin tyytyväisiä ulkoasuun, joten päätös syntyi uudesta ulkoasusta silloin. Nykyinen ulkoasu on paljon selkeämpi, lisäksi uudet sisältötekstit tukevat uuden ulkoasun suunnitelmaa. Kumppanilla meni yli kuukausi uuden ulkoasun suunnitteluun ja se kyllä näkyy laadussa.



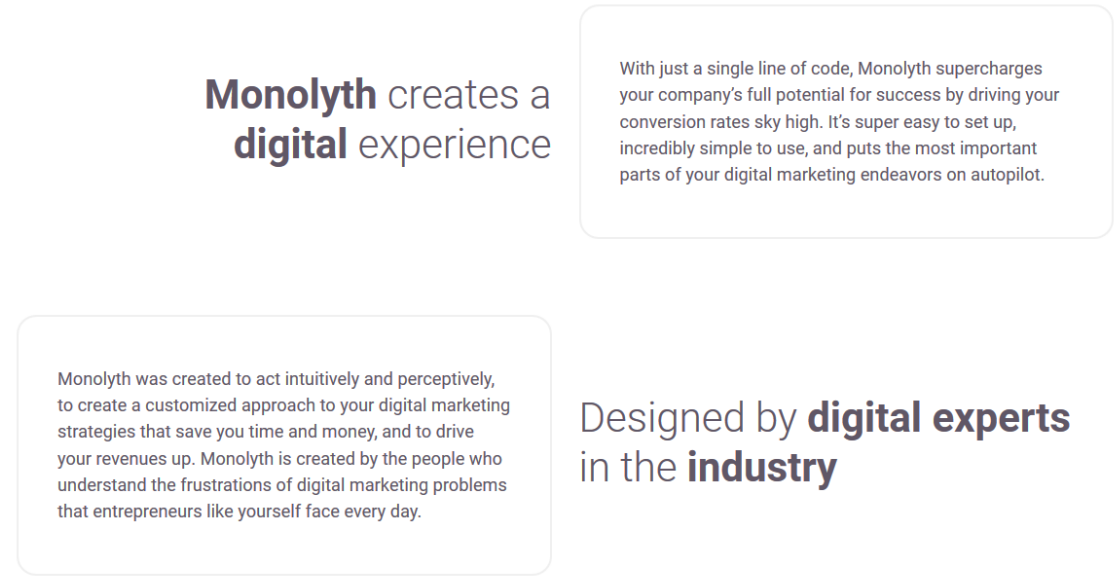
Kuva 3. Sivuston uuden ulkoasun header osio Photoshop suunnitelmassa.

Aloitin tekemällä kopion nykyisestä tiedostosta, joka sisältävät sivuston ulkoasun ja poistin kaiken sieltä.

Työpäivän lopussa:

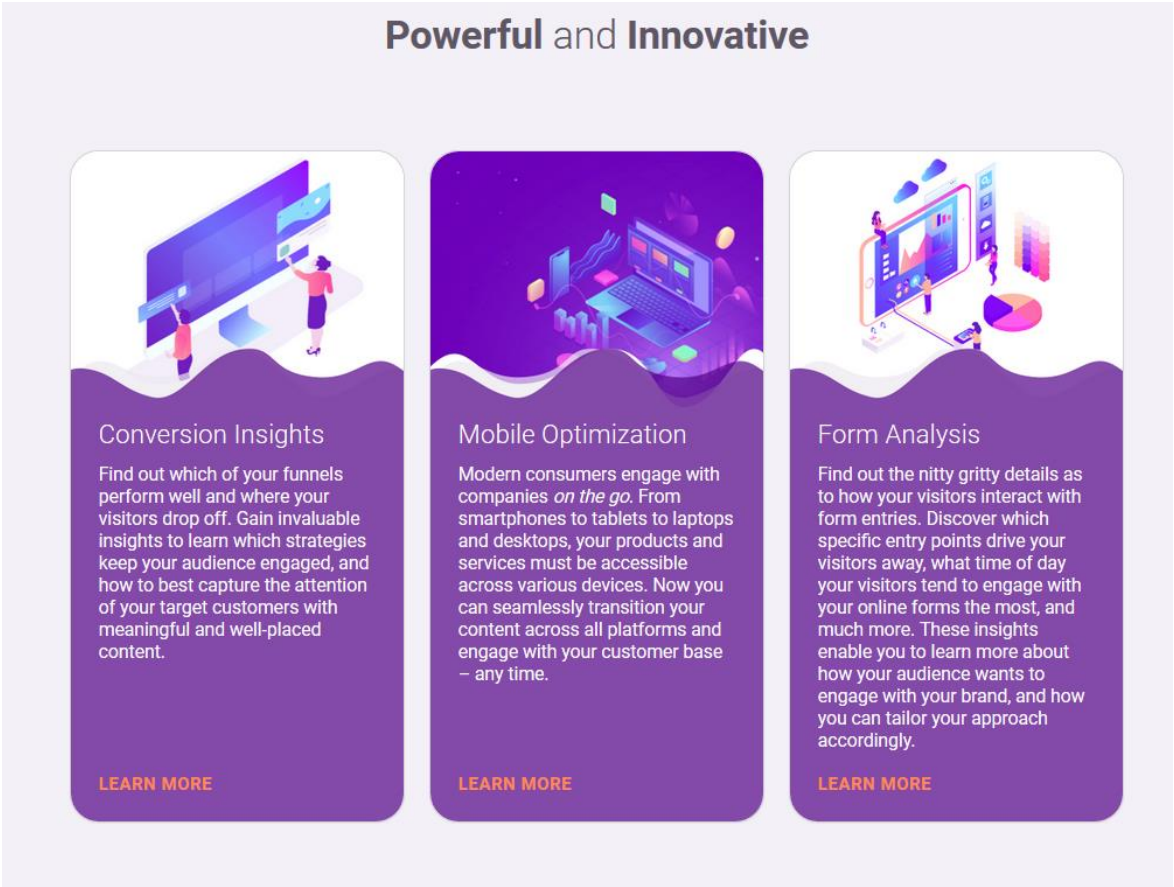
Pääsin tänään hyvin alkuun sivuston kanssa, sain tehtyä puolet sivun ulkoasusta. Vastoinkäymisiä ei päässyt tapahtumaan ja homma eteni suunnitellusti.

Sain päivän aikana tehtyä sivuston *header (kuva 3)*, *yhteenveto (kuva 4)* ja *pääominaisuudet (kuva 5)* osion. Yhteenveto osiossa kuvataan mitä sivustomme tekee ja kenelle se on tarkoitettu.



Kuva 4. Sivuston yhteenveto.

Työtaakkaamme keventääksemme, yhteistyökumppani oli palkannut copywriterin kirjoittamaan sisällön puolestamme. Kumppani toimii projektin rahoittajana, joten vaikka tuotteen kautta ei ole vielä tuloja, voimme nopeuttaa kehitystä hänen kauttaan.



Kuva 5. Kuvaus sivuston pääominaisuuksista.

Jatkan huomenna sivuston ulkoasun kanssa ja kokeilen saada sen valmiiksi.

Kuulin päivän loputtua jo, että yhteistyökumppani olisi löytänyt markkinointipartnerin, joka olisi kiinnostunut mainostamaan meidän tuotetta. Tavoitteena olisi saada sivusto nopeasti valmiiksi, jotta ennakkomarkkinointi beta julkaisua varten voitaisiin aloittaa. En itse tiedä mitä kaikkea diili sisältää yksityiskohtineen, mutta markkinointipartnerin kautta tuleva liikenne menee affiliate järjestelmän kautta, joka tulee kirjaamaan kaikki maksavat asiakkaat, jotka ovat rekisteröityneet heidän kauttaan. Joten maksamme komissiota heille, jokaisesta maksavasta asiakkaasta.

Keskiviikko 6.3.2019

Työpäivän alussa:

Pidin eilen välipäivän työnteosta, jatkan tänään siitä mihin maanantaina jäin. Tavoitteena olisi saada valmiiksi sivuston ulkoasu.

Avasin koodieditorin ja ryhdyin viimeistelemään sivustoa.

Työpäivän lopussa:

Sain tehtyä päivän aikana sivuston loppuun asti, vielä jäi muutamia juttuja tekemättä, kuten responsiivisuus mobiililaitteilla. Onneksi semantic ui kirjasto on täysin mobiiliystävällinen ja muutoksissa ei pitäisi mennä muutamaa minuuttia enempää. Päivä venyi yllättävän pitkäksi, koska hommaa oli todella paljon.

Vielä olisi tekemättä viraalinen betaan liittyminen, jonka jälkeen sivusto olisi valmis vastaanottamaan kävijöitä. Jatkan siitä seuraavalla kerralla.

Perjantai 8.3.2019

Työpäivän alussa:

Päivän tavoitteena olisi saada aloitettua viraalinen betaan liittyminen. Kyseessä on siis lomake, jolla käyttäjä voi halutessaan ennakko liittyä palveluumme, vaikka se ei ole vielä

auki julkisesti. Käytännössä se menee niin, että syöttämällä sähköpostin käyttäjä saa ilmoituksen, että hänet on liitetty beta-testaukseen mukaan ja samalla annetaan paikkanumero jonossa. Markkinointitaktiikkana on ilmoittaa, että päästämme sisään vain tietyn määrän henkilöitä, esimerkiksi 100 ensimmäistä ilmoittautujaa. Jakamalla linkin Facebookissa, Linkedinissä, Twitterissä tai lähettämällä kutsun sähköpostilla, eli jakamalla palveluamme eteenpäin, käyttäjä voi varmistaa paikkansa varmasti. Jos käyttäjä saa viisi ystäväänsä liittymään betaan kutsulinkin kautta, lähetetään hänelle viesti automaattisesta betaan hyväksymisestä. Kumppani löysi tämän viraalisen betan, eräs kilpailijamme on käyttänyt sitä erityisen hyvällä menestyksellä, joten on järkevää yrittää samaa markkinointitekniikkaa. Lisäksi, mainostaminen ja näkyvyys tulee teoriassa kasvamaan ilman menojen nousua.

Työpäivän lopussa:

Sain päivän aikana tehtyä lomakkeen, joka on valmis ottamaan sähköpostit vastaan, lisäksi tein seurannan, joka pitää kirjaa käyttäjän linkin kautta liittyneistä henkilöistä.

Toteutin seurannan käyttämällä evästeitä apuna. Kutsulinkki on muotoa https://monolyth.io/invite/{numero}, kun kävijä tulee sivulle linkin kautta, tehdään tarkistus, löytyykö osoitteessa olevaa numeroa tietokannasta, joka kertoo käyttäjän, kenelle linkki kuuluu. Seuraavaksi katsotaan, löytyykö IP osoitetta jo betaan liittyjistä ja mahdollisesti estetään liittyminen, mikäli IP osoite löytyy jo tietokannasta. Näin ollen voidaan estää tuplakäyttäjät henkilöiltä, jotka haluavat saada viisi henkilöä helposti linkkinsä kautta.

Lisäksi tein lomakkeeseen Facebook ja LinkedIn jakonapin, käyttämällä niiden omia JavaScript kehittäjille tarkoitettuja kirjastoja.

Jätän ensiviikolle lomakkeen viimeistelyn, jonka jälkeen olisi tarkoitus rakentaa palvelinta.

Viikkoanalyysi

Viikko oli erittäin tuottoisa, sain tehtyä suurimmaksi osaksi kotisivut ja viraalisen betaan liittymisen valmiiksi. Vielä olisi tekemättä pieniä hommia, kuten sähköpostituslistalle liittyminen ja integrointi sähköpostimarkkinointi alustalle. Lisäksi olisi myös tekemättä eväste ja yksityisyys sopimus sivut.

Pääsin viikolla kertaamaan miten Laravel 5 frameworkissa voi asettaa evästeen uudelleenohjauksen lomassa, samantyyppisen ratkaisun olen tehnyt ennen aikaisemmassa Laravelin versiossa.

3.4 Seurantaviikko 4

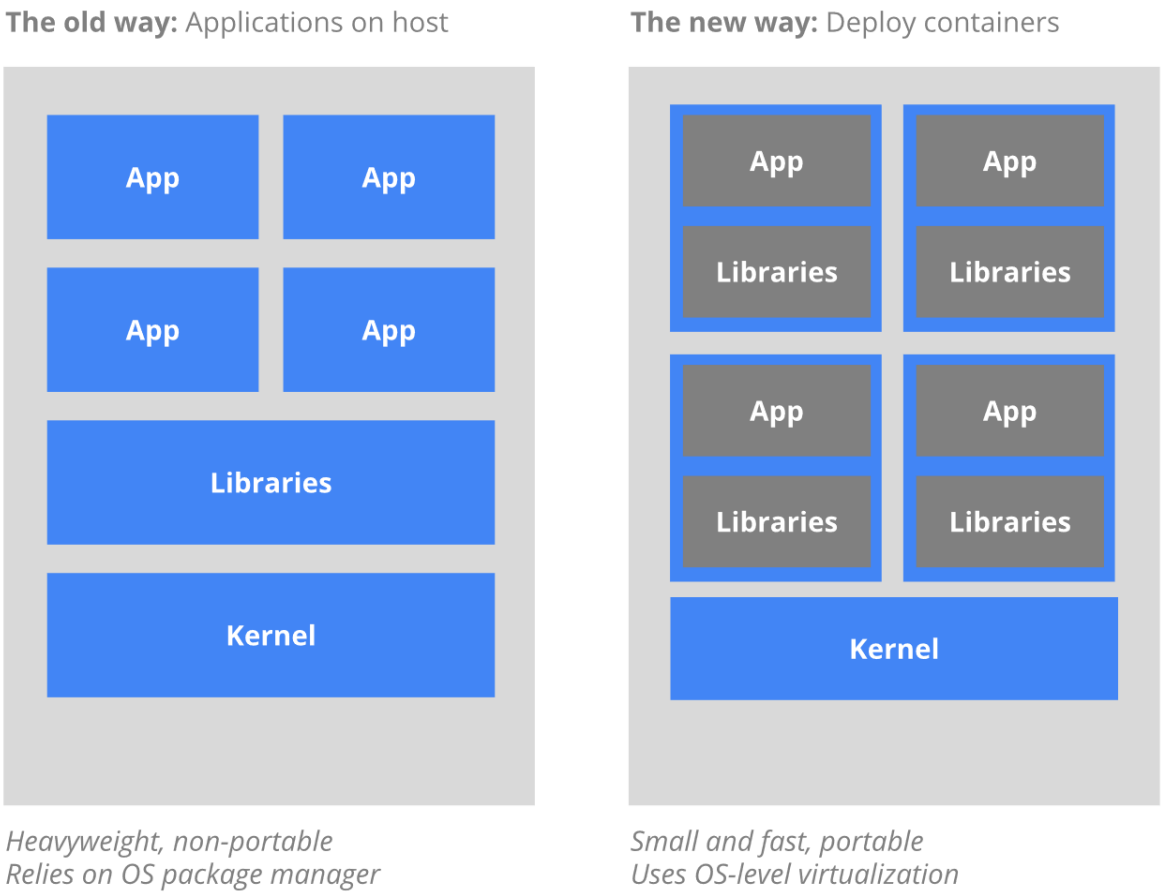
Maanantai 11.3.2019

Työpäivän alussa:

Seuraavien viikkojen tavoite on saada sivut siihen kuntoon, että voimme aloittaa ennakkomarkkinoinnin. Olen tehnyt jo aikaisemmin hiukan tutkimusta ja päätynyt Docker + Kubernetes ratkaisuun.

Docker on kehitysmenetelmä, jolla voidaan muuttaa tavallinen sovellus, web-applikaatio tai lähes mitä vain ajettavaa koodia kontrolloituun virtuaaliseen järjestelmään. Käyttännössä tämä tarkoittaa sitä, että kehittäjä voi muuntaa projektin Docker imageen, joka voidaan ladata palvelimelle ja palvelin ajaa sovellusta täysin samoilla asetuksilla virtuaalikoneen sisällä (Docker.com dokumentaatio).

Kubernetes on menetelmä, jossa ajetaan Docker paketteja yritys ympäristössä (Kubernetes.io dokumentaatio). Kubernetes mahdollistaa ja tekee helpoksi esimerkiksi sovelluksen päivityksen rolling update menetelmällä, jossa päivityksen sattuessa Kubernetes lataa päivitetyn Docker kuvan, käynnistää uuden palvelimen, ohjaa uuden liikenteen sinne ja loppuenlopuksi tuhoaa vanhan version. Kuberneteksen avulla kehittäjä voi myös asettaa automaattisen skaalautuvuuden sovellukselle. Mikäli sovellus saa yhtäkkiä esimerkiksi miljoonia kävijöitä tunnissa, Kubernetes luo lisäpalvelimia tai antaa enemmän prosessointitehoa ja RAM muistia sovelluksen käyttöön. Kun liikenne rauhoittuu, Kubernetes osaa säätää palvelimien määrän minimiin ja näin ollen samalla se säästää rahaa. (Sirish Raghuram, 2017).



Kuva 6. Docker ja Kubernetes palvelinympäristön visualisointi. (Lähde: kubernetes.io).

Kubernetes on vielä varsin uusi menetelmä, mutta suosittu ja hyväksi koettu. Päädyin Kubernetes ratkaisuun, koska halusin helpon ratkaisun tuotteelle, joka tukee automaattista skaalautumista ja mahdollisimman vaivatonta päivitystä. Kubernetes on rakennettu juuri sitä varten. Docker on vaatimus Kubernetes järjestelmälle. Kubernetes on tehty ajamaan Docker paketteja kuvan 6. mukaisesti.

Työpäivän lopussa:

Maanantai meni kokonaan oppiessa Kubernetes ympäristöä. Opettelin, miten se toimii ja mitä hyötyjä siinä on. Samalla tein Google Cloud tutoriaalin opastuksella ensimmäisen Kubernetes sovelluksen. Paljon tullut uutta asiaa yhden päivän aikana, enkä ole vielä varma, miten tämän saisi toimimaan meidän tuotteessamme. Jatkan tästä huomenna.

Tiistai 12.3.2019

Työpäivän alussa:

Päivän tavoite olisi saada Kubernetes ympäristössä Laravel frameworkin tyhjä asennus Google Cloud palvelimelle. Tämä siksi, että tuotteessamme on niin paljon lisäosia ja lisäksi se on riippuvainen Elasticsearch hakukoneesta, sekin pitää erikseen asentaa Kubernetes ympäristössä.

Tein Google haun ja etsin tuloksia ”*Laravel Kubernetes Nginx*” hakusanalla. Nginx on palvelin, joka tulee ajamaan PHP koodia.

Työpäivän lopussa:
Päivän tavoitteet jäivät toteutumatta. Etsin tietoa koko päivän eri Kubernetes asetuksista, miten palvelin kannattaisi rakentaa. Kaikki oppaat ovat tehneet asetukset hiukan eri tyylillä ja on vaikea päätyä sellaiseen mihin olisi itse tyytyväinen, koska ei ole kokemusta Kubernetesta.

Päivän lopussa löysin julkaisuvalmiin Kubernetes paketin Github palvelusta. Se sisältää automaattisen Travis CI integraation, joka ajaa automaattisesti kooditestit ja Github paketti sisältää asetukset, jotka ohjaavat Travis CI:n puskemaan päivitykset palvelimelle, mikäli testit ovat onnistuneet. Jatkan paketin kanssa työskentelyä seuraavalla kerralla.

Keskiviikko 13.3.2019

Työpäivän alussa:

Päivän tavoitteena olisi vihdoin saada Laravel frameworkin tyhjä asennus Google Cloudiin. Jatkan työskentelyä paketin kanssa, jonka löysin Github palvelusta.

Tein uuden Laravel asennuksen ja lisäsin sinne kyseisen Kubernetes paketin, jonka löysin.

Työpäivän lopussa:

Päivä oli hyvin tuottoisa siinä mielessä, että vaikka en saanut paljon näkyvää tulosta aikaseksi, sain palvelimen ajetuksi Google Cloudiin ja yllättävästi se meni sinne nätisti ilman suurempia murheita.

Kohtasin ensimmäiset ongelmat asennuksen kanssa, kun kokeilin tehdä päivitystä testiversiolla, jostain syystä, vaikka päivitys meni läpi, Kubernetes ei tajunnut päivittää uusimpaan versioon, vaikka sellainen olisi tarjolla. En keksinyt syytä, miksi se tekee tällä tavalla.

Päädyin laittamaan viestiä kaverille, joka työskentelee ohjelmistoyrityksessä, jossa on samantyylinen palvelinratkaisu käytössä. Hän sai työkaverilta vinkin mitä kokeilla. Kävi ilmi, että piti vaihtaa asetus nimeltä ”*PullPolicy*” arvoon ”*allways*”. Kyseinen asetus ohjaa toimintoa, milloin uusin versio Docker kuvasta ladataan. *Allways* asetuksen ollessa päällä, joka päivityksen yhteydessä Kubernetes lataa Docker palvelimelta uusimman version.

Pieniä ongelmia myös HTTPS salauksen kanssa, mietin pitkään miksi sivu ei lataudu, vaikka ilman salattua yhteyttä se toimi hyvin. Kävi ilmi, että vaikka Kubernetes asetuksista on portti 443 avattuna salatulle liikenteellä, pitää erikseen avata sama portti Google Cloud palomuurista.

Perjantai 15.3.2019

Työpäivän alussa:

Päivän tavoitteena olisi saada siirrettyä oikea versio sivustamme Google Cloudiin. Aloitin työskentelyn samalla tavalla kuin edellisenäkin päivänä, liitin Github paketin projektiin PHP:lle tehdyllä paketinhallintasovelluksella, composerilla.

Työpäivän lopussa:

Päivä jäi hiukan tynkäpäiväksi, tuli muuta menoa ja piti lopettaa työnteko tältä päivältä. Sain siirrettyä oikean tuotteen Google Cloudiin, lisäksi ostin Wildcard SSL sertifikaatin ja asensin sen palvelimelle. Nginx palvelinasetuksista oltiin määritelty, että HTTPS salaus käyttää itse todennettua salausavainta, joka on hyvä testikäytössä, mutta oikeassa tilanteessa se ei ole soveltavaa käyttää, koska selain saa varoituksen, jos selaa itsevarmennettua sivua.

Löysin samalla Cert Manager paketin, minkä pitäisi hoitaa ilmainen SSL sertifikaatti Letsencrypt palvelun kautta mutta päätin, että en ala näkemään vaivaa sen eteen, varsinkin kun Wildcard SSL sertifikaatin sai ostettua alennuksesta 40. dollarilla. Wildcard sertifikaatti toimii jokaisessa osoitteessa, jotka ovat liitettynä domainiin. Esimerkiksi sähköpostipalvelinta käyttäessä sertifikaatti suojaa myös https://mail.monolyth.io osoitteen.

Viikkoanalyysi

Viikosta tuli raskain tähän mennessä, ei työmäärän takia vaan uuden tiedon opettelun takia. Opin käyttämään Kubernetes järjestelmää, lisäksi tutustuin Travis CI palveluun.

Kun kaikki oli asennettu ja kunnossa, pääsin huomaamaan kuinka helppoa palvelimen päivitys voi olla automaation takia. Olen ennen hallinnut sivuja *cPanel* hallintasovelluksen kautta, joka on hyvä, jos pyörittää pienempiä sivuja halvoilla palvelimilla, mutta välillä liian työläs. Varsinkin kun pitää päivittää sivua.

Automaatio yhdistettynä Kuberneteksen kanssa, tekee päivityksen mahdolliseksi yhden komennon avulla.
Innolla odotan mitä ensiviikko tuo tullessaan.

3.5 Seurantaviikko 5

Maanantai 18.3.2019

Työpäivän alussa:

Viikko alkaa tuttuun tapaan Monolyth projektin parissa, tämän päivän suunnitelmana olisi muuttaa sivuston käyttäjäpuolen ulkoasua käyttäjäystävällisemmäksi.

Yhteistyökumppani teki viikonlopun aikana muutamia Photoshop suunnitelmia, miltä käyttäjäpaneeli ja sen sisältö voisi näyttää. Ulkoasu on samanlainen kuin nykyinenkin, mutta lähinnä fontit, ikonit, värit ja asettelu on muuttunut. Aloitin työt tekemällä uuden käyttäjäpaneelin ulkoasun.

Työpäivän lopussa:

Sain tehtyä päivän aikana käyttäjäpaneelin uuden ulkoasun, lisäksi sivupalkin navigoinnin niin tietokoneilla, kuin tableteilla ja mobiiliversiossakin.

Kävimme päivän aikana nopean palaverin ja sovimme ominaisuuksista ja mitä kaikkea tulisi olla tehtynä, kunnes olemme valmiina avaamaan ovet betaan liittyjille. Uutena ominaisuutena sivustolle on tulossa sähköposti-ilmoitukset tietyistä tapahtumista, kuten:

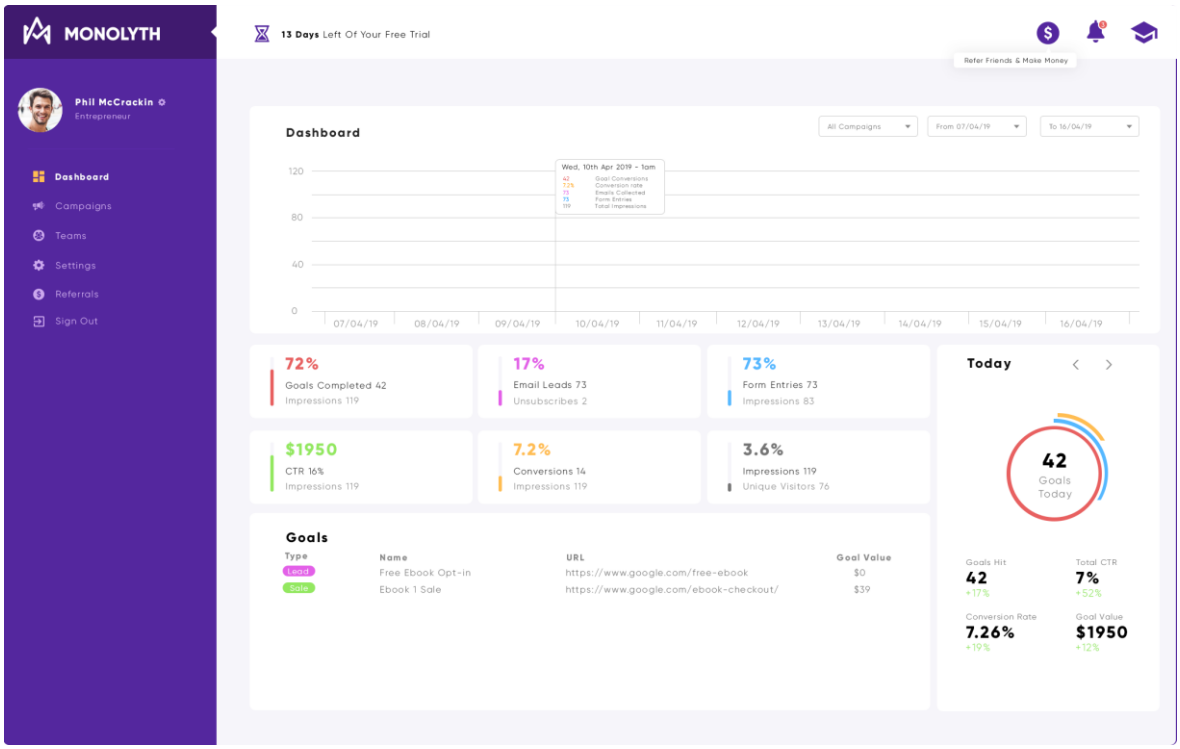
- Kampanjan konversio prosentit lähtevät huomattavaan nousuun
- Kampanja ei tuota konversioita ja lisäohjeet mitä käyttäjä voi tehdä korjatakseen tilanne
- Käyttäjä ei ole kirjautunut sisälle rekisteröitymisen jälkeen ja rekisteröitymisestä on kulunut X määrä päiviä
- Yleiset ilmoitukset kun kampanjassa tapahtuu muutoksia

Asiakas voi lisätä konversioseurannan kaikkiin palvelumme ominaisuuksiin. Konversio voi olla esimerkiksi sähköpostituslistalle liittyminen tai social proof toiminnon kautta tapahtunut myynnin rekisteröityminen.

Tiistai 19.3.2019

Työpäivän alussa:

Tiistai aamu jatkuu samalla hommilla, mihin eilen jäin. Tämän päivän työnä olisi tehdä käyttäjäpaneelin päänäkymä, jossa käyttäjä näkee suoraan yhteenlasketut kampanjoiden konversioprosentit. Päänäkymä tulisi luoda kuvan 7. Photoshop suunnitelman mukaisesti.



Kuva 7. Käyttäjäpaneelin päänäkymän suunnitelma.

Työpäivän lopussa:

Päivän aikana sain tehtyä päänäkymään lähes kaiken esitäytetyllä datalla. Ylläolevassa kuvassa näkyvä ympyrän muotoinen taulukkonäkymä jäi tekemättä. Muuten sain tehtyä sivun hyvälle mallille.

Ilman ongelmia päivä ei kuitenkaan mennyt, mittareiden sivussa näkyvien palkkien takia. Edistymispalkit ovat vaakatasossa vakiona ja pystypalkkia ei ole tehty HTML kirjastoon. Ainut keino tehdä pystypalkki on kääntää se CSS tyylitiedoston avulla.

```
.ui.stats.progress {
    -webkit-transform-origin: 0 100%;
    -webkit-transform: rotate(-90deg);
    -moz-transform-origin: 0 100%;
    -moz-transform: rotate(-90deg);
    -ms-transform-origin: 0 100%;
    -ms-transform: rotate(-90deg);
    transform-origin: 0 100%;
    transform: rotate(-90deg);
    width: 65px;
    float: left;
    position: absolute;
    margin: 0;
    padding: 0;
    bottom: 0;
    margin-bottom: 15px;
    margin-top: calc(100% - 10px);
}
```

Kuva 8. Edistyspalkin (progressbar) kääntäminen CSS tyylitiedoston avulla.

Kuvassa 8 on ratkaisu, jolla käänsin edistyspalkin vaakatasosta pystyasentoon. Maalatut rivit kuvaavat kääntämiseen tarvittavia asetuksia ja alla olevat ovat asetuksia, jotka korjaavat tiettyjä asetuksia, jotta palkki ja sen vieressä oleva sisältö menee vierekkäin nätisti.

Lisäksi tein JavaScript funktion, joka kutsutaan aina kun selain ladataan ja selaimen kokoa muutetaan. Funktio on asetettu päivittämään pystypalkkien korkeutta sopivaksi, mikäli näytön resoluutio kasvaa tai pienenee. Tämä siksi, että elementti, joka sisältää palkin samalla muuttaa kokoa ja pelkällä CSS tyylillä ei olisi saanut dynaamisesti muuttuvaa asetusta tälle.

Keskiviikko 20.3.2019

Työpäivän alussa:

Jatkan tänään käyttäjäpaneelin sivujen tyylien muuttamista uuteen tyyliin sopivaksi. Vielä olisi n. 10 eri sivua, jotka pitäisi muuttaa uudelle käyttöliittymälle sopivaksi. Lisäksi on paljon asetussivuja, jotka on lähinnä tehty minun omaan käyttöön, eivätkä ne sopisi maksavalle asiakkaalle. Puutteita ovat muun muassa puuttuvat kuvaukset mitä tietyt asetukset tekevät, epäselvät lomakkeet ja muutamissa sivuissa on myös asetuksia, jotka eivät ole enää käytössä.

Työpäivän lopussa:

Lähes kaikki sivut ovat muutettu uuden käyttöliittymän tyyliin. Vielä on paljon korjattavaa lomakkeiden kanssa, niihin en ehtinyt tänään koskea.

Kuulin kumppanilta tänään, että olisi kiire saada sähköpostipalvelin. Hän on onnistunut saamaan sopimuksen SendPulse palvelun kanssa ja onnistui saamaan meille ilmaiseksi premium tason sähköpostien lähetykseen ja markkinointiin tarkoitetun tilin moneksi vuodeksi. Olemme alussa riippuvaisia kyseisestä palvelusta ja jos saamme käyttää sitä ilmaiseksi tietyn ajan verran, eikä ongelmia synny. Niin tulemme jatkossakin käyttämään heidän palveluitaan, ilmaisen ajanjakson jälkeen. He auttavat samalla meitä pääsemään toiminnassa alkuun ja minimoimaan kulut. Molemmat osapuolet hyötyvät tilanteesta, heille tietysti hyöty tulee myöhemmin.

Perjantai 22.3.2019

Työpäivän alussa:

Eilisen sähköpostiuutisen kuultua, tämän päivän suunnitelmat muuttuivat sivuston päivityksestä sähköpostipalvelimen asenteluun. Suunnitelmana olisi saada Kubernetes alustalle sopiva sähköpostipalvelin.
Löysin sähköpostipalvelimen nimeltään Mailu, se on Docker paketti, joka sisältää useita sovelluksia, kuten Dovecot SMTP palvelimen, Postfix sähköpostin välitysohjelman, hallintapaneelin ja verkko-osoitteessa toimivan sähköpostin lukijan. Lisäksi Mailu paketti sisältää Kubernetes ympäristöön sopivan asennuspaketin.

Dovecot palvelin on sähköpostiohjelmisto, joka määrittää sähköpostipalvelimessa asetukset, miten ja missä sähköpostit varastoidaan. Postfix ohjelmisto taas välittää tulevat ja lähtevät sähköpostit ja keskustelee muiden sähköpostipalvelimien kanssa.

Aloin työstämään sähköpostipalvelimen asennusta.

Työpäivän lopussa:

Ajauduin palvelimen asennuksen kanssa ongelmiin heti alkujaan. Kävi ilmi, että Mailu palvelin vaatii jo aikaisemmin mainitun Cert Manager paketin toimiakseen. Koko päivä meni tutkiessa Cert Manager paketin Kubernetes dokumentaatiota. Ongelmaksi syntyi sen asennus palvelimelle.

Koko päivä oli ongelmaa ongelman perään, kun korjasi yhden vian niin toinen ilmestyi. Varmasti monet viat olisi onnistunut välttämään, jos olisi enemmän kokemusta Kubernetes ympäristöstä.

Cert Managerin asennus jäi tältä päiviltä valmistumatta, olisi tarkoitus jatkaa siitä ensi kerralla.

Viikkoanalyysi

Viikon aikana on tapahtunut paljon näkyvää edistystä Monolyth tuotteen parissa. Käyttäjäpaneelin tällä hetkellä sisältämät sivut ovat noin 70% käännetty uudelle tyylille sopivaksi.

Uutena tällä viikolla tuli myös Mailu sähköpostipalvelimen ja Cert Manager sertifikaattihallintaohjelman asennus. Tuotteessa ei itsessään tullut suurempia ongelmia, mutta Kubernetes ympäristö tuotti paljon päänvaivaa.

Cert Managerin asennus ei onnistunut virallisia dokumentaatioita seuraamalla, jouduin turvautumaan perinteiseen Googailuun ja löysinkin paljon apua, mistä viat yleensä johtuivat.

Monolyth Kubernetes hallintapaketti toimii Helm pakettimanagerin kanssa ja kävikin ilmi, että asentaakseen Cert Managerin, tulee käyttää eri Helm pakettia, mitä virallisessa dokumentaatiossa neuvottiin. Vasta sen jälkeen sain paketin asennettua Google Cloudiin, mutta onglemat eivät loppuneet siihen.
Kun Cert Manager oli vihdoin asennettu palvelimelle, tuli ongelmaksi Kubernes enginen automaattisesti skaalautuvat asetukset. Vaikka olin mielestäni asettanut palvelimen skaalautumaan, kun resurssit loppuvat, niin silti palvelut vilkuttivat punaista ja tuli ilmoitus, että liian vähän CPU ja RAM resursseja.

Ensi viikolla jatkan Kubernetes asetusten kanssa.

3.6 Seurantaviikko 6

Maanantai 25.3.2019

Työpäivän alussa:

Kuudes seurantaviikko alkaa Kubernetes asetusten kanssa työstämisellä. Haluisin tänään saada Mailu sähköpostipalvelimen asennetuksi. Ongelmia on vain tuottanut Cert Manager ohjelmisto.

Aloitin päivän korjaamalla ongelman mihin jäin viime viikolla. Kubernetes clusterille ei riittänyt CPU ja RAM resursseja. Google Cloud Kubernetes enginellä voi hallita helposti käyttöliittymän kautta lähes kaikkea mitä Kubernetes palvelin pitää sisällään. Lisäsin uuden palvelimen node pooliin, joka on valmiina ottamaan työtaakkaa vastaan heti kun sille on tarvetta.

Node pool on asetus Kuberneteksessa, joka kertoo mitä palvelinresursseja on tarjolla. Yksi node voi olla esimerkiksi 4 CPU ydintä ja 8 gigaa RAM muistia. Lisäksi asetuksista voidaan laittaa automaattisesti skaalautuvat asetukset, esimerkiksi minimi- ja maksimimäärä nodeja tietylle palvelinasetukselle. Tämän lisäksi Google Cloud tarjoaa beta vaiheessa olevan ominaisuuden, joka mahdollistaa nodelle lisä prosessoritehoa ja RAM muistia tarvittaessa ilman, että tarvitsisi käynnistää uutta nodea.

Työpäivän lopussa:

Luulin, että suurimmat ongelmat olisivat vihdoin takana. Sain Cert Managerin asennettua, mutta seuraavaksi tulikin ongelmia Mailu palvelimen kanssa. Sekin asentui tällä kertaa ensimmäisellä yrittämisellä, mutta ongelmaksi muodostui tiedostoja säilyttävän varastointitilan varaaminen.

Kubernetes on yhteydessä Googlen palvelimiin ja vaadittu tiedostojen säilyttämiseen tarvittava tila varataan etukäteen lähettämällä kutsu palvelimelle. Tällä kertaa kutsu ei mennyt läpi, koska Google Cloud Kubernetes Engine ei tue varastointitilaa, joka sallii useamman tiedoston lukemisen ja kirjoittamisen samaan aikaan. Googlaamalla löysin ratkaisun, jossa minun piti asentaa palvelimelle NFS (verkkolevyjärjestelmä) palvelu ja ohjasin tiedoston siirron kyseiselle palvelulle.

Seuraavaksi ongelmaksi syntyi pääsy Mailu palvelimeen käsiksi verkkoselaimesta. Jostain syystä, mennessäni palvelimelle osoitettuun osoitteeseen, sain ilmoituksen, että palvelinta ei löydy. Päätin lopettaa työt tältä päivältä ja päätin samalla, että en ala tuhlaamaan aikaa Kuberneteksen alla toimivaan sähköpostipalvelimeen, vaan alan etsiä vaihtoehtoisia ratkaisuja. Kubernetes jääköot käyttöön vain päätuotteen kanssa.

Tiistai 26.3.2019

Työpäivän alussa:

Päivä alkoi poistamalla kaikki viittaukset Mailu palvelimen vaatimiin paketteihin ja päivittämällä palvelin. Tällä hetkellä palvelimella pyörii vain verkkosivut, Redis välimuistin tallennuksen hoitava palvelin ja MySql palvelin, kaikki Kubernetes enginessä.

Päätin eilen, että menee liian hankalaksi yrittää laittaa kaikkea Kubernetes engineen. Varsinkin, kun jos kyse on sähköpostipalvelimesta, se ei tarvitse useita päivityksiä ja jatkuvaa huoltoa, joten sen voi asentaa virtuaalipalvelimelle erikseen koko Kuberneteksestä.

Löysin valmiin sovelluksen nimeltään iRedMail, joka sisältää lähes samat sovellukset kuin Mailu paketti. Pienenä erona on se, että verkkoselaimessa toimiva hallinnointipaneeli on eri ja iRedMailin mukana tulee Roundcube webmail sovellus, jonka kautta voi lukea sähköpostit selaimessa.

Tein Google Cloudissa uuden Linux Debian pohjaisen VM (virtual machine) serverin ja aloin asentamaan sähköpostipalvelinta.

Työpäivän lopussa:

Sain tänään laitettua sähköpostipalvelimen kuntoon ja lisäksi päivitin palvelimen DNS asetukset, jotta sähköpostia voidaan lukea ja lähettää mail.monolyth.io osoitteen kautta.

Koska Google on estänyt portin 25 palvelimiltaan, se tarkoittaa sitä, ettei sähköpostia voida lähettää miltään Googlen maksullisilta palvelimilta, pelkästään lukea, joten tässä vaiheessa astuu kuvaan SendPulse palvelu, johon kumppani sai meille ilmaisen premium jäsenyyden. Valitettavasti SendPulsen työntekijän tulee sallia SMTP (Simple Mail Transfer Protocol) asetukset tilille erikseen, ennen kuin sen kautta voidaan lähettää SMTP kutsuja. Tässä vaiheessa en voi muuta tehdä, kuin vain odottaa, että tilimme hyväksytään.

Torstai 28.3.2019

Työpäivän alussa:

Päivän alussa kävin katsomassa SendPulse palvelua, ovatko he hyväksyneet palvelimemme. Yllätykseni tilimme SMTP asetukset olivat nyt päällä ja seuraavana hommana olisi asettaa iRedMail sovelluksen kautta tuleva Postfix palvelin lähettämään lähtevät sähköpostit kulkemaan SendPulsen palvelun kautta.

Työpäivän lopussa:

Työpäivän aikana tein muutoksia sähköpostipalvelimeen ja pienen taistelun jälkeen sain sähköpostit kulkemaan SendPulse palvelun kautta. Tuli pieni ongelma asetuksia testatessa, sähköpostit eivät lähteneet kuten piti. SendPulse palautti viestin virheen kanssa, jossa kerrottiin, että autentikoiminen epäonnistui. Sain pienen tutkiskelun kautta selvitettyä, että olin asettanut relay-asetukset virheellisesti. Päivitin asetukset ja yritin uudestaan, nyt viestit kulkivat läpi niin kuin piti.

Perjantai 29.3.2019

Työpäivän alussa:

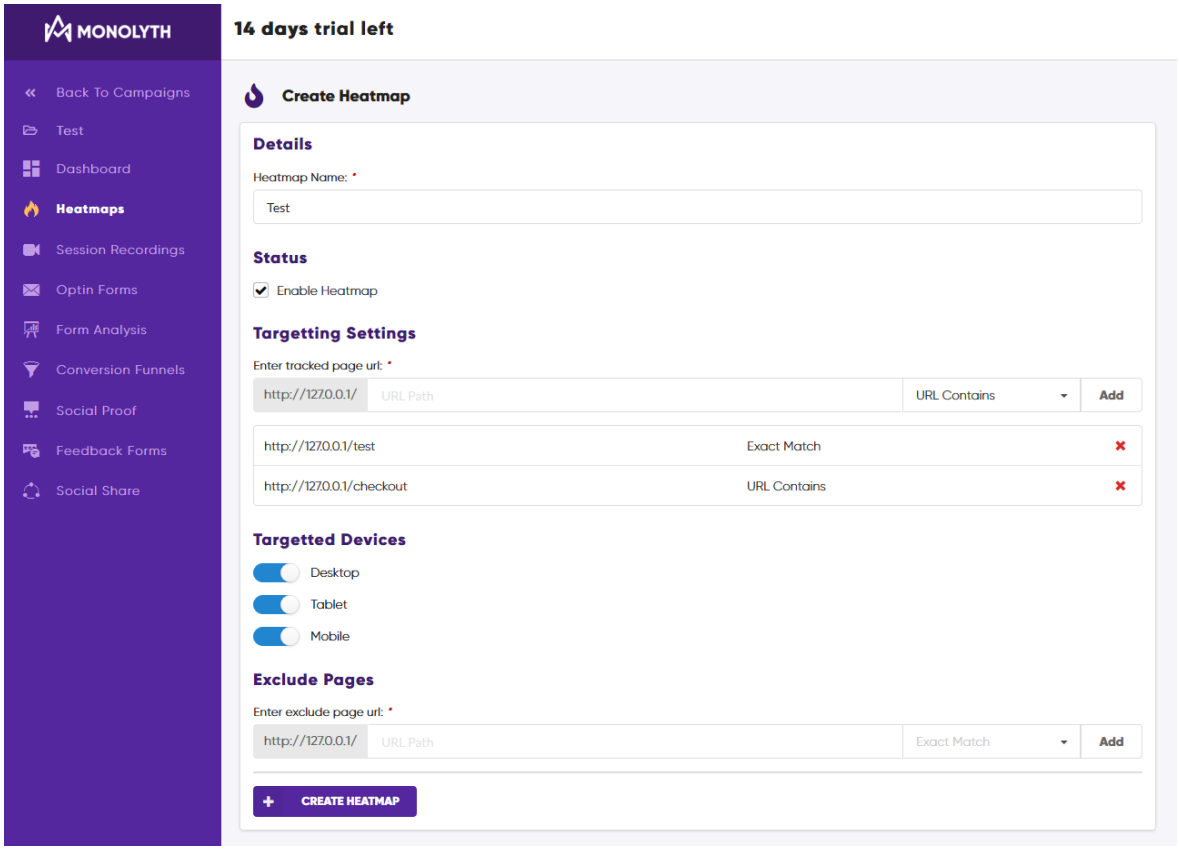
Tänään jatkan hommia sivuston kanssa, vielä olisi paljon tehtävää ennen kuin olemme valmiita julkaisemaan palvelun kaikkien nähtäville.

Ennen kuin aloin työskentelemään sivuston kanssa, kumppani ilmoitti, ettei hän saa hänen omaansa @monolyth.io sähköpostiinsa viestiä, jonka hän lähetti toiselta hänen omistamaltaan palvelimeltaan.

Tutkin asiaa, mistä tämä voisi johtua. Katsoin sähköpostipalvelimen lokeja, joissa ilmoitettiin estetystä osoitteesta. Jostain syystä palvelimen mukana tuleva harmaan listan manageri päätti, että kumppanin osoitteesta tuleva sähköposti on roskapostia ja esti sen automaattisesti. Yksi syy voi toki olla se, että hän käyttää jaettua palvelinta ja IP osoite, josta viesti lähtee, on estolistalla. Päädyin korjaamaan tilanteen ottamalla pois harmaan listan filtterin.

Työpäivän lopussa:

Sain työpäivän aikana muutettua muutamat sivut uuteen ulkoasuun sopivaksi, sekä tein lämpökarttaominaisuudelle uuden luontisivun (kuva 9). Sekä lisäsin ominaisuuden, jotta käyttäjä voi asettaa lämpökarttaseurannan yhdelle tai useammalle verkko-osoitteelle samaan aikaan. Lisäksi tein samanlaisen mustanlistan sivuille, mikäli käyttäjä ei halua seurata tiettyjä sivuja, jotka sisältyvät seurantalistaan.



Kuva 9. Lämpökarttojen luontisivu.

Viikkoanalyysi

Viikon aikana opin paljon uutta lähinnä Kuberneteksen kanssa työskentelystä, lisäksi pääsin ensimmäistä kertaa tekemään sähköpostipalvelimen Postfixin kanssa, olen ennen käyttänyt Sendmail ohjelmistoa. Postfix sovelluksen tarkoitus oli alun perin korvata Sendmail sovellus, sen vaikeakäyttöisyyden takia (Using Postfix for Secure SMTP Gateways, 2000). En muista paljoakaan Sendmailin kanssa työskentelystä, siitä on niin kauan, kun olen viimeksi joutunut asentamaan sähköpostipalvelimen itse, mutta muistan sen, ettei se ollut muutaman minuutin helppo asennus, kuten Postfix oli tällä kertaa.

Harmikseni muutama päivä meni aivan hukkaan Mailu sähköpostipalvelimen takia, huomasin samalla mitä ongelmia Kubernetes voi tuoda. Luin myöhemmin verkosta, että muutamat kehittäjätkin ovat samaa mieltä, ettei kaikkea kannata laittaa Kubernetes enginen sisälle. Yksi niistä on myös tietokanta, joka kannattaisi hoitaa hallitun palvelun kautta. Suunnitelmana ensi viikolla olisi päästä eroon Kubernetes enginessä toimivasta MySQL palvelimesta ja siityä käyttämään Google Cloudin tarjoamaan Managed SQL palvelua.

Google tarjoaa Cloud palveluna MySQL tai PostgreSQL palvelinta. Lisäksi sen mukana tulee automaattiset varmuuskopioinnit, käyttäjähallinta ja laskujeni mukaan se olisi lähes saman hintaista, kuin se, että hallinnoisin omaa MySQL palvelinta Kuberneteksen sisällä.

3.7 Seurantaviikko 7

Maanantai 1.4.2019

Työpäivän alussa:

Seitsemäs seurantaviikko jatkui lämpökarttojen parissa, sain tehtyä viimeviikolla ominaisuuden, jossa lämpökarttoihin voi lisätä useita seuranta- ja estosivuja. Seuraavaksi olisi vuorossa tehdä back-end toteutus, jotta ominaisuus oikeasti toimisi.

Työpäivän lopussa:

Työpäivä meni lähinnä lämpökarttojen parissa, lisäsin Elasticsearch indeksiin tiedot, jotta lämpökarttojen asetuksiin voidaan tallentaa tiedot URL osoitteista.

Yhtä elasticsearchin indeksiä voidaan verrata relaatiotietokannan tauluun. Elasticsearchissa tallennettavat dokumentit tallennetaan, eli indeksoidaan tiettyyn indeksiin, jolle voidaan ennalta määrittää dokumentin rakenne, eli tiedot mitä dokumentti tulee sisältämään. Elasticsearch on tyypitetty tietokantaratkaisu, mutta samalla sinne voidaan varastoida dokumentteja, joiden sisältöä ei välttämättä tiedetä ennalta tai sisältöä, joka muuttuu paljon. Kun elasticsearchin indexi rekisteröidään ja indeksiin lisätään tietoa, käyttäjä voi tehdä hakuja pelkästään niillä tiedoilla, mitä on ennalta määritetty indeksiin.

Lämpökarttojen lisäksi tein pieniä satunnaisia korjailuja, jotka ovat jääneet tekemättä jo pidemmältä ajalta.

Tiistai 2.4.2019

Työpäivän alussa:

Työpäivän tavoitteena olisi korjata eräs vika, joka lähettää tuplasti liikaa API kutsuja, kun käyttäjän sivulle asennettu seurantakoodi seuraa kävijän hiiren liikkeitä.

Olen tehnyt järjestelmän aikaisemmin, jonka pitäisi pitää huoli siitä, ettei API palvelimelle lähde liian montaa turhaa tai päällekkäistä kutsua. Jostain syystä, tekemäni kutsujen jonotus järjestelmä ei toimi. Ideana on se, että kun kävijä selaa asiakkaan sivuilla, seurantadata lähetetään järkevin väliajoin, eikä kutsuja saisi mennä päällekkäin, toisin sanoen edellisen kutsun pitää palauttaa API palvelimelta vastaus, ennen kuin toinen voidaan lähettää. Lisäksi olen tehnyt järjestelmän, joka katsoo, että kutsut lähtevät vasta sitten, kun uutta dataa ei tule muutamaan sekuntiin, mutta tämäkin on mennyt rikki jossain vaiheessa päivityksiä.

Työpäivän lopussa:

Ongelman korjauksessa meni hiukan enemmän aikaa mitä olin kuvitellut, kävi ilmi, että olin vahingossa poistanut edelliset apufunktiot, jotka pitävät huolen, ettei kutsuja lähde liian monta kerralla. Kirjoitin uudelleen koko käyttäjäseurantaluokan muutamia kuukausia sitten ja en ollut tehnyt perusteellisia testejä ennen kuin lopetin sen parissa työskentelyn.

Päivä meni uudelleen kirjottaessa puuttuvat metodit ja niiden testailussa.

Keskiviikko 3.4.2019

Työpäivän alussa:

Kuten jo aikaisemmin tuli mainittua, Google Cloud tarjoaa omaa Cloud SQL palveluaan. Päätin siirtää MySQL palvelimen toimimaan Google Cloud SQL:n kautta. Päivän tavoitteena olisi saada uusi tietokantaratkaisu toimivaksi.

Ongelmana on vain se, miten voin yhdistää Kubernetes palvelimelta Cloud SQL palvelimelle. Koska Kubernetes ajaa palveluita virtuaaliympäristössä Docker kuvina, ei ole mahdollista yhdistää lokaalilla IP osoitteella kuten normaalissa palvelinympäristössä. Tein hiukan tutkimusta ja luin Google Cloudin dokumentointia, että yhdistääkseen Cloud SQL palvelimelle, yksi keinoista on avata SQL Proxy Kubernetes ympäristössä, jolloin saadaan tekstimuodossa oleva ”IP” osoite, jonka kautta voidaan luoda yhteys.

Tutkiessa Kuberneteksen Github sivua, löysin heiltä Helm paketin, joka tekee proxy yhteyden Googlen Cloud SQL palvelimelle. Paketin asennus onnistui helposti lisäämälle asennuskoodi Travis CI production version ajettaviin koodeihin (kuva 10). Travis CI on asetettu tekemään ohjelmistotestit ennen kuin uudemmat päivitykset pusketaan Google

Cloudiin, samalla se on asetettu ajamaan vaadittavat koodit mitkä ajetaan Google Cloud palvelimella. Lisäsin SQL proxyn asennuspaketin jälkimmäiseen skriptiin.

```
28  # Install/Upgrade Google Cloud SQL Proxy
29
30  helm repo add rimusz https://charts.rimusz.net
31  helm repo update
32
33  helm upgrade pg-sqlproxy rimusz/gcloud-sqlproxy --namespace="$RELEASE_NAME" \
34      --set serviceAccountKey="$(cat ./tools/secrets/service-account.json | base64 | tr -d '\n')" \
35      --set cloudsql.instances[0].instance="mysql" \
36      --set cloudsql.instances[0].project=[redacted] \
37      --set cloudsql.instances[0].region="us-central1-a" \
38      --set cloudsql.instances[0].port=3306 -i
39
```

Kuva 10. Google Cloud SQL Proxyn asennus Helm pakettimanagerilla. Google Cloud projekti id sensuroitu.

Työpäivän lopussa:

Sain päivän aikana luotua tietokantayhteyden SQL proxylla. Samalla siivosin Kubernetes dokumenteista viittaukset alkuperäiseen Kuberneteksen alla ajettavaan MySQL palvelimeen ja ajoin uusimmat päivitykset. Tällä tavoin ajoin alas alkuperäisen MySQL palvelimen ja samalla poistin vanhat tietokannan tiedot Googlen palvelimilta.

Perjantai 5.4.2019

Työpäivän alussa:

Tänään kävimme muutaman tunnin palaverin yhteistyökumppanin kanssa, keskustelimme lähinnä aikataulusta, milloin olemme valmiita ennakkomarkkinointiin. Lisäksi hän oli onnistunut hoitamaan meille uusia yhteistyökumppaneita ja pääsyn heidän tuotteisiinsa. Lisäksi hän oli onnistunut löytämään meille Google Cloud partnerin, jonka kautta saimme useita kymmeniä tuhansia krediittejä Google Cloud alustaan. Iso stressiä aiheuttava ongelma saatiin samalla ratkaistua, enää meidän ei tarvitse huolehtia palvelinmaksuista useampaan vuoteen ja voimme keskittää enemmän varoja markkinointiin.

Työpäivän lopussa:

Varsinaista suunnitelmaa minulla ei ollut tänään työskentelyn suhteen. Päädyin refaktoroimaan paljon koodia järkevämpään muotoon.

Olen tehnyt viimepäivinä paljon uutta koodia nopeasti, samalla se tarkoittaa sitä, ettei se ole parhainta luettavaa. Vielä kun on tuoreessa muistissa mitä on tehnyt viime aikoina, niin oli hyvä refaktoroida ja järjestää ohjelmistokoodia omiin luokkiin ja apumetodeihin.

Viikkoanalyysi

Viikko meni pääasiassa vanhan koodipohjan päivittämisessä ja koodin refaktoroimisessa. Refaktorointi on prosessi, jossa koodia muutetaan luettavampaan muotoon. Refaktoroimisessa ei ole kyse ohjelmistobugien tai uusien ominaisuuksien lisäyksestä, vaan pelkästään vanhan koodin uudelleenjärjestämistä, jottei se olisi niin epäselvää luettavaa. Refaktorointiin voi tutustua kirjallisuudessa Martin Flowlerin teokseen *Refactoring. Improving the Design of Existing Code. 1999.*

Pieniltä haasteiltakaan ei vältytty, mutta suurempia ongelmia ei viikon aikana tullut.

Sain korjattua jo pitkään vaivanneen bugin kävijäseurannassa, jossa JavaScript luokka lähetti liian paljon API kutsuja. Ongelmana oli muutamat puuttuvat funktiot, jotka olivat ennen paikallaan, mutta olin poistanut ne epähuomiossa, kun uudelleenkirjoitin koko luokan muutamia kuukausia sitten.

Lisäksi sain siirrettyä Kubernetes enginen alla toimivan MySQL tietokannan käyttämään Google Cloud SQL palvelun tarjoamaa MySQL tietokantaa.

Oma oppiminen on kehittynyt viikon aikana hurjasti varsinkin Kuberneteksen kanssa työskentelyllä. Muistan jo helposti tärkeitä komentoja katsomatta muistiinpanoja ja olen oppinut muutaman viikon aikana tutkimaan Kubernetes lokeja, löytämään ongelman aiheuttajia ja korjaamaan niitä. Tämän huomasi jo Cloud SQL proxyn asennuksessa, paketin asennus onnistui helposti ja eikä tarvinnut epäröidä miten se asennetaan ja miten sitä käytetään.

3.8 Seurantaviikko 8

Tiistai 9.4.2019

Työpäivän alussa:

Kahdeksannen seurantaviikon tavoite on tehdä kokonaan uusi optin lomakkeiden ohjattu luontisivu. Käytin tähän jo aikaisemmin paljon aikaa, mutta sivu ei vieläkään miellytä silmää ja se vaikuttaa hiukan vaikeakäyttöiseltä. Tuntuu, ettei nykyinen sivu ole sellainen, jota kaikki asiakkaat osaisivat käyttää. Isoina eroina on esimerkiksi välilehdissä hiukan muuttuva tyyli ja muutamia epäselvyyksiä, miten osa toiminnoista toimivat. Hyvä tuuri kävi

siinä mielessä, että suuri osa logiikasta on jo tehty valmiiksi, vain ulkoasu muuttuu ja osan koodipohjasta kirjoitan uusiksi. Tällä hetkellä koodipohja on hiukan vaikealukuista muutamissa osissa.

Työpäivän lopussa:

Päivän aikana ei juuri suurempia muutoksia tapahtunut. Sain suunniteltua Photoshopissa uuden raakile mallin optin lomakkeiden luontisivusta ja sain tehtyä tyhjän pohjan valmiiksi.

Tein varmuuskopiot vanhasta koodipohjasta ja korvasin palvelimen lataamaan sisällön uusista ulkoasun sisältävistä tiedostoista.

Kuva 11. Optin lomakkeiden tyhjä luontisivu.

Keskiviikko 10.4.2019

Työpäivän alussa:

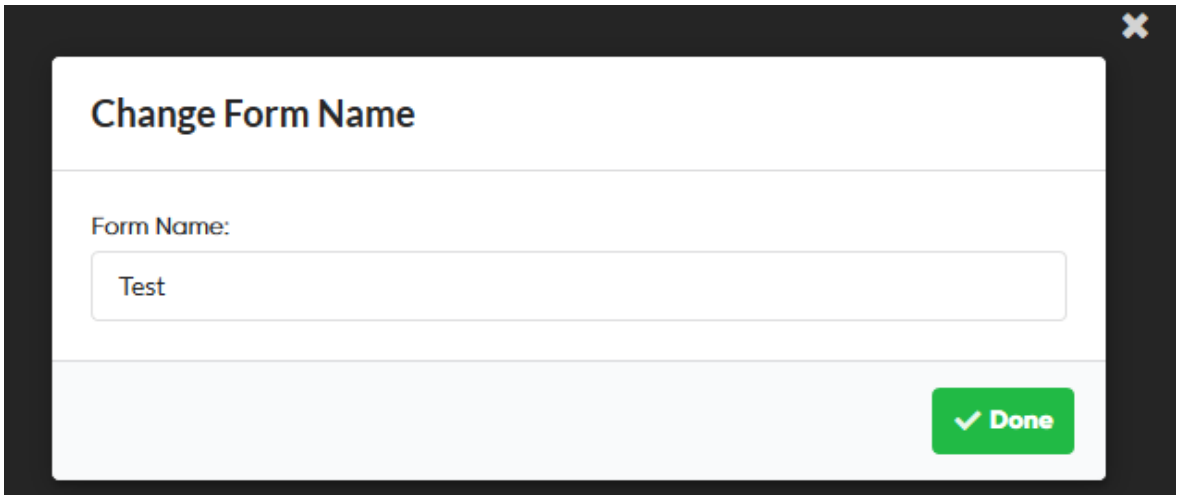
Työpäivä alkoi optin lomakkeiden parissa, päivän tavoitteena olisi saada mahdollisuus antaa nimi optin lomakkeelle ja saada lomaketta tehtyä sen verran mitä ehtii.

Edelliseen lomakkeen ulkoasuun verrattaessa, uudessa välilehdet ovat sivun vasemmassa reunassa allekkain. Kun edellisessä teemassa sivu oli jaettu puoliksi ja vasemmalla puolella oli verkkoselainten tyylinen välilehti ratkaisu.

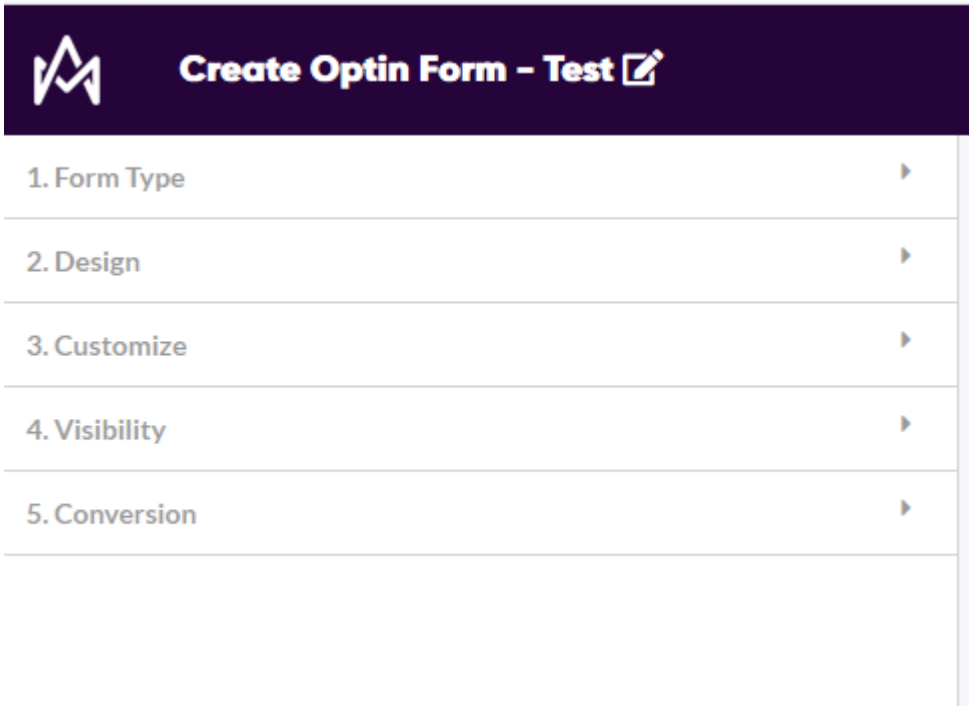
Uusi on visuaalisesti modernimpi ja käyttäjäystävällisempi. Vanha ulkoasu oli edelleen liian tekninen ja epäselvä, varsinkin henkilöille, jotka eivät ole tottuneet käyttämään sivuja, jossa on monta asetusta listattuna allekkain. Se voi tuntua liian raskaalta joillekin käyttäjille. Uusi tyyli jaottelee selvästi missä kukin ominaisuus on ja ne on helppo löytää omista välilehdistä. Lisäksi teen järjestelmän, jossa optin lomakkeen yksityisiin asetuksiin, kuten otsikon väriin ja kokoon, pääsee käsiksi klikkaamalla visuaalisesti näkyvää lomaketta. Joten vanha tyyli, jossa kaikki asetukset ovat allekkain listattuna menee pois.

Työpäivän lopussa:

Työpäivän aikana optin lomakkeiden luontisivu edistyi mallikkaasti eteenpäin. Sain tehtyä paljon toiminnallisuutta lomakkeen luontisivuun (kuva 11). Sain tehtyä tarvittavat välilehdet sivuston vasempaan reunaan (kuva 13), lisäksi mahdollisuuden antaa nimen optin lomakkeelle ja muokkaustoiminnon sille (kuva 12).



Kuva 12. Optin lomakkeen uudelleen nimeäminen.



Kuva 13. Optin lomakkeen luontisivun vasen sivupalkki ja sen välilehdet.

Torstai 11.4.2019

Työpäivän alussa:

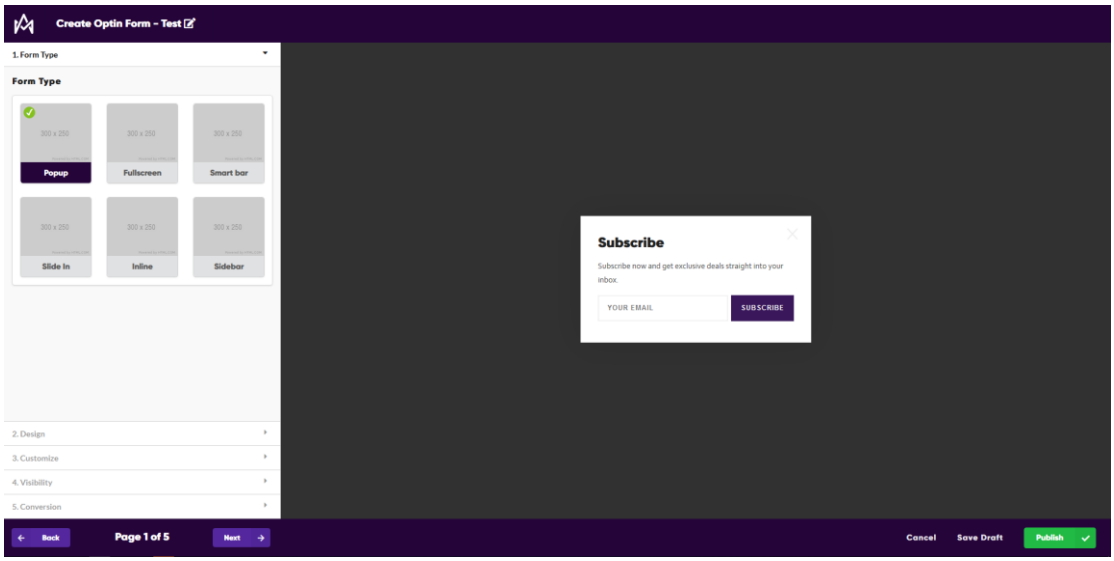
Työpäivän tavoitteena on jatkaa optin lomakkeiden luontisivun uudelleentekoa. Näyttää siltä, että työtaakasta on tulossa isompi, mitä ajattelin. Ajattelin, että kirjoitan koko koodipohjan uusiksi, lähinnä sen takia, että edellisessä versiossa jouduin turvautumaan paljon esitäytettyyn HTML koodiin lomakkeessa ja tämä tuo omia ongelmia, kun lomakkeessa on paljon muuttuvia osia. Vaikka osa välilehdistä latasi sisällön käyttäen Vue kirjastoa, päätin olla käyttämättä sitä tällä kerralla ja päätyä pelkästään natiiviin JavaScript / jQuery ratkaisuun.

Halusin dynaamisen ratkaisun, jossa kaikki muuttuva sisältö ladataan JavaScript luokan avulla.

Työpäivän lopussa:

Päivän loputtua, sain tehtyä ensimmäisen välilehden lähes valmiiksi. Suunnittelin JavaScript luokkaan metodin, joka lataa listan elementeistä, jotka ladataan selaimen näkymään dynaamisesti.

Samalla tein apumetodin, mitä kutsutaan aina, kun lomakkeessa tehdään valinta. Apumetodi lataa kävijälle näkyvän optin lomakkeen HTML koodin ja näyttää sen esikatselunäkymässä oikealla puolella sivua (kuva 14).



Kuva 14. Optin lomakkeen ensimmäinen välilehti ja esikatselunäkymä.

Perjantai 12.4.2019

Työpäivän alussa:

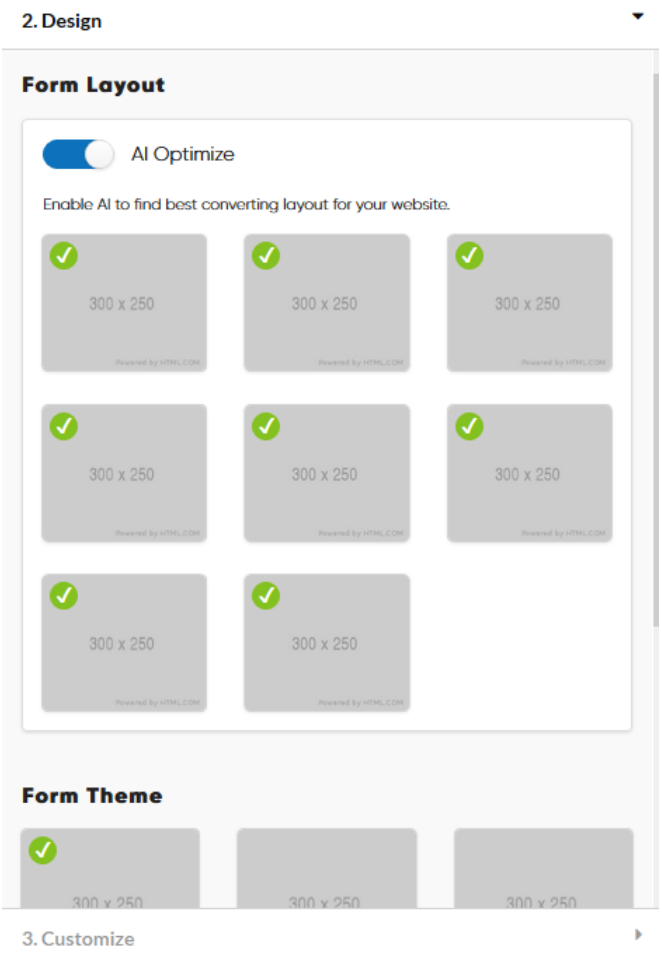
Tänään tehtävänä oli tehdä seuraava välilehti optin lomakkeisiin, enemmänkin mikäli ehtii.

Seuraavan välilehden idea on lähes sama kuin ensimmäisenkin. Ensimmäisessä välilehdessä valitaan optin lomakkeen tyyppi, eli onko se popup lomake, kokonäytön lomake, sivun ylä- tai alalaitaan kiinnittyvä palkki jne. Toisessa välilehdessä taas voi valita tyylin ja teeman. Lisäksi tyylille voi vapaaehtoisesti laittaa päälle itseoppivan ominaisuuden.

Itseoppivan ominaisuuden päälle laittamista en ole vielä tehnyt, päivän tavoitteena olisi laajentaa JavaScript luokkaa ja tehdä kyseisen ominaisuuden lisääminen.

Työpäivän lopussa:

Sain tehtyä tänään toisen välilehden kokonaan, lisäksi tein itseoppivan ominaisuuden valitsijan, joka päällä ollessaan antaa käyttäjälle mahdollisuuden valita monta optin lomakkeen ulkoasua. Samalla laajensin JavaScript luokkaa ja tein funktion, jota kutsutaan aina kun hiiri on valitun sivupalkin valintaelementin päällä. Kyseinen metodi näyttää oikealla puolella esikatselunäkymässä, miltä kyseinen asetus näyttäisi oikeasti (kuva 15).



Kuva 15. Optin lomakkeen teeman valitsija itseoppiva ominaisuus päällä.

Viikkoanalyysi

Viikko kului nopeasti ja pääsin palaamaan vielä kertaalleen optin lomakkeiden pariin. Tein viikon aikana uuden JavaScript luokan, joka hoitaa kaiken optin lomakkeiden tekoon liittyvät tarpeet ja aloin rakentamaan ominaisuutta uusiksi.

Päätin, että kirjoitan luokan kokonaan uusiksi, vaikka olin jo aikaisemmin päätynyt tekemään varsin hyvän JavaScript luokan, jota olisi voinut laajentaa ja muokata helposti. Mutta entinen lomakkeen luontisivu turvautui varsin paljon esitäytettyyn HTML koodiin, joka olisi tehnyt päivittämisestä hiukan hankalaa, koska sivu muuttui täysin dynaamiseksi ja kyseisen järjestelmän teko tyhjästä on paljon helpompaa.

Viikon aikana huomasin, miten olen viimeisen kuukauden aikana kehittynyt ohjelmoijana, niin JavaScriptin kanssa, kuin myös loogisen ajattelun kanssa. Tuntui todella helpolta tehdä dynaamista muuttuvaa sisältöä ja luodut JavaScript metodit olivat koodiltaankin sellaista jälkeä, mitä kehtaisi näyttää, jopa avoimen lähdekoodin projekteissa.

Itseoppivan ominaisuuden päälle- ja poislaittamisen toiminnon luonti ei tosin sujunut aivan niin mutkattomasti mitä luulin. Luonnin aikana tuli satunnaisia ongelmia, kuten valintaruudut eivät rekisteröityneet halutulla tavalla, sivua ladatessa valintaruutuja ei voinut klikata ja muita samantyyppisiä pieniä ongelmia. Onneksi ratkaisut olivat yleensä helppoja, eivätkä vieneet paljoa aikaa hukkaan.

Seuraavan viikon tavoitteena olisi vielä jatkaa optin lomakkeiden luonnin parissa ja tehdä puuttuvat välilehden toimiviksi. Integraatio välilehti saattaa tosin vielä jäädä tekemättä, koska yhteistyökumppani juttelee juuri sopimuksesta Segment yrityksen kanssa, joka tarjoaa kehittäjille helppoa integrointipalveluita useiden kymmenien eri alustoiden kanssa. Tämä säästäisi minulta useita kymmeniä tunteja aikaa, jos voisimme yhdistää asiakkaan integrointitarpeet toimimaan Segment palvelun kautta. Siksi en käytä vielä aikaa kyseisen välilehden toimintojen luomiseen, koska en tiedä varmuudella vielä lähestymistapaa, miten sen tulen rakentamaan.

3.9 Seurantaviikko 9

Maanantai 15.4.2019

Työpäivän alussa:

Yhdeksäs seurantaviikko jatkuu vielä optin lomakkeiden parissa. Päivän tavoitteena on jatkokehittää JavaScript luokan logiikkaa, parannella toimivuutta ja virhehallintaa.

Kun tekee tuotetta, jossa käyttäjänä ovat pääasiassa yritykset ja heidän työntekijät, kenellä ei välttämättä ole teknistä taustaa, on pidettävä huoli, että tietokantaan tallennettu tieto on juuri sitä mitä halutaan, lisäksi tuotteessa ei voi tehdä valintoja, jotka hajottaisivat toiminnot. Päivän tavoitteena olisi siis parantaa optin lomakkeiden toimivuutta ja tehdä virhehallintaa parantavia toimintoja, jotta käyttäjä ei voi esimerkiksi tallentaa lomaketta täyttämättä kaikkia tietoja. Toinen päivän tavoite on tehdä järjestelmä mikä hoitaa paljon taustalla tapahtuvia tapahtumia, jotta käyttäjän optin lomakkeen luonti on helppoa ja sulavaa.

Työpäivän lopussa:

Sain päivän aikana tehtyä paljon taustalla tapahtuvaa logiikkaa, joka pitää huolen siitä, ettei käyttäjän optin lomakkeen luonti voi kaatua niin sanotusti. Vielä päivän alussa lomakkeen toiminnoissa oli pieniä puutteita, kuten mitä tapahtuu, kun lomake on täytetty ja yhtäkkiä lomakkeen tyyppiä vaihdetaan? Vanhat asetukset jäivät päälle ja JavaScript

luokka kokeili ladata kävijälle näkyvää lomakkeen HTML koodia väärällä avaimella ja lomakkeen tekijä näki oikealla puolella sivua pelkän ”*undefined*” tekstin. Päivän aikana korjasin paljon tällaisia toimintoja.

Keskiviikko 17.4.2019

Työpäivän alussa:

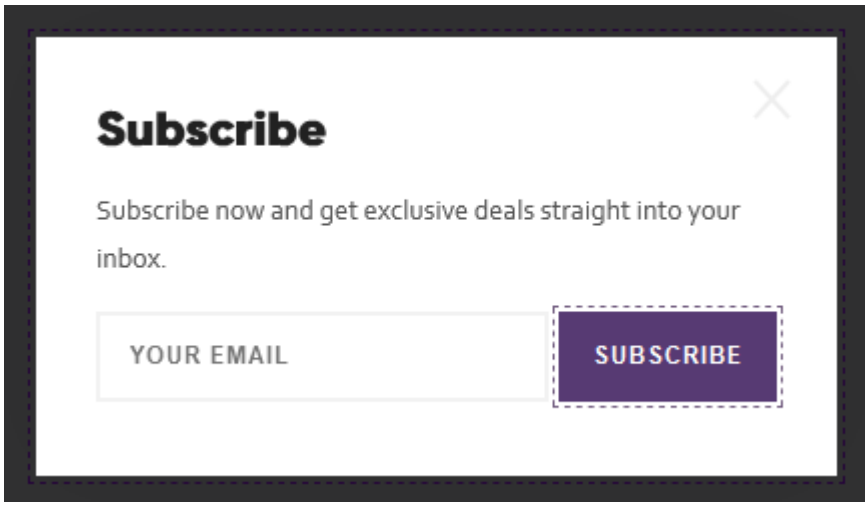
Keskiviikon tavoitteena olisi saada ominaisuus aloitettua, jossa käyttäjä voi valita optin lomakkeessa hiirellä elementin, jonka tietoja voi muokata vapaasti, kuten tekstiä, väriä, kokoa ja sijaintia. Lisäksi kaikille näille ominaisuuksille tulisi olla mahdollista käyttää itseoppivaa ominaisuutta.

Tiesin jo valmiiksi, etten voi käyttää edellisessä versiossa tullutta ratkaisua, jossa lomakkeen näkymä ladataan iframe elementin sisälle, vaan sivun oikea puoli tulisi korvata elementillä, johonka HTML koodi ladataan. Haluan ratkaisun, joka kuuntelee käyttäjän hiiren liikkeitä, esimerkiksi jos käyttäjä klikkaa lomakkeessa olevaa nappulaa, näkyisi elementti ensiksi korostettuna ja klikkaamalla vasemmalle puolelle sivupalkkia tulisi asetussivu näkyviin. Tämä ei onnistuisi iframe elementin kanssa, koska lomaketta kohdeltaisiin toisena verkkosivuna ja tiettyjä sivussa tapahtuvia muutoksia ei voi kuunnella iframen isäntäsivulta, tämä olisi muuten erittäin iso tietoturvauhka, jos esimerkiksi rikolliset käyttäisivät pankin sivuja iframe koodin sisällä ja kuuntelisivat sivustolla tapahtuvia muutoksia JavaScript eventtien avulla.

Aloitin työpäivän iframe elementin korvaamisesta.

Työpäivän lopussa:

Sain tehtyä päivän aikana optin lomakkeiden luontia eteenpäin, tällä hetkellä käyttäjä voi hiirtä liikuteltaessa valita lomakkeen elementin ja se näkyy korostetusti käyttäjälle.



Kuva 16. Optin lomakkeen elementin valitseminen. Kuvassa kursori on asetettu ”Subscribe” nappulan päälle.

Klikattaessa valittua elementtiä vasemmalle sivupalkille avautuu tyhjä näkymä, jossa käyttäjä voi asettaa elementin asetukset, kun saan tehtyä kyseisen toiminnon.

Torstai 18.4.2019

Työpäivän alussa:

Torstai aamu jatkuu edelleen optin lomakkeiden parissa. Tämän päivän tehtävänä olisi tehdä optin lomakkeen elementtien asetus paneeli. Kun optin lomakkeen elementtiä painetaan, tulisi vasemmalle puolelle sivua, sivupalkkiin näkymään allekkain lista kyseisen elementin asetuksista. Jokainen asetus tulisi tallentaa muistiin, kun asetusta vaihdetaan.

Päivän ensimmäinen työ on suunnitella HTML pohja jokaiselle asetukselle. Tekstikentälle oma, värin valinnalle oma jne. Seuraavaksi tulisi suunnitella dynaamisesti sivun latauksessa toimiva funktio, joka rekisteröi ja lisää valitut asetukset tietyille elementeille sivun HTML koodiin.

Työpäivän lopussa:

Työpäivän aikana sain tehtyä suunnitellun järjestelmän lähes valmiiksi. Tällä hetkellä asetukset latautuvat ja vaihtuvat elementtiä valittaessa, mutta en vielä ehtinyt tehdä toimintoa, joka tallentaisi tiedot heti, kun asetusta muutetaan. Samalla tulisi päivittää esikatselunäkymä, kun asetusta muutetaan. Nämä toiminnot jäävät vielä huomiselle.

Perjantai 19.4.2019

Työpäivän alussa:

Päivän tavoitteena olisi viimeistellä optin lomakkeen elementtien asetus välilehti. Jätin eilen työn siihen vaiheeseen, että tekemättä on vielä toiminto, joka kuuntelee, kun asetusta vaihdetaan. Toiminnon tulisi tallentaa asetus välimuistiin ja samalla päivittää esikatselunäkymä.

Tein eilen toiminnon, joka lataa dynaamisesti kaikki elementtiin kuuluvat asetukset ja ajattelin, että olisi järkevintä laajentaa kyseistä toimintoa, jotta saan tämän päivän työt tehtyä helposti.

Työpäivän lopussa:

Päädyin ratkaisuun, jossa tietylle elementin asetukselle rekisteröidään lennosta ”*onChange*” toiminnon kuuntelija. Eli tässä tapauksessa, jos tekstipalkin teksti muuttuu, kutsuu se automaattisesti rekisteröityä funktiota, jolle on rekisteröity samalla tiedot mistä asetuksesta kutsu tulee ja asetus päivitetään myös taustalla, eikä vain näkyvällä sivulla. Jonka jälkeen esikatselunäkymä tulee päivittymään.

Viikkoanalyysi

Viikon aikana ei ole tapahtunut kehitystä paljon varsinkaan uusissa taidoissa. Viikon ohjelmointi on ollut lähinnä sellaista, mikä on ollut tuttua viimeiset vuodet.

Viikon aikana suurimmat ongelmat koskivat dynaamisesti muuttuvaa sisältöä ja virhehallintaa. Aina kun tekee jotain suurta ja paljon muuttuvaa sisältöä, harvemmin virheiltäkään vältytään. Tällä kertaa ongelmana oli JavaScriptin kanssa muutamia ongelmia, kuten se, että tietyt sivulla tapahtuneet muutokset eivät rekisteröityneet niin kuin piti.

Muutamia ongelmia tuli myös lomakkeen elementtien kanssa. Törmäsin ongelmaan, jossa lomakkeen elementtiä klikatessa, järjestelmä ei tunnistanut oikeaa klikattua elementtiä. Eli jos haluisin klikkaa hiirellä lomakkeen nappulaa, näki tapahtumien kuuntelija, että klikkasin taustakuvaa. Ongelman aiheuttaja oli tapahtumien kuuntelijan liika pääsy eteenpäin koodissa. Heti kun kuuntelija näki, että jotain tapahtui, se rekisteröi samalla viimeisemmän elementin tapahtumat. Koska lomakkeen lähetysnappula on päällimmäisenä ja taustakuva takana, tarkoittaa se sitä, että tapahtumien kuuntelija rekisteröi viimeisimpänä taustaku-

vassa tapahtuvat muutokset. Korjasin ongelman lopettamalla tapahtumien kuuntelijan heti, kun ensimmäinen tapahtuma havaitaan. Näin se ei jatka eteenpäin ja rekisteröi sen, mitä käyttäjä haluaa tehdä.

Ainakin ongelmanratkaisutaidot ovat kehittyneet viikko viikolta, muuta kehitysideaa ei tule tältä viikolta mieleen.

3.10 Seurantaviikko 10

Maanantai 22.4.2019

Työpäivän alussa:

Viimeinen seurantaviikko alkaa vielä optin lomakkeiden parissa. Päivän tavoitteena olisi saada päivitettyä Elasticsearch indeksi, joka vastaa optin lomakkeiden asetuksista, vastaamaan kaikkia muutoksia, joita olen tehnyt viimepäivinä. Lisäksi tavoitteena olisi tehdä toiminto, jolla lomake voidaan tallentaa tietokantaan.

Työpäivän lopussa:

Päivä meni Elasticsearchin dokumentaatiota tutkiessa, halusin opetella miten indeksin voisi päivittää poistamatta sitä. Koska indeksin rakennetta ei voi Elasticsearchissa muuttaa lennosta, kuten relaatiotietokannoissa, tulee se indeksoida uuteen indeksiin, joka sisältää uudet rakennetiedot.

Tiedot tulevat hyödyksi jatkossa, kun alustaa tullaan jatkokehittämään ja haluttuja ominaisuuksia tullaan lisäämään.

Samalla sain päivitettyä indeksin ottamaan vastaan uudet optin lomakkeeseen vaadittavat tiedot ja tein painikkeen toimivaksi, joka lähettää lomakkeen tiedot tallennettavaksi.

Tiistai 23.4.2019

Työpäivän alussa:

Päivän tavoite olisi päivittää tuotteemme optin lomakkeista vastaava osio, joka lataa lomakkeen asiakkaan omalle verkkosivulle. Kyseinen toiminto on jo tehty aikaisemmin, mut-

ta se ei toimi tällä hetkellä, koska lomakkeiden tiedot ovat muuttuneet niin paljon viimeisimpien päivitysten takia.

Työpäivän lopussa:

Sain työpäivän aikana tehtyä lomakkeista vastaavat toiminnot päivitettyä. Lisäksi päivitin API osiossa itseoppivan järjestelmän, joka hoitaa viimeisempien päivitysten myötä, myös lomakkeen asetusten optimoinnin.

Päivästä tuli hiukan tynkä, kun minun piti lähteä hoitamaan yritykseen liittyviä asioita. Avasimme briteissä Metro pankkiin yrityspankkitilin ja he vaativat minulta kahta vahvistettua osoitetta ja henkilöllisyystodistusta notaarin allekirjoituksella. Kävin hoitamassa asian maistraatissa.

Keskiviikko 24.4.2019

Työpäivän alussa:

Kesiviikko alkoi vihdoin pitkään odotetulla päivällä, voisin vihdoin keskittyä uusiin sivun ominaisuuksiin. Olen joutunut korjailemaan ja jatkokehittämään pelkästään vanhoja toimintoja viimeviikot. Nyt pääsen toteuttamaan yhtä sivustomme toimintoa, konversion seurantaa.

Toiminto näyttää visuaalisesti missä vaiheessa kävijät tippuvat asetetulta verkkopolulta pois. Asiakas voi esimerkiksi asentaa seurannan verkkokaupan monivaiheiselle tilaus prosessille, jossa ensimmäisenä asiakas menee ostoskoriin, seuraavalla sivulla täyttää henkilötiedot, seuraavassa toimitustavan ja maksutiedot ym. Tavoitteena on tehdä seuranta, joka näyttää prosentuaalisesti, monta kävijää on päätynyt seuraavalle sivulle koko seurannan aikana.

Aloin päivän suunnittelemalla Elasticsearch indeksin, jonne seurantatiedot tallennetaan.

Työpäivän lopussa:

Sain päivän aikana tehtyä konversioseurannan hyvälle alulle. Suunnittelin tietokannan seurannan, sekä tein näkymän tallennetuista konversioseurannan asetuksista ja lomakkeen seurannan luomiseen.

Kun toiminto on tehty valmiiksi verkkosivujen konversion seurannalle, toiminto on helppo muuntaa lomakkeen seurantaan. Sekin on yksi sivustomme toiminnoista, joka on suunniteltu osaksi tuotettamme.

Torstai 25.4.2019

Työpäivän alussa:

Viimeisen seurantaviikon viimeinen päivä jatkuu konversioseurannan parissa. Päivän suunnitelmana olisi tehdä asiakkaan sivulle ladattavaan skriptiin toiminto, joka lataa konversioseurannan asetukset ja lähettää seurantadataa API palvelimelle.

Onneksi työ on aika paljolti kopiointia jo valmiista koodipohjasta, koska asetusten lataaja toimii aina samalla periaatteella ja se on jokaisessa ominaisuudessa sama, vain tiedot muuttuvat.

Työpäivän lopussa:

Päivän aikana tuli huomattua, että en ole tehnyt kävijänseurantaan toimintoa, joka seuraa uniikkeja kävijöitä päivän aikana. Konversioseurantaan vaaditaan kyseinen toiminto ja tein ratkaisun, jossa kävijän IP tallennetaan kampanjan ID:n ja aikaleiman kanssa.

Samalla sain aloitettua hyvälle mallille konversion seurannan. Tällä hetkellä voin jo seurata monta kävijää on käynyt tietyssä osoitteessa, mutta logiikka on tekemättä, joka seuraa tietyn kävijän polkua ja laskee yhteistuloksen konversion seurannan visuaaliseen näkymään.

Viikkoanalyysi

Viikon aikana pääsin aloittamaan uuden ominaisuuden sivustolle, konversion seurannan. Kyseessä on siis ominaisuus, johon voidaan asettaa osoitepäätteitä, joita seuranta kuuntelee ja tietokantaan rekisteröidään prosenttimäärä kävijöistä, jotka ovat edenneet seurannassa eteenpäin.

Ominaisuus on erittäin hyödyllinen kaikille verkkosivuille, jossa tehdään myyntiä tai idea on rahan tuottaminen. Jos asiakkaalla on ongelmia saada myyntejä sivun kautta, voi asiakas asentaa seurannan ja nähdä tietyn sivun missä vaiheessa asiakas tippuu myyntiprosessista pois. Ja näin asiakas saa arvokasta tietoa siitä, missä sivustolla on vikaa.

Sama ominaisuus tulee erikseen sivulla näkyviin lomakkeisiin. Ajatellaan esimerkki, jossa sivuston omistaja on huomannut, että harva asiakas täyttää päätyy myyntiprosessissa omien tietojen täyttölomakkeeseen. Asiakas voi näin asettaa seurannan jokaiselle lomakkeen elementille ja tuotteemme tekee samanlaista konversioseurantaa, kuin sivuston osoitteisiin liitetty konversioseuranta.

Viikon aikana tein ominaisuuden, joka tekee osoitetarkistuksen seurantaan liitetyistä sivuista. Tällä hetkellä kaikki toiminnot toimivat missä vain osoitteessa, vaikka asiakas olisi määrännyt sen eri tavalla. Tämä ominaisuus muutti toiminnon toimivaksi. Se vaati hiukan tutkiskelua siitä, miten saan esille osoitteen, josta API kutsu tulee. Päädyin ratkaisuun, jossa ensin teen validoinnin JavaScript puolella, näin voin estää suoraa tuhansien turhien kutsujen teon API palvelimelle. Lisäksi API palvelimella on erikseen osoitteen validointi, joka katsoo alkuperäisen kutsun osoitteen ”*referrer*” tiedon mukaan. Näin mahdollisten sivuston haavoittuvuuksien etsijöiden täytyy selvittää yksi este enemmän ja samalla varmistutaan siitä, että kutsu tuli oikeasti halutusta osoitteesta.

Suurempia takapakkeja tai outoja ongelmia en kohdannut viikon aikana.

4 Pohdinta ja päätelmät

Opinnäytetyön raportointiosuuden aikana olen työskennellyt monien eri tehtävien parissa. Pienimmillään tehtävä on voinut olla bugin korjaus, mutta huomattavasti suurin työtehtävä minulla on ollut tuotteen siirto Kubernetes enginelle Google Cloud alustalle, käyttäen automatisoitua Travis CI palvelua. Ainut suuri toteutunut kokonaisuus onkin vain tämä palvelimen siirto Kubernetes alustalle, lähes kaikki tuotteen ominaisuudet vaativat vielä jatkokehitystä, ennen kuin voimme avata ovet tuotteeseen julkisesti. Kehitys jatkuu edelleen näitten ominaisuuksien parissa opinnäytetyön aikana käytetyn seurannan jälkeenkin. Opin seurannan aikana palvelinympäristöstä erittäin paljon. Kubernetes oli täysin uusi asia minulle, ennen opinnäytetyötä. Se on isoin asia, jonka kanssa olen joutunut työskentelemään tämän vuoden aikana. Opin käyttämään Kubernetes palvelinta alkutilanteesta loppuun. Olen oppinut, miten PHP alusta voidaan sulkea virtualisoidun kontin sisään käyttäen Dockeria ja miten tätä Docker konttia voidaan ohjata Kuberneteksen avulla. Lisäksi olen oppinut miten Kubernes palvelimella toimivaa tuotetta voidaan päivittää ja tehdä muutoksia. Koen kehittymisen palvelinympäristössä työskentelyn erittäin tärkeäksi osaksi ohjelmoijana. Se tuo lisäarvoa jo työntekijänä, mikäli haen palkkatöihin jossain vaiheessa. Kubernetes on varsinkin kehitetty yritysympäristöön. Aikaisempi palvelinkokemukseni on ollut lähinnä webhostien parissa työskentely, jossa suurin työ on tehtynä jo valmiiksi maksavalle asiakkaalle ja päivitys ja ohjelmiston hallinta on tapahtunut visuaalisen käyttöliittymän kautta.

Palvelinympäristön lisäksi, olen oppinut paljon siitä, kuinka tärkeää ohjelmistotuotannossa on hyvän koodipohjan tekeminen jo heti alussa. Tällä hetkellä kaikki Monolyth tuotteessa käytetyt PHP ja JavaScript luokat muistuttavat metodeiltaan, niin nimeämiskäytännössä, kuin toiminnallisuuksiltaan toisiaan hyvin paljon. Tämä johtuu siitä, että olen käyttänyt PHP ja JavaScript koodissa tyhjiä, esitäytettyjä luokkia, jotta kehitysvauhti nopeutuisi. Lisäksi koodi on tuttua, oli sitten mikä vain luokka kyseessä. Jos koodissa on paljon eroavaisuuksia, on helpompi unohtaa jatkossa, miten kyseinen ominaisuus on ohjelmoitu. Mutta jos on päivittäin samantyyppisen ohjelmistokoodin parissa, on vanhan ominaisuuden pariin helpompi palata. Olen kehittynyt myös ohjelmistokoodin ylläpidon kanssa, varsinkin sen takia, että koodipohja on ollut niin samantyyppistä jokaisessa eri luokassa. Nykyisin on erittäin helppoa tehdä uusi ominaisuus ja alkaa kehittämään sitä lähes suoraan.

Opinnäytetyön kannalta on huonoa se, että en koe kehittyneeni ohjelmistokehittäjä niin paljoa, kuin olisin ehkä halunnut. Tuntuu ettei minulla ole paljoa kirjotettavaa omasta kehityksestä, koska suurin kehitys on ollut palvelimien kanssa työskennellessä. Mutta tämäkään ei ole mielestäni huono asia, koska olen toiminut PHP ja JavaScript ympäristössä jo

vuosien ajan. Harvemmin kohtaan uusia asioita, joita joutuisin opettelemaan. Uudet asiat, joiden kanssa työskentelen, on yleensä apukirjastoja, joita käytän ohjelmiston toiminnan tukemiseen. Olen kyllä tullut tämän koko projektin aikana enemmän itsevarmaksi omista taidoistani, ei vain pelkästään tämän opinnäytetyön aikana. Minulla oli suuria paineita alussa, miten alkaa suunnittelemaan tämän kokoisen tuotteen rakennusta alusta loppuun asti. En ole katunut päivääkään, että lähdin tähän projektiin mukaan. Suuret itsenäiset projektit kuten tämä, avaa paljon uusia näkemyksiä ohjelmistokehityksessä. On kiva toteuttaa suurta tuotetta ja nähdä miten se on rakentunut ja kehittynyt alusta lähtien. Se, että lähdin tähän projektiin mukaan, on avannut itselle sitä mitä kaikkea oikeasti pystyn tekemään ohjelmistokehittäjänä.

Yksi heikkouksista minkä huomasin työskentelyprosessissa, on jatkuvat suunnitelman muutokset. On turhauduttavaa käyttää monta päivää ominaisuuden parissa vain, että koodi voidaan pahimmassa tapauksessa pyyhkiä pois ja tehdä uudestaan eri tyylillä. Vain sen takia, että ominaisuuden jotkin osat jäivät suunnittelematta kunnolla alkutilanteessa.

Päiväkirjamallinen opinnäytetyö on opettanut minulle seurannan tärkeyttä ja aion jatkossakin keskittyä oman työni seurantaan, jotta voin parantaa tehokkuutta ja omaa kehitystäni. Seuranta ei tule olemaan päiväkirjamallista, mutta aion tutkia käytettyyn työhön käytettyä aikaa ja huomioimaan siihen liittyviä seikkoja, mitä voisin tehdä paremmin. Päiväkirjamuotoista opinnäytetyötä kirjoittaessa, sain selkeämmän kuvan työtehtävistäni ja mitä kaikkea päivä sisältää kehittäjänä. On helppo lukea seurantaviikon tekstiä ja palata keskeneräisen ominaisuuden jatkokehittämiseen, kun saa heti selville mihin tilanteeseen kehitys jäi.

Opinnäytetyön tavoitteena oli alun perin jatkokehittää Monolyth tuotetta sen verran mitä raportointiajanjakson aikana ehti tehdä ja samalla yksi päätavoitteesta oli siirtää tuote Google Cloud palvelimelle. Onnistuin mielestäni tavoitteessa erittäin hyvin.

Tulevaisuuden haasteina on saada rakennettua Monolyth palvelu siihen pisteeseen, että voimme virallisesti avata palvelun asiakkaille. Monolythia varten on jo tehty pieniä jatkosuunnitelmia ja kehitetty ideapohjalla olevia ominaisuuksia, joita haluisimme tulevaisuudessa palvelun sisälle. Tällä hetkellä pääpointti on keskittyä keskeneräisiin ominaisuuksiin ja saada ne valmiiksi, jotta voimme aloittaa alustan beta testauksen oikeilla asiakkailla ja seurantadatalla. Vasta, kun beta on käyty ja olemme saaneet maksavia asiakkaita käyttämään tuotettamme, voimme jatkokehittää tuotetta, lisätä ominaisuuksia ja kuunnella asiakkailta saatua palautetta tuotteen parantamiseksi.

Lähteet

Evanyou.me 2014. First Week of Launching Vue.js. Luettavissa: https://blog.evanyou.me/2014/02/11/first-week-of-launching-an-oss-project/. Luettu: 8.3.2019

Docker.io. Docker Documentation. Luettavissa: https://docs.docker.com/. Luettu: 11.3.2019

Kubernetes.io. What is Kubernetes. Luettavissa: https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/. Luettu: 11.3.2019

Cert Manager dokumentaatio. Luettavissa: https://docs.cert-manager.io/en/latest/. Luettu: 22.3.2019

Mailu.io. Kubernetes Setup. Luettavissa: https://mailu.io/1.6/kubernetes/mailu/index.html. Luettu: 22.3.2019

Linuxjournal 2000. Using Postfix for Secure SMTP Gateways. Luettavissa: https://www.linuxjournal.com/article/4241. Luettu: 26.3.2019

iRedMail. iRedMail Documentation. Luettavissa: https://docs.iredmail.org/. Luettu: 26.3.2019

Google Cloud. Connecting from Google Kubernetes Engine. Luettavissa: https://cloud.google.com/sql/docs/mysql/connect-kubernetes-engine. Luettu: 3.4.2019

Github. Cloud SQL Proxy. Luettavissa: https://github.com/GoogleCloudPlatform/cloudsql-proxy. Luettu: 3.4.2019

Elasticsearch. Reindex API dokumentaatio. Luettavissa: https://www.elastic.co/guide/en/elasticsearch/reference/6.4/docs-reindex.html. Luettu: 22.4.2019

Mozilla.org. Referer. Luettavissa: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referer. Luettu: 25.4.2019

Sirish Raghuram, InfoWorld, 4 reasons you should use Kubernetes. Luettavissa: https://www.infoworld.com/article/3173266/4-reasons-you-should-use-kubernetes.html. Luettu: 21.5.2019