Isopahkala Jere

Verkkokaupan tuotehakujärjestelmän kehityksen kannattavuus Elasticsearch -alustalla

Tradenomi

Tietojenkäsittely

Kevät 2019

KAMK • University of Applied Sciences

Tiivistelmä

Tekijä: Isopahkala Jere

Työn nimi: Verkkokaupan tuotehakujärjestelmän kehityksen kannattavuus Elasticsearch -alustalla

Tutkintonimike: Tradenomi (AMK), Tietojenkäsittely

Asiasanat: elasticsearch, klevu, haku, asiaankuuluva haku, oleellinen haku, verkkokauppa,

Opinnäytetyötä voi käyttää apuna hakujärjestelmän valinnassa sekä Elasticsearch järjestelmän käyttöönotossa.

Tämä opinnäytetyö on toteutettu Pulse247 Oy:lle. MyCashflow on Pulse247 Oy:n kehittämä verkkokauppa-alusta, jolla on noin 1500 aktiivista verkkokauppaa. Pulse247 ylläpitää kaikkia MyCashflow-alustalla olevia verkkokauppoja. Opinnäytetyön tarkoituksena oli tutkia MyCashflow-alustan hakutoimintoa, joka toteutushetkellä oli ulkoistettu. Opinnäytetyön tuloksesta pystyy päättelemään, oliko kannattavaa toteuttaa ja ylläpitää hakutoiminnallisuus sisäisesti sekä millaisia haasteita sen kehityksessä tulisi vastaan. Ensisijaisesti opinnäytetyössä verrataan nykyistä toteutusta Klevulla Elasticsearchin hakujärjestelmään. Nykyisen toteutuksen ja Elasticsearchin välisiä eroavaisuuksia selvitettiin ja tutkittiin. Lisäksi opinnäytetyön ohella toteutettiin soveltuvuusselvitys eli yksinkertainen järjestelmä Elasticsearchilla, jotta voitiin todeta tiettyjen ominaisuuksien toiminnallisuus.

Opinnäytetyön tuloksista voidaan arvioida tapauskohtaisesti, onko kannattavaa ryhtyä kehittämään hakujärjestelmää Elasticsearchin alustaa käyttäen. Soveltuvuusselvityksestä on selvää erityisesti, mitkä toimeksiantajan näkökulmasta tärkeät ominaisuudet ovat yksinkertaisesti toteutettavissa Elasticsearchin alustalla sekä haastavampien toteutettavien ominaisuuksien haasteet ja vaativuuden. Kappaleessa 6.7 on eriteltynä testausominaisuudet ja niihin liittyvät onnistumiset sekä vaikeudet.

Tulosten perusteella ja toimeksiantajan tarpeet huomioiden, toimeksiantajan tapauksessa tuotehakujärjestelmän toteutus on kannattavaa käyttäen Elasticsearch alustaa.

Abstract

Author: Isopahkala Jere

Title of the Publication: E-commerce product search system development on the Elasticsearch platform

Degree Title: Bachelor of Business, Information technology

Keywords: elasticsearch, klevu, search, relevant search, online shop,

The objective of this Bachelor's thesis was to help the commissioner decide whether to use Elasticsearch as the product search application and how to deploy Elasticsearch.

This research has been carried out for Pulse247 Oy. MyCashflow is a web based online web store platform developed by Pulse247 Oy, which has about 1500 active online stores. Pulse administrates all of the online stores on MyCashflow platform. The purpose of the study was to examine the current search solution used on MyCashflow, which has been outsourced at the time of the research. The research was meant to help deduce whether it is worthwhile to implement and maintain a search solution functionality internally. The second purpose was to research possible challenges the development will face. Primarily, the research compares current implementation with Elasticsearch search solution. In this research, the differences between the current implementation and Elasticsearch were investigated and studied. In addition, the work includes a proof of concept test environment meant to determine the functionality of certain properties.

The results of the study can be evaluated on a case-by-case basis; whether it is profitable to start developing the search system using Elasticsearch platform. The proof of concept testable features where selected from the requirements of the client. The proof of concept test cases shows, in particular, which features are simple to implement for the Elasticsearch platform and which features contain challenges. Proof of concept also helps to determine how difficult different features are to implement and what is required to implement them. Section 6.7 specifies the features tested and the associated successes and difficulties.

Based on the results and considering the viewpoint of the client, the implementation of product search using the Elasticsearch platform is worthwhile.

Sisällys

Symboliluettelo

API	Application programming interface eli ohjelmistorajapinta
Bool -haku	Bool on hakutyyppi, jolla voidaan yhdistää hakutermejä käyttäen yhdistinmerkkejä AND, NOT ja OR.
Boost	Säädettävä arvo Elasticsearchissa, joka nostaa hakutuloksen arvoa.
Cluster	Rykelmä Elasticsearchin servereitä eli nodeja.
Indeksi	Tarkoittaa listaa hakutermeistä, jotka on kohdistettuna niitä koskeviin dokumentteihin.
Node	Yksittäinen Elasticsearchin serveri
PGP	Pretty Good Privacy
Replica	Kopio sirpaleesta
REST/ RESTful	Representational State Transfer
Shard	Sirpale on indeksin osa
Token	Tarkoittaa yksittäistä hakutermiä

1 Johdanto

Tämä tutkimus pyrkii selvittämään yrityksen näkökannalta, onko kannattavaa kehittää hakujärjestelmää verkkokauppojen tuotteiden hakua varten, käyttäen Elasticsearchin alustaa. Tutkimuksessa Elasticsearchin alusta on verrattuna yrityksen nykyiseen toteutukseen. Nykyinen toteutus on ostettu ulkoiselta palveluntarjoajalta Klevulta. Tutkimuksen kohteena olevien verkkokauppojen tuotteet sisältävät yllättävän laajasti vaatimuksia hakujärjestelmältä. Tästä johtuen tutkimusta voi soveltaa hyvin muihinkin hakujärjestelmien suunnitelmiin, joissa hakukohteiden asettamat vaatimukset sisältävät esimerkiksi täyden tekstin hakuja sekä vaativat erityisesti olennaiset tulokset.

Tutkimuksen aihe on tärkeä toimeksiantajalle sekä muille ihmisille, jotka käyttävät hakukoneita. Hakujärjestelmiä on nykyään kaikkialla ja me käytämme niitä niin paljon, ettemme edes huomaa niiden olemassaoloa. Silti hakujärjestelmät eivät aina palauta olennaisia hakutuloksia. Elasticsearch soveltuu moneen erilaiseen tarkoitukseen, skaalautuu yksinkertaisesti suuriin tarpeisiin ja se on käytössä monissa arvaamattomissa paikoissa. Tutkimuksen toimeksiantaja on kajaanilainen verkkokauppaohjelmistoa kehittävä yritys. Yritys ylläpitää kaikkia verkkokauppoja, jotka toimivat toimeksiantajan verkkokauppaohjelmistolla.

Työssä keskitytään toimeksiantajan vaatimuksiin verkkokauppojen ylläpitäjänä. Vaatimuksiin kuuluu siis ottaa huomioon useiden täysin eri verkkokauppojen tuotehakujen eriävät tarpeet. Verkkokauppojen suuren määrän sekä haastavien eroavaisuuksien lisäksi työssä huomioidaan erityisesti verkkokauppojen tuotteiden olennaisen haun vaatimukset.

Tavoitteena on tutkia nykyisen tuotehakujärjestelmän ominaisuudet ja selvittää, mitkä niistä ovat sellaisia ominaisuuksia, joita vaaditaan korvaavalta järjestelmältä. Ovatko edellisen järjestelmän tärkeimmät ominaisuudet toteutettavissa vaihtoehtoisella järjestelmällä ja millaisiin esteisiin vaihtoehtoisen järjestelmän kehityksessä tullaan törmäämään? Selvitetään, onko kannattavaa kehittää vaatimukset täyttävä tuotehakujärjestelmä itse, ja mitä sen kehitys tulee sekä vaatimaan että antamaan toimeksiantajan näkökulmasta. Tavoitteiden saavuttamiseen vaaditaan kattava pohja siitä, mitä vaatimuksia järjestelmälle esitetään sekä tuetaanko näitä vaatimuksia vaihtoehtoisessa järjestelmässä vai vaativatko ne kehitystä. Tärkein osa tavoitteesta on toimeksiantajan näkökulman ymmärtäminen ja siitä näkökulmasta katsoen tutkimuksen suorittaminen. Tähän opinnäytetyöhön valittiin testiesimerkeiksi tyypillisiä verkkokaupan haun

käyttötapauksia, joissa perinteiset tuotehakukoneet palauttavat huonoja tuloksia. Kaikki testitapaukset ovat tuettu toimeksiantajan nykyisessä hakukoneessa.

2 Tuotehaku

Tässä opinnäytetyössä keskitytään useimmista verkkokaupoista löytyvän hakukentän taustalla olevaan tuotehakujärjestelmään. Hakukenttään voi syöttää vapaata tekstiä, johon halutaan palauttaa vain olennaisia tuotteita mahdollisimman olennaisessa järjestyksessä.

Koska luonnollisen tekstihaun käyttötapauksia on paljon, dataa voi olla valtava määrä ja kysyntäpiikit kovia sesonkiaikoina, niin hakutuloksia ei voida etsiä järjestelmästä reaaliaikaisesti. Hakujärjestelmään indeksoidaan tuotetietoja niiden muuttuessa erityyppisiä hakuja varten ja varsinaiset käyttäjien verkkokaupassa tekemät tuotehaut etsivät hakutulokset nopeasti indeksistä.

Teknisellä tasolla tuotehakujärjestelmä toimii siten, että verkkokauppa lähettää kaikista tuotteista tiedon tuotehakujärjestelmään, joka rakentaa niistä indeksin. Indeksoinnin jälkeen tuotteet ovat haettavissa, joten verkkokaupan käyttäjät voivat syöttää hakuehtonsa hakukenttään. Verkkokauppajärjestelmä pyytää hakuehtoihin vastaavia tuotteita tuotehakujärjestelmästä, joka palauttaa tuotenumerot tärkeysjärjestyksessä ja verkkokauppa tulostaa sivulle tuotenumeroita vastaavat tuotteet.

Tuotehakutilanteissa voidaan myös antaa rajauksia, esimerkiksi vain tuotteet, joiden hinta on 20-50 € tai kauppakohtaisten tietotyyppien rajaukset, kuten autonrenkaan vannekoko. Luonnollisesti kaikilla kaupoilla ei ole tarvetta vannekoon mukaiseen tuotteiden rajaukseen mutta muunlaiselle tuotteiden rajaukselle on tarvetta. Tuotehakujärjestelmän on tuettava edellä mainittua tapausta, jotta se voidaan toteuttaa verkkokauppaan. Esimerkiksi autonvanteita myyvä verkkokauppa voi toivoa, että tuotteet voidaan rajata vannekoon mukaan, mutta hakujärjestelmää suunnitellessa emme voi varautua jokaiseen eri tilanteeseen. Hakujärjestelmän pitää olla joustavasti muokattavissa indeksi- eli verkkokauppakohtaisesti, jotta kaupat voivat asettaa omat rajausvaihtoehdot.

Verkkokauppiaat haluavat usein hallita, millaisia hakutuloksia järjestelmästä tulee. Yleinen tapaus on mainostettavat tuotteet. Kun verkkokauppias mainostaa tai suosittelee jotain tuotetta, hän haluaa, että kyseinen tuote tulee ensin asiakkaan näkyville. Verkkokauppiaat voivat asettaa tuotteen näkyville etusivulle, mutta myös hakutuloksiin, jos asiakas hakee esimerkiksi t-paitaa. Tulokseksi tulee kaksi tuotetta, joiden molempien nimessä on ”t-paita”. Koska hakusana esiintyy molempien tuotteiden nimessä, ei hakujärjestelmä osaa sanoa, kumpi tuote pitäisi näyttää ensin.

Tässä tilanteessa verkkokauppias voi asettaa toisen tuotteista mainostettavaksi. Kun tämä tieto indeksoidaan hakujärjestelmään, osaa hakujärjestelmä ensi kerralla asettaa sen arvon korkeammaksi. Aina kun mainostettava tuote tulee hakutuloksiin, niin se asetetaan hieman korkeammalle kuin muussa tapauksessa olisi asetettu.

Hakujärjestelmän on hyvä myös tukea erikielisiä tilanteita, esimerkiksi kirjoitusvirheet ja synonyymit. Kielellisissä tilanteissa tulee ottaa huomioon erikieliset verkkokaupat sekä erikieliset verkkokaupan asiakkaat. Tällöinkin hakujärjestelmän tulisi tukea käytössä olevaa kieltä ja sen mahdollisia kirjoitusvirheitä tai synonyymejä. Muita kielikohtaisia tapauksia ovat hakusanojen normalisointi, haun rikastuttaminen ja yhdyssanojen irrottaminen.

Hakujärjestelmän kehityksen kannalta tärkein osa on analytiikka. Hakuanalytiikan tarkoitus on, että sieltä voi seurata, mitä hakusanoja ihmiset käyttävät, kuinka usein ja mitä tuloksia hakujärjestelmä palauttaa kyseisille hakusanoille. Hakuanalytiikkaa seuraamalla voidaan kehittää hakujärjestelmää siten, että se palauttaa tarkempia tuloksia tai tuloksia hakusanoille, jotka eivät ennen palauttaneet tuloksia ollenkaan. [1.] [2.]

Hakujärjestelmän mittapuita ovat siis: Kuinka luotettavasti hakutuloksia tulee? Ovatko hakutulokset toivottuja? Kuinka nopeasti uudet tuotteet ovat haettavissa hakujärjestelmästä? Kuinka helppoa hakujärjestelmään on lisätä rajaus ja erikoistilanteita? Kuinka järjestelmästä saadaan analytiikkaa ja onko data sovellettavissa järjestelmän kehittämiseen?

3 Nykyinen toteutus Klevu

Nykyinen hakujärjestelmä on toteutettu Klevu-integraatiolla. Klevu on maksullinen suomalainen palvelu, joka tarjoaa hakukoneintegraatiota erilaisiin verkkokauppajärjestelmiin. Toimeksiantajan verkkokauppa alustaan Klevun integraatio toteutettiin vuonna 2015. Klevu on ohjelmoitu Apache Lucene-kirjaston päälle. [3.]

Klevun tarjoamista ominaisuuksista tärkeimmät ovat tuotehaku, tuotteiden rajaus, tuotteiden arvon boostaus, hakuanalytiikka ja kielien tuki. Kielien tuesta etenkin hakusanojen normalisointi, haun rikastuttaminen, yhdyssanojen irrottaminen ja automaattinen täydennys ovat tärkeitä ominaisuuksia. [4.]

Mielipiteet Klevusta ovat toimeksiantajan yrityksessä yksimieliset. Klevun järjestelmä toimii luotettavasti harvinaisia katkoksia lukuun ottamatta, mutta ominaisuuksia puuttuu. Suurin puute on hakuanalytiikan toiminta. Hakuanalytiikka on mainostettu Klevun ominaisuus, mutta käytännössä se on osoittautunut hyödyttömäksi. Klevun pidettävyyttä laskee myös indeksin toiminta, joka ei päivity kuin kerran päivässä. Tämä tarkoittaa sitä, että kun uusi tuote lisätään, se saapuu hakutuloksiin vasta seuraavana päivänä. Puutteelliset ominaisuudet ja indeksin hidas päivittyminen tulivat esiin ensimmäisenä mielipiteistä. Klevusta ei pidetä myöskään siksi, koska sitä ei voi itse päivittää. Toimeksiantajan yrityksessä on paljon osaajia, jotka haluavat kehittää hakutoimintoja mutta Klevu on ulkopuolinen järjestelmä, joten sinne ei päästä tutkimaan vian lähdettä, saati kehittämään uusia ominaisuuksia. Kitkaa aiheuttaa tilanteessa myös se, että kun ongelmia ilmenee, kysymys täytyy aina ohjata Klevulle ja odottaa sieltä vastausta. [1.]

Toimeksiantajalla selkeinä kehitystoiveina ovat pääsy järjestelmään, hakuanalytiikka sekä hakukoneen räätälöinti. Jos toimeksiantajalla olisi pääsy järjestelmään, ongelmatilanteita voitaisiin ratkoa itse ja nopeammin.

Hakuanalytiikka on, kuten jo mainittua, todella tärkeä osa hakutulosten kehityksessä ja sillä voi olla vaikutusta koko verkkokaupan kehittämiseen. Klevun hakuanalytiikka kaatuu useasti aikakatkaisuun ja data on myös väärää tai puutteellista, siis hyödytöntä. Hakuanalytiikan avulla voitaisiin räätälöidä hakujärjestelmää paremmin, mikäli Klevun järjestelmässä hakujärjestelmän räätälöinti olisi helposti toteutettavissa.

Klevun hyviä ominaisuuksia ovat luotettavuus ja hakutulokset. Klevun haku toimii ja asiakkaat löytävät sen avulla tuotteita. Hakutuloksia parantavat ominaisuudetkin toimivat oletettavasti. Palvelun luonteesta johtuen ongelmatilanteissa käyttökatkos voi olla pitkä ja siksi haitallinen.

4 Elasticsearch

Elasticsearch on tietokanta, joka on rakennettu käsittelemään suuria määriä dataa korkealla saatavuudella siten, että se jakautuu usealle laitteelle vikasietoisuutta ja skaalautuvuutta varten. Samalla se sisältää yksinkertaisen, mutta tehokkaan ohjelmointirajapinnan (API), joka sallii ohjelmien käyttää Elasticsearchia lähes millä tahansa ohjelmointikielellä tai ohjelmistokehyksellä. [5.]

Shay Banon on Elasticsearchin alkuperäinen ohjelmoija, ja ensimmäinen Elasticsearch versio julkaistiin vuonna 2010. Elasticsearch-yritys on nykyisin nimeltään Elastic N.V. Elastic tuottaa avoimen lähdekoodin tuotteita, kuten Elasticsearch, Kibana ja Logstash. Edellä mainittujen lisäksi Elastic myy tuotteita ja palveluita avoimen lähdekoodin tuotteiden ympärille. Elasticsearch pohjautuu Apache Lucenen ohjelmistokirjastoon. [5.]

Elasticsearch on erittäin skaalautuva avoimen lähdekoodin täyden tekstin haku- ja analytiikkamoottori. Sen avulla voi tallentaa, etsiä ja analysoida suuria tietomääriä nopeasti ja lähes reaaliajassa. Sitä yleensä käytetään perustana sovelluksille, joilla on monimutkaisia hakutoimintoja ja vaatimuksia. [5.]

Elasticsearch kykenee muun muassa reaaliaikaiseen analytiikkaan sekä hajautettuun reaaliajan dokumenttien varastointiin, jossa jokainen kenttä on indeksoitu sekä haettavissa [6].

Elasticsearch on kirjoitettu Javalla, ja kaikki toiminnallisuus on toteutettavissa yhdellä palvelimella, jonka käyttö onnistuu yksinkertaisella REST-ohjelmointirajapinnalla (REST API). Näin Elasticsearch on yhteensopiva kaikkien ohjelmointikielien kanssa, jotka tukevat RESTful-ohjelmointirajapintaa. [5.]

Elastic tarjoaa myös muita hyviä palveluita, jotka tukevat Elasticsearchin käyttöä. Esimerkkinä Logstash, joka prosessoi lokidataa ja lähettää ne Elasticsearchiin, sekä Kibana, jonka avulla voi hakea ja visualisoida Elasticsearchin dataa kätevästi web-käyttöliittymässä.

Elasticsearchin luotettavuutta on kehuttu paljon, ja Elasticsearchia käytetään hyvinkin kriittisissä järjestelmissä ilman ongelmia. Esimerkiksi NASAn Curiosity Rover -projekti käyttää Elasticsearchia analysoimaan kaikki data, mitä Curiosity Rover lähettää ja tekee niiden avulla päätöksiä. Elasticsearch auttaa päättämään, mihin on parasta käyttää Curiosityn vähäinen energia sekä minne on paras mennä, jotta tehtävä etenee. [7.] [8.]

4.1 Elasticsearch toimeksiantajalla

Toimeksiantaja käyttää Elasticsearchia verkkokauppojen lokitietojen tallentamiseen sekä bisnes-analytiikkaan. Myös Elastic yrityksen Logstash-palvelu on käytössä lokitietojen lukemiseen ja hyödyntämiseen. Elasticsearch on ollut toimeksiantajalla kyseisessä käytössä vuodesta 2014 asti. [1.] Keväällä 2015 Logstash kaatui, mistä johtuen Elasticsearchin käyttö estyi. Siitä huolimatta Elasticsearch ei ole koskaan kaatunut eikä sen käyttö toisen ohjelmointirajapinnan kautta ole koskaan estynyt. [1.] Toimeksiantajan yrityksessä Elasticsearchia käytetään Logstashin kautta mutta Elasticsearch on saatavilla myös ohjelmointirajapinnan kautta. Tästä johtuen Elasticsearchia pystyi käyttämään Logstashin kaatumisesta huolimatta.

Elasticsearchin käyttökokemus ja luotettavuus ovat olleet erityisen hyvä toimeksiantajalla. Elasticsearchista on pidetty, ja se soveltuu moneen erilaiseen toimintaan. Toimeksiantajalla on useampia kehityskohteita, joihin Elasticsearch soveltuisi hyvin.

Elasticsearch on ilmainen ohjelmisto, joten toimeksiantaja pystyy käyttämään enemmän resursseja sen kehittämiseen sekä ylläpitoon. Elasticsearch on myös hyvin skaalautuva suurellekin datamäärälle, mikä on toimeksiantajalle tärkeää. Elasticin skaalautuvuudesta todisteena on monet suuret yritykset, jotka käyttävät sitä juuri ison datan käsittelyyn. Esimerkkinä Netflix, Goldman Sachs, StackExchange ja LinkedIn. [8.] [9.]

4.2 Muita järjestelmiä

Apache Solr on avoimen lähdekoodin hakualusta, joka on kirjoitettu Java-ohjelmointikielellä Apache Lucenen päälle, kuten Elasticsearch. Apache Solr tarjoaa hyvin samanlaiset ominaisuudet kuin Elasticsearch. Perustuvathan ne sentään samaan alustaan. Apache Solr on laajasti käytössä myös isoilla yrityksillä. Apache Solr tukee REST-, HTTP/XML- ja JSON-ohjelmointirajapintoja, joiden avulla sen käyttö onnistuu useilla eri ohjelmointikielillä. Apache Solr asennetaan omaksi täydentekstin hakupalveluksi, kuten Elasticsearch. Apache Solrista eroten Elasticsearch tarjoaa skaalautuvuuden, joka on rakennettu Elasticsearch-pakettiin Apache Lucenen päälle. Tämä tarkoittaa sitä, että skaalautuessa usealla palvelimelle Elasticsearch osaa luoda ja synkronoida useita Apache Lucene -indeksejä, mutta kaikki on saatavilla saman REST-rajapinnan kautta. Myös Klevun hakukone perustuu Apache Lucenen teknologiaan.

5 Vertailu

Klevua ja Elasticsearchia voidaan verrata seuraavilla ominaisuuksilla: hinta, luotettavuus, jatkokehitysmahdollisuus, kehityksen helppous, asiakaspalveltavuus, skaalautuvuus ja indeksointinopeus.

Kun verrataan Elasticsearchin ja Klevun mainostettuja ominaisuuksia, niin on selvää, että Elasticsearchissa on toteutettavissa paljon enemmän ominaisuuksia. Tämä johtuu tietysti siitä, että Elasticsearch ja Klevu perustuvat samaan alustaan, mutta Klevu ei ole implementoinut kuin heille tärkeät ominaisuudet. Toimeksiantaja ei voi lisätä Klevun järjestelmään ominaisuuksia vaan voi vain käyttää Klevun tarjoamia ominaisuuksia. Elasticsearchin tapauksessa ominaisuuksien lisäämisessä on kysymys siitä, kuinka paljon toimeksiantaja siihen haluaa laittaa resursseja. Toimeksiantajalla on halukkuutta sijoittaa resursseja hakujen kehitykseen mutta nykyisen järjestelmän kanssa itsenäinen kehitys ei ole mahdollista. [10.] [11.]

Toimeksiantajalla on kokemusta Elasticsearchin luotettavuudesta, sillä kuten aikaisemmin mainittiin, Elasticsearchia käytetään jo lokien tallentamiseen sekä niiden tarkasteluun. Elasticsearchin luotettavuudesta on yleisesti puhuttu paljon hyvää jopa useilta isoilta tekijöiltä. Klevun luotettavuudesta onkin jo mainittu paljon. Klevun luotettavuudesta ei ole ollut suurempia valituksia. Useimmiten järjestelmä toimii, mutta järjestelmän kaatumisia ja muita luotettavuusongelmia esiintyy toimeksiantajalla aina silloin tällöin. [1.] [8.]

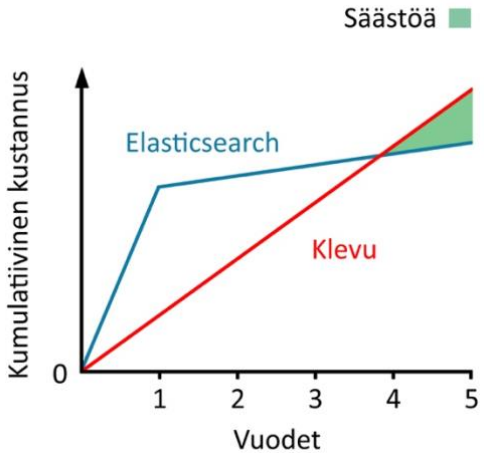
Skaalautuvuus ja luotettavuus ovat samaa järjestelmää Elasticsearchilla. Soveltuvuusselvityksen kohdassa ”Elasticsearch asennus” kerrotaan tarkemmin näiden järjestelmien toiminnasta mutta selkeytettynä Elasticsearch kykenee jakautumaan usealle palvelimelle ja tekemään itsestään kopioita, jotta kaikki data on usealla eri palvelimella ja mahdollisesti myös useaan kertaan. Tämä mahdollistaa sen, että vaikka yksi palvelin menisi rikki, niin mitään dataa tai toiminnallisuutta ei menetetä, kunhan muut palvelimet vielä toimivat. Tästä samalla etuna tulee skaalautuvuus. Elasticsearch on ainoa hakualusta, jossa skaalautuvuus tulee paketoituna mukaan. Elasticsearch luo useita indeksejä, synkronoi ne automaattisesti ja ulospäin Elasticsearchia käytetään aina saman REST-rajapinnan kautta. Elasticsearchin ollessa usealla palvelimella onnistuu useiden kyselyiden suorittaminen yhtä aikaa. Klevun järjestelmän toiminnasta ei ole tarkempia tietoja. Käyttökokemuksesta tiedetään kuitenkin, ettei Klevu ole osoittanut ongelmia skaalautuvuudessa.

Toimeksiantajan yrityksessä Klevun yksi heikoista kohdista on tuen saatavuus. Toimeksiantajan asiakkaat ottavat tuotehakuun liittyvissä vikatilanteissa yhteyttä heidän asiakaspalveluunsa, mutta heitä ei aina voida heti auttaa. Yleensä sellaisessa tilanteessa voidaan vain todeta ongelma ja kertoa, että sitä selvitellään. Samaan aikaan täytyy ottaa yhteyttä Klevuun, jotta ongelmaa voidaan alkaa selvittämään. Elasticsearch sisäisesti toteutettuna olisi nopeampi, sillä asiakas voitaisiin heti yhdistää ylläpitäjään ja ongelmaa voitaisiin alkaa selvittämään. Tiettyjä tilanteita voitaisiin myös antaa asiakaspalveluhenkilöstön käsiin, jotta pienet toimenpiteet eivät vaatisi useampia käsiä. Asiakastyytyväisyys voisi lisääntyä. Lisäksi indeksointi saataisiin toteutumaan lähes reaaliajassa. Tämä tarkoittaa sitä, että asiakkaan lisätessä tuotteen kauppaansa. Se on lähes välittömästi haettavissa kaupassa. Samoin muutokset, esimerkiksi tuotteiden mainostukseen, ottaisi vaikutusta yhtä nopeasti. Asiakkaat voisivat itsenäisesti tarkkailla kauppansa hakutoimintaa samalla, kun lisäävät, poistavat tai muokkaavat mainostettavia tuotteita. Klevulla indeksointi toteutetaan tällä hetkellä kerran vuorokaudessa.

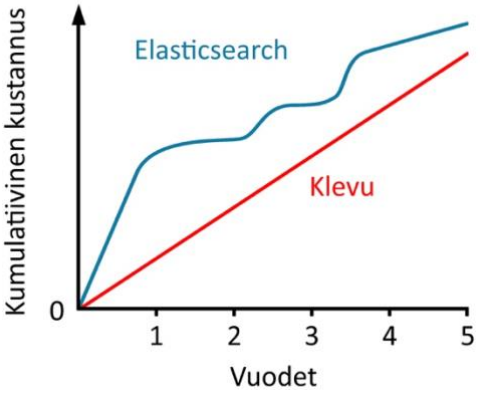
Iso huomioon otettava tekijä, kun verrataan kyseessä olevia järjestelmiä, on hakutulosten oleellisuus. Oleellisuus on merkittävä verkkokauppojen tuotehaussa, sillä jos asiakas ei saa hänen hakutermeilleen olennaista tuotetta heti ensimmäiselle tuotelistaukselle, voi se merkitä asiakkaan menettämistä. Hakutulosten oleellisuutta täytyy säätää hakujärjestelmän asetuksilla. Klevulla säätöä tekee Klevun tuki, joka huolehtii hakutulosten oleellisuudesta. Toimeksiantajan itse ylläpitämässä järjestelmässä tämä pitäisi tietenkin tehdä itse. Oleellisuuden säätämiseen menee paljon resursseja ja tietyt muutokset vaativat uudelleen indeksointia, joka täytyy ottaa huomioon, kun mietitään Elasticsearchin käyttöönottoa.

Viimeisenä puhutaan hinnasta, Klevu laskuttaa kiinteän kuukausihinnan, joka kattaa kaiken. Siirryttäessä Elasticsearchiin on huomioitava kaksi muuttujaa: kehityskustannukset sekä ylläpitokustannukset. Kehityskustannukset tulevat ensimmäisenä, ja niiden suuruuteen vaikuttaa kehityksen kesto. Toimeksiantajalla työntekijät ovat kuukausipalkalla, eli jos kehitys venyy, niin kustannukset nousevat. Ylläpitokustannukset ovat hankala arvioida toimeksiantajalle, sillä Elasticsearchia käytetään jo muuhunkin kuin tuotehakuun. Ylläpitokustannukset kuitenkin koostuvat servereiden vuokrasta sekä työntekijöitä vaativien ongelmien ratkomiseen. Huomioimalla pelkästään tuotehaun tuomat lisäkustannukset voidaan olettaa, että Elasticsearchilla toteutettu hakujärjestelmä tulisi halvemmaksi kuin Klevu.

Alla kuvat siitä, miten Elasticsearchin käyttöönotto ja Klevun käytössä pito todennäköisesti eroaa kustannuksissa. Toisessa kuvassa otetaan huomioon, että Elasticsearch-tuotehakua kehitetään, vaikka sen tarjoama toiminnallisuus jo korvaa Klevun.



Kuva 1. Elasticsearch vs. Klevu pelkällä korvaavalla toiminnallisuudella



Kuva 2. Elasticsearch vs. Klevu jatkokehityskustannuksilla

6 Soveltuvuusselvitys

Toimeksiantajan ylläpitämä verkkokauppa alusta on koti jopa 1500 verkkokaupalle. Luonnollisesti tämän kokoisen järjestelmän päivitys ei ole hetken työ. Päivitys sisältää testausta ja paljon kehitystä. Siksi tämän työn tavoite ei ole toteuttaa toimivaa hakujärjestelmää vaan selvittää kannattavuutta sekä mahdollisia vastaantulevia haasteita. Soveltuvuusselvityksen toteuttamisessa kuitenkin asennetaan Elasticsearch, kokeillaan asennusprosessia sekä muutamia toimintoja.

6.1 Tarkoitus ja tavoitteet

Selvityksen tarkoitus on saada käsitystä siitä, kuinka tärkeiden ominaisuuksien käyttöönotto toimii Elasticsearchissa. Tavoitteina on siis selvittää, kuinka Elasticsearch otetaan käyttöön, mitkä ovat tärkeitä ominaisuuksia, kuinka vaativaa näiden ominaisuuksien käyttöönotto on. Selvityksestä tarkoituksella siten rajataan pois ison järjestelmän luotettavuus- ja skaalautuvuustestaus. Järjestelmän luotettavuus sekä skaalautuvuus ominaisuutena on äärimmäisen tärkeä, mutta niiden testaaminen tämän tason selvityksessä ei ole realistista. Huomioitavaa on myös se, ettei Elasticsearchin perusversio sisällä käyttäjän todennusta. Käyttäjän todennus tulisi toteuttaa Elasticsearchin maksullisilla lisämoduuleilla tai asentamalla sen verkkoon niin, ettei siihen saa lähiverkon ulkopuolelta yhteyttä.

Tavoitteisiin kuuluu selvittää muutamia erikoistapauksia. Erikoistapauksiin kuuluvat esimerkiksi yhdyssanat, tuotekoodit, mainostettavat tuotteet ja tuotteiden saatavuus. Mainostettavat tuotteet ja tuotteiden saatavuus ovat samantyyppiset erikoistapaukset. Molemmat täytyy käsitellä siten, että tuotteet, jotka on asetettu mainostettaviksi, nostetaan listauksessa korkeammalle eli lähemmäs asiakasta. Samoin varastossa saatavilla olevat tuotteet on nostettava korkeammalle, jottei asiakkaalle ensimmäisenä tarjota tuotetta, jota ei ole varastossa. Tuotekoodit on käsiteltävä erikoistapauksena, sillä tuotekoodit saattavat olla lähellä toisiaan (esimerkiksi XOF-2265 ja XOF-2267). Tuotekoodien samankaltaisuus ei tarkoita lähes vastaavaa tuotetta, vaan kyseessä voi olla aivan eri kategorian tuote. Asiakkaan etsiessä hakutermillä ”XOF-2267” on äärimmäisen todennäköistä, että asiakas haluaa juuri tämän tuotteen ja siinä tapauksessa ensimmäisenä on näytettävä kyseessä oleva tuote. Tuotekoodilla ”XOF-2265” on kuitenkin jätettävä pois tuloksista. Yhdyssanatapaukset ovat itsestään selvät. Asiakkaan etsiessä

”pillifarkkuja” voidaan näyttää myös ”farkut”-hakutermillä löytyvät tuotteet. Ensin tietenkin pillifarkut.

Testauskysymykset:

- “Paita”- ja ”Paidat”-hakusanat palauttavat ”T-Paita” tuotteet sekä ”Paidat”-tuoteryhmään kuuluvat tuotteet.
- “Naisten T-Paita”-haku palauttaa kahdessa eri tuoteryhmässä olevat “T-Paita”-nimiset tuotteet siten, että “Naisten vaatteet/Paidat”-tuoteryhmässä oleva tuote hakutuloksissa ennen “Miesten vaatteet/Paidat”-ryhmässä sijaitsevaa tuotetta.
- “Tummansininen” palauttaa myös siniset tuotteet (tummat ensin) ja “Sininen”-haku palauttaa myös “tummansiniset”.
- Haku tuotekoodilla “ZOF-2267" palauttaa vain tuotteet, joissa tämä termi esiintyy tarkasti. Esim. ei tuotetta “ZOF-2265”.
- Mainostettavaksi merkitty tuote nousee hakutuloksissa normaalia sijoitustaan korkeammalle.
- Varastosta loppunut tuote laskee hakutuloksissa normaalia sijoitustaan matalammalle.

6.2 Elasticsearchin toiminta

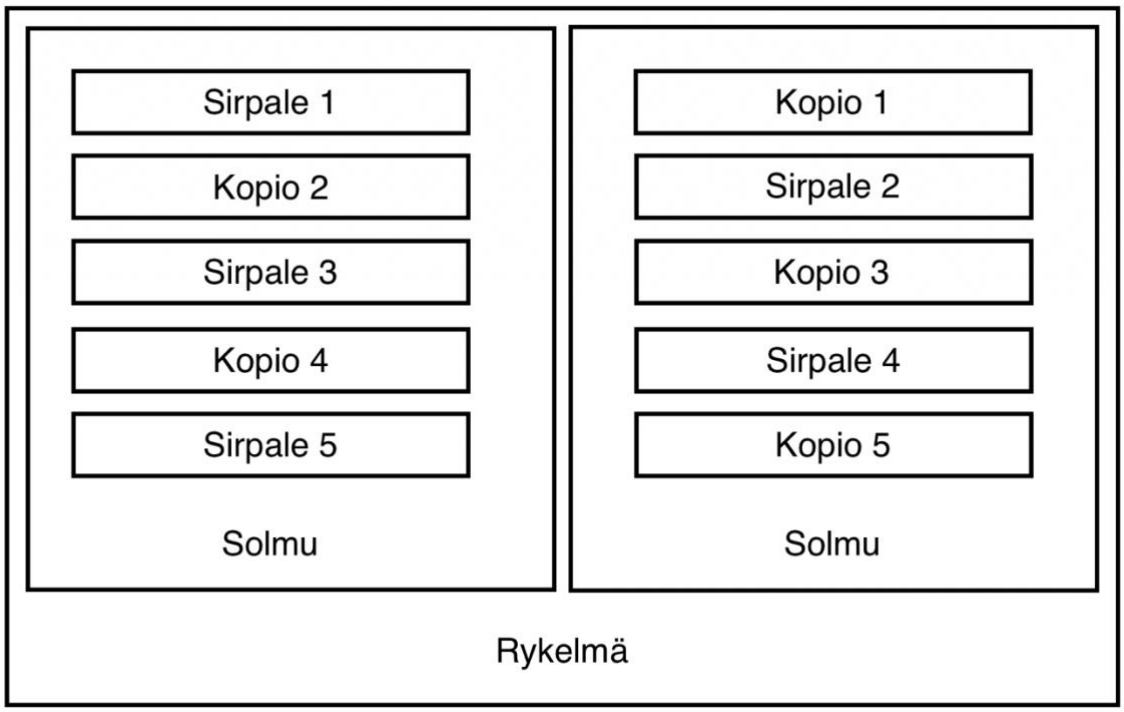
Elasticsearch tukee lähes kaikkia Klevun tarjoamia ominaisuuksia. Vaikka kaikkia ominaisuuksia tuetaan, halusin selvittää, kuinka helppo Elasticsearch ja tietyt ominaisuudet on ottaa käyttöön.

Elasticsearch on todella skaalautuva alusta. Tämä on toteutettu siten, että Elasticsearch kykenee jakautumaan usealle serverille. Isoilla yrityksillä ja yhdistyksillä voi olla useita servereitä mutta vain yksi indeksi, jolta voi hakea. Jos yksi serveri kaatuu, niin toiminnallisuus säilyy.

Yksittäinen Elasticsearch serveri on nimeltään solmu (Node). Elasticsearchissa solmu kuuluu aina rykelmään (Cluster), eli rykelmä on aina ryhmä servereitä. Rykelmässä on sallittua olla yksi tai useampi solmu eli serveri. Asennettaessa ensimmäisen solmun se luo automaattisesti rykelmän nimeltä ”elasticsearch”, joka sisältää vain yhden serverin, solmun, joka juuri asennettiin.

Elasticsearchissa dokumentit tallennetaan ja ovat haettavissa indeksiltä. Indeksiä luotaessa sille määritellään asetukseksi sirpaleiden (Shard) ja kopioiden (Replica) määrä. Sirpaleiden määrä määrittää, kuinka moneen osaan indeksi jakautuu, ja kopioiden määrä määrittelee, kuinka useasti indeksi on kopioitu. Esimerkiksi indeksille voidaan määrittää sirpaleiden määräksi viisi ja kopioiden määräksi yksi. Tällöin luodaan indeksi, jolla on sirpaleet 1, 2, 3, 4 ja 5 sekä kopiot 1, 2, 3, 4 ja 5. Kopioiden määrä on yksi, jolloin jokaisesta sirpaleesta on yksi kopio. Indeksi luodaan rykelmälle, ja indeksille määritellyt sirpaleet ja kopiot jakautuvat automaattisesti rykelmän eri solmuihin. Kopiot jaetaan aina eri solmuun kuin sitä vastaava sirpale, jos vain mahdollista. [12.]

Indeksin sisältämät ja haettavissa olevat dokumentit ovat jakautuneet seuraavan kuvan mukaisesti. Näin ollen toiminnallisuus säilyy, jos toinen solmuista kaatuu. Sillä toisella solmulla on kaatuneiden sirpaleiden kopiot ja se voi edelleen tarjota saman toiminnallisuuden vain yhdellä solmulla. Skaalautuvuus tulee Elasticsearchin mukana. Jos kaksi käyttäjää hakee dataa, joka on sirpaleessa 2, niin toista käyttäjää voidaan palvella kopiosta 2 jne. Käytännöllisesti katsoen kaksinkertaistaa yhtä aikaa palveltavissa olevat käyttäjät.



Kuva 3. Elasticsearch-järjestelmärakenne

Koko Elasticsearch järjestelmä on siten rakennettu, että rykelmään voidaan nyt lisätä solmuja niin paljon kuin tarve vaatii ja sirpaleet sekä kopiot jakautuvat tarpeen mukaan [12].

6.3 Elasticsearchin asennus

Testiympäristön luominen aloitettiin asentamalla Elasticsearchin-testiympäristöön. Tarkoituksena oli suorittaa vain muutamia testejä pienellä datamäärällä, joten asennettiin vain yksi solmu. Solmu luo automaattisesti rykelmän nimellä ”elasticsearch”.

Elasticsearch asennettiin Ubuntu Bionic Beaver käyttöjärjestelmään käyttäen Elasticsearchin ohjeita, jotka on tarkoitettu Debian-pohjaisille käyttöjärjestelmille. Samoja ohjeita voi siis käyttää asennettaessa muillekin Debian-pohjaisille käyttöjärjestelmille. Käyttöjärjestelmän asennuksen yhteydessä ajettiin käyttöjärjestelmäpäivitykset.

Elasticsearch-asennus vaatii PGP-avaimen, jonka avulla turvataan Elasticsearch-asennuspaketin alkuperäisyys. Ohjeen mukaan voidaan tarvita myös apt-transport-https-paketti, joten sekin asennetiin. Viimeinen rivi määrittää säilytyspaikan.

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-
key add -

sudo apt-get install apt-transport-https

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee
 -a /etc/apt/sources.list.d/elastic-7.x.list
```

Seuraavaksi päivitetään käyttöjärjestelmä ja asennetaan Elasticsearch.

```
sudo apt-get update && sudo apt-get install elasticsearch
```

Seuraava komento kertoo, onko käyttöjärjestelmässä käytössä SysV init vai systemd.

```
ps -p 1
```

Jos käytössä on systemd, niin Elasticsearch määritellään käynnistymään käyttöjärjestelmän kanssa seuraavilla komennoilla.

```
sudo /bin/systemctl daemon-reload

sudo /bin/systemctl enable elasticsearch.service
```

Todetaan Elasticsearchin toimivuus lähettämällä GET-pyyntö käyttäen HTTP-protokollaa. Kyseessä olevalla käyttöjärjestelmällä tämä onnistuu helpoiten seuraavalla komennolla.

```
curl -X GET localhost:9200
```

Komennon tulos tulisi olla seuraavanlainen:

Kuva 4. Toimivan Elasticsearchin rykelmän tiedot

Elasticsearchin asennus on valmis. Asennuksen jälkeen Elasticsearch toimii HTTP-pyyntöjen kautta ja voidaan määrittää asetuksia, tallentaa dataa sekä suorittaa hakuja indeksiin. Tässä on esitetty itse tekemäni asennusvaiheet. Asennusprosessi voi olla erilainen riippuen käyttöjärjestelmästä tai muista versioeroavaisuuksista. [13.]

6.4 Ohjelmistorajapinta verkkokauppaan

Testidatan saamiseksi toteutukseen kuuluu hyvin yksinkertainen ohjelmistorajapinta verkkokauppa alustalle. Tämä ohjelmistorajapinta on vain testikäyttöön ja siitä johtuen testidataksi tulee muutamat tuotteet.

Ohjelmistorajapinnalle voi tehdä vain GET-pyynnön ja vastaukseksi se aina palauttaa kaupan nimen, kaupan yksilöivän tunnisteen sekä tuotteet. Tuotteiden data sisältää seuraavat:

ID	Tuotteen yksilöivä tunniste
NAME	Tuotteen nimi
CATEGORY	Tuotteen kategoria
PATH	Tuotteen polku, esimerkiksi /Vaatteet/Paidat/T-Paidat
CODE	Tuotteen ulkoinen yksilöivä tunniste
DESC	Tuotteen kuvaus
FEATURED	Tuotteen mainostus
IN STOCK	Tuotteen saatavuus

6.5 Elasticsearchin indeksin ja asetusten määrittäminen

Elasticsearchissa on paljon määritettävää. Elasticsearch ei vaadi mitään määrityksiä vaan määrittää itse oletetut määritykset datalle, joka lähetetään Elasticsearchiin. Elasticsearch ei kuitenkaan tiedä, mikä tarkoitus datalla on. Täytyy määrittää kaikille kentille sopivat asetukset, jotta saadaan hyviä tuloksia.

Sopivien asetusten löytäminen on aluksi hankalaa. Lähtöpisteen ollessa nollakokemus Elasticsearchista täytyy lähteä liikkeelle Elasticsearchin dokumentaatiosta. Dokumentaatio on täynnä eri asetuksia, mutta lähes kaikille on esimerkkitapauksia. Sen jälkeen, kun on nähnyt muutamia toteutuksia dokumentaatiosta ja internetin foorumipalstoilta, alkaa itse päästä alkuun.

Aluksi testialustassa on hyvin yksinkertainen listaus siitä, mitä kenttiä tallennetaan ja pieni määrä dataa testausta varten. Tässä vaiheessa voi ryhtyä rohkeammin itse kokeilemaan, millaisia tuloksia eri määritykset tuovat. Määrityksiä muuttaessa täytyy datakin uudelleen indeksoida. Uudelleen indeksointiin kuuluu uuden indeksin luominen uusine määrityksineen sekä datan uudelleen indeksoiminen uuteen indeksiin. Indeksin luominen tapahtuu aivan kuten ensimmäistä kertaa mutta datan voi kopioida edellisestä indeksistä olettaen, että siellä on käytössä _source-kenttä, johon tallentuu alkuperäisen dokumentin data ilman muokkausta. Vaihtoehtoisesti datan voi lähettää uuteen indeksiin samoin kuin se lähetettiin ensimmäiseenkin indeksiin. Testatessa datan lähetettiin käyttäen Elasticsearchin bulk API:a, jonka avulla voin yhdellä pyynnöllä lähettää monta dokumenttia kerrallaan. Testaus käytössä oli koko ajan sama data, joten sen uudelleen lähettäminen ei ollut ongelma.

On hyvä huomioida, ettei datan uudelleenindeksointitarve rajoitu pelkästään testaustilanteeseen. Myös tuotantoversiossa uudelleen indeksointi on tarpeellista, mikäli hakujärjestelmän toimintaan halutaan tehdä muutoksia. Vain muutamassa rajatapauksessa ei ole välttämätöntä indeksoida uudelleen. On hyvä, että jo testausvaiheessa on tarpeellista indeksoida dataa uudelleen, sillä se tulee olemaan keskeinen osa hakujärjestelmän kehitystä.

Indeksille oleellisimmat määritykset ovat analysaattorit, tokenisoijat ja suodattimet. Analysaattori määrittää sen, miten data tulkitaan ja kuinka siitä muodostetaan tokenit. Analysaattorille voidaan määrittää tokenisoija ja suodattimia. Analysaattoreita, tokenisoijia ja suodattimia on Elasticsearchissa oletuksena oletusarvoilla määritettynä, mutta kaikkia voidaan muokata ja määrittää omaan käyttöön sopiviksi. Kaikki testauksessa käyttämäni analysaattorit ovat muokattuja, joihin on itse määritetty asetukset.

Tokenisoijat ottavat merkkijonon ja jakavat sen tokeneiksi. Tokeneista muodostuu indeksi, joka on joukko haettavia termejä. Esimerkiksi ”whitespace”-tokenisoija jakaa tekstin sanoiksi joka välilyönnin kohdalta. ”Musta Naisten T-Paita” muuttuu ”Musta”-, ”Naisten”- ja ”T-Paita”-tokeneiksi. Kaikista tokeneista muodostuu lista, ja kun käyttäjä hakee termillä ”Musta”, järjestelmä löytää samanlaisen tokenin ja tietää, missä dokumenteissa termi esiintyy.

Suodattimet ottavat tokenisoijan antavat tokenit ja suodattavat sen suodattimen mukaisesti. Yhdelle analysaattorille voidaan määrittää nolla tai useampia suodattimia. Esimerkkinä ”lowercase”- suodatin pienentää kaikki tokenit. ”Musta”-tokeni muuttuu ”musta”-tokeniksi. Tällöin vaikka käyttäjä hakee termillä ”mUsta”, löydetään oikea tokeni ja voidaan palauttaa olennainen tulos. Suodatin voi myös lisätä tokeneita. Esimerkiksi synonyymisuodattimelle voidaan itse määrittää, mitkä sanat ovat synonyymejä. Synonyymisuodatin lisää dokumentille kaikkien dokumentissa esiintyvien sanojen synonyymit, jotta synonyymihaku voi palauttaa tuloksen.

6.6 Testialustan määrittäminen

Testausta varten määriteltiin analysaattorit ”finnish”, ”code” ja ”code_search”, suodattimet ”finnish_stop”, ”finnish_stemmer” ja ”synonym” sekä tokenisoija ”code_tokenizer”. Näillä määrityksillä päästiin lähimmäksi soveltuvuusselvityksen tavoitteiden saavuttamista. Määritysten löytäminen vaati paljon testausta, selvitystä ja kokeilua. Täytyy vain määrittää uudet asetukset indeksille, syöttää data ja kokeilla, miten hakujärjestelmä reagoi.

Analysaattori ”finnish” käyttää tokenisoijana Elasticsearchin oletusarvoista ”standard”-tokenisoijaa sekä suodattimia ”lowercase”, "finnish_stop", "finnish_stemmer" ja "synonym". ”standard”- tokenisoija jakaa merkkijonon Unicoden tekstisegmentointialgorytmin mukaisesti. Käytännössä se erottelee sanat. ”lowercase- suodatin on Elasticsearchin oletusarvoinen suodatin, joka pienentää aakkoset. ”finnish_stop” on stop-tyyppinen suodatin, joka on määritetty käyttämään Elasticsearchin omaa listaa suomen kielen poistosanoista ja se estää poistosanojen päätymistä tokeni-indeksiin. Poistosanoja ovat etuliitteet, sidesanat ja prepositiot jne. ”finnish_stemmer” on suomen kielen stemmeri, joka typistää suomen kielen sääntöjen mukaisesti muuttuvat sanaliitteet ja johtimet pois sanojen perästä. Esimerkiksi ”pyörät”-sanasta indeksoidaan ”pyörä”. ”synonym” on synonym -tyyppinen suodatin, jolle on tässä tapauksessa

määritetty esim. ”hattu, lakki”. Synonym-suodatin käyttää sille määritettyä synonyymisanastoa ja törmätessään sanaan, joka löytyy synonyymisanastosta, indeksoi myös kyseisen sanan synonyymit. Tässä tapauksessa, jos indeksoidaan sana ”hattu”, niin indeksoidaan myös ”lakki”.

Synonyymisuodatin on testitapauksessa määritelty indeksointiajan analysaattorille, mikä tarkoittaa, että synonyymit indeksoidaan hakujärjestelmään. Sama toiminnallisuus on kuitenkin saavutettavissa määrittämällä suodatin hakuajan analysaattorille. Hakuajan analysaattori määrittelisi esimerkkitapauksessa ”hattu”-hakusanalla hakiessa haettaviksi tokeneiksi myös ”lakki”-termin. Jos synonyymit indeksoidaan hakujärjestelmään, niin pelkkä synonyymisanaston päivitys vaatisi uudelleenindeksoinnin.

Analysaattori ”code” käyttää itse määritettyä tokenisoijaa ”code_tokenizer” ja suodatinta ”lowercase”. Tämä analysaattori on määritetty vain ulkoisia yksilöllisiä tuotekoodeja varten. ”code_tokenizer” on edge_ngram -tyyppinen tokenisoija. Se muodostaa tokenit alusta loppuun jakamalla sanan pieniin osiin. Tokenisoija on määritetty paloittelemaan minimissään kuudesta merkistä ja maksimissaan kahteenkymmeneen merkkiin asti. Lisäksi määrittelin sen käyttämään kirjaimia, numeroita sekä välimerkkejä mukaan lukematta välilyöntiä. Esimerkkinä analysaattorin toiminnasta tuotekoodi ”ZOF-2265” muuttuu tokeneiksi ”zof-22”, ”zof-226” ja ”zof-2265”. Ensimmäinen tokeni on siis 6 merkkiä pitkä ja sitten luodaan aina yhden merkin verran pidempi tokeni, kunnes merkit loppuvat tai tulee maksimimerkkimäärä eli kaksikymmentä täyteen. Käytännössä tämä palauttaa tuotteet, joiden tuotekoodin alku tai koko koodi täsmää hakusanaan.

Analysaattori ”code_search” on määritetty käyttämään ”lowercase’’-suodatinta sekä ”whitespace”-tokenisoijaa. ”whitespace”-tokenisoija ei vaadi määrittelyä. Se jakaa merkkijonon tokeneiksi välilyöntien kohdalta. Kyseistä analysaattoria käytetään ainoastaan tuotekoodien hakutermien analysointiin.

```
"settings": {
  "analysis": {
    "filter": {
      "finnish_stop": {
        "type":       "stop",
        "stopwords":  "_finnish_"
      },
      "finnish_stemmer": {
        "type":       "stemmer",
        "language":   "finnish"
      },
      "synonym" : {
        "type" : "synonym",
        "synonyms" : [
          "paita, paida",
          "hattu, lakki"
        ]
      }
    },
    "analyzer": {
      "finnish": {
        "tokenizer":  "standard",
        "filter": [
          "lowercase",
          "finnish_stop",
          "finnish_stemmer",
          "synonym"
        ]
      },
      "code": {
        "tokenizer":  "code_tokenizer",
        "filter": [
          "lowercase"
        ]
      },
      "code_search": {
        "tokenizer":  "whitespace",
        "filter": [
          "lowercase"
        ]
      }
    },
    "tokenizer": {
      "code_tokenizer": {
        "type": "edge_ngram",
        "min_gram": 6,
        "max_gram": 20,
        "token_chars": [
          "letter",
          "digit",
          "punctuation"
        ]
      }
    }
```

Kuva 5. Testialustan indeksin settings -määritykset

```
"mappings": {
  "products": {
    "properties": {
      "id":       {
        "type": "text",
        "analyzer": "code"
      },
      "name":     {
        "type": "text",
        "analyzer": "finnish"
      },
      "category": {
        "type": "text",
        "analyzer": "finnish"
      },
      "path":     {
        "type": "text",
        "analyzer": "finnish"
      },
      "code":     {
        "type": "text",
        "analyzer": "code",
        "search_analyzer": "code_search"
      },
      "desc":     {
        "type": "text",
        "analyzer": "finnish"
      },
      "featured": {
        "type": "text"
      },
      "in_stock":    {
        "type": "integer"
      }
    }
  }
}
```

Kuva 6. Testialustan kenttien mappings -määritykset

```
"should" : [
    {
        "match" : {
            "name" : {
              "query": "farkut",
              "boost": 0.5
            }
        }
    },
    {
        "match" : {
            "desc" : {
              "query": "farkut",
              "boost": 0.2
            }
        }
    },
    {
        "match" : {
            "category" : {
              "query": "farkut"
            }
        }
    },
    {
        "match" : {
            "path" : {
              "query": "farkut",
              "boost": 0.8
            }
        }
    },
    {
        "match" : {
            "code" : {
                "query": "farkut",
                "boost": 10
            }
        }
    },
    {
        "term" : {
            "featured" : {
                "value": "1",
                "boost": 0.4
            }
        }
    },
    {
        "term" : {
            "in_stock" : {
                "value": 1,
                "boost": 0.4
            }
```

Kuva 7. Testitapausten Bool -hakujen rakenne

6.7 Testaus ja tulokset

Testauksen aikana selkeytyi uudelleenindeksoinnin tarve sekä metodit. Data on uudelleen indeksoitava, kun indeksin määritelmiä muutetaan. Tämä tuli hyvin selväksi testiympäristöä määrittäessä sekä testitapausten toteutuksessa. Tuotantokäyttöön tuleva järjestelmä täytyy suunnitella siten, että sen sisältämä data on helposti uudelleenindeksoitavissa. Etenkin aluksi muutoksia indeksin määrityksiin tulee varmasti ja niiden tuotantoon saanti vaatii uudelleenindeksoinnin. Testatessa kokeiltiin kahta eri uudelleenindeksointitapaa. Uudelleenindeksointirajapinta (Reindex API) on Elasticsearchin ominaisuus, joka on tarkoitettu auttamaan uudelleenindeksoinnissa sekä uuden indeksin luomisessa ja datan lähetyksessä uuteen indeksiin.

Toimeksiantajan tapauksessa on kannattavampaa luoda uusi indeksi ja päivittää data uuteen indeksiin. Tähän syynä on se, että Elasticsearchin uudelleenindeksointirajapinta vaatii kaiken datan tallennuksen Elasticsearchin sisäisesti. Toimeksiantajan verkkokaupoissa kaikki data on jo tallennettuna verkkokauppojen tietokannoissa, joten ei välttämättä ole kannattavaa tallentaa kaikkea myös Elasticsearchin sisäisesti. Toinen syy on se, että toimeksiantajan toteutukseen tulee uusien kauppojen luominen ja niiden indeksoiminen tyhjään indeksiin. Huomioiden tämän ja sen, ettei toimeksiantajan nykyisellä toimintasuunnitelmalla ole tarvetta tallentaa dataa Elasticsearchiin on kannattavampaa toteuttaa uudelleen indeksointi näin.

Testitapaukset ovat kaikki hakutilanteita, joissa oletetaan tiettyä tulosta tietyille hakutermeille. Soveltuvuusselvityksessä saavutettu toiminnallisuus toteutettiin kokonaan Bool -kyselyllä. Bool -kyselyssä hakutermiä haetaan kaikilta tuotteen kentiltä yhtä aikaa. Esimerkissä käytetään ”should”-ehtoa, eli ei haittaa, vaikka kentät eivät saa osumia. Mitä useammasta kentästä hakutermi löytyy, sitä suuremman pisteen dokumentti saa ja sitä olennaisempi tulos. Hakutulokset palautetaan pistejärjestyksessä suuremmasta pienempään. Hakukenttien merkittävyyttä voi nostaa ”boost”-arvolla. Esimerkiksi jos ”name”-kentälle annetaan ”boost” -arvoksi 3, niin dokumentit, joilla hakutermi löytyy ”name”-kentästä, ovat arvokkaampia kuin dokumentit, joilla hakutermi esiintyy useammin mutta ei ”name”-kentässä.

- “Paita”- ja ”Paidat”-hakusanat palauttavat ”T-Paita” tuotteet sekä ”Paidat” tuoteryhmään kuuluvat tuotteet.

Tämä testitapaus saatiin toimimaan synonyymisuodattimen ja stemmauksen avulla. ”Paita”-hakutermi palauttaa ”T-Paita”-tuotteet ilman kumpaakaan edellä mainituista määrityksistä, sillä standardi tokenisoija luo ”T-Paita”-tuotteelle tokenit ”T” ja ”Paita”. Tokeni ”Paita” yhdistyy hakutermiin. Samoin ”Paidat”-haku palauttaa ”Paidat”-kategorian tuotteet, sillä käytössä ovat samat termit.

Haastavia tapauksia ovat saada ”Paita”-hakutermi palauttamaan ”Paidat”-tuoteryhmään kuuluvat tuotteet sekä ”Paidat”-hakutermin palauttamaan ”T-Paita”-nimiset tuotteet. Stemmaus typistää ”Paidat”-termit ”Paida”-termeiksi. Silti ”Paita”-hakutermi ei löydä tuotetta, koska se on indeksoitu ”Paida”-tokenilla. Tätä varten määriteltiin synonyymisuodatin, jonka synonyymilistaan on merkitty ”paita, paida”. Tämän ansiosta tuotteet indeksoidaan molemmilla tokeneilla. Eli ”Paidat”-tuoteryhmään kuuluvat tuotteet ja ”T-Paita”-nimiset tuotteet indeksoidaan tokeneilla ”Paita” ja ”Paida”.

Näiden määritysten avulla testitapaus saatiin toimimaan mutta huomioitavaa on, että synonyymilistaan on listattu vain yhden sanan kaksi muotoa. Vastaavia tapauksia on muillakin sanoilla ja on oikeita synonyymejä, jotka täytyy huomioida myös. Tämä täytyy ottaa huomioon, kun mietitään, onko kannattavaa kehittää Elasticsearchia. Synonyymilistoja suomen kielelle on varmasti olemassa mutta se on implementoitava toimeksiantajan järjestelmään ja sitä on ylläpidettävä.

- “Naisten T-Paita” -haku palauttaa kahdessa eri tuoteryhmässä olevat “T-Paita”-nimiset tuotteet siten, että “Naisten vaatteet/Paidat”-tuoteryhmässä oleva tuote hakutuloksissa ennen “Miesten vaatteet/Paidat”-ryhmässä sijaitsevaa tuotetta.

Haettaessa hakusanoilla ”Naisten T-paita” analysaattori muodostaa hakusanoista kolme hakutermiä: ”naisten”, ”t” ja ”paita”. ”t”- ja ”paita” -termit löytävät myös miesten T-paidat, mutta hakutermi ”naisten” nostaa naisten T-paitojen arvoa, koska niillä paidoilla on ”path”-kentän kohdalla ”Naisten vaatteet/Paidat”. Tämän pitäisi olla selkeä tapaus, mutta en saanut hakuarvoja muuttamalla tapausta toimimaan halutulla tavalla. Kuitenkin vaikuttaa siltä, että tämä on yksinkertaisesti toteutettavissa.

Eri kenttien arvoja täytyy säädellä, jotta saadaan hakujärjestelmä palauttamaan tulokset olennaisessa järjestyksessä. Kuvassa 7 näkyy eri kentät ja niille asetettu ”boost”-arvo, jota käytetään haun aikana.

- “Tummansininen” palauttaa myös siniset tuotteet (tummat ensin) ja “Sininen”-haku palauttaa myös “tummansiniset”.

Suomenkieliset yhdyssanat ovat haaste. Niiden erotteluun vaaditaan esimerkiksi yhdyssanakirjasto, johon on lueteltu säännöt yhdyssanojen erotteluun. Elasticsearchin ohjeet ehdottaakin kirjastoa, joka tukee myös suomen kieltä, mutta tässä selvityksessä ei saatu siitä toiminnallista. Tämä on varmasti sellainen ongelmakohta, joka lisää kehitys- sekä ylläpitokustannuksia.

- Haku tuotekoodilla “ZOF-2267" palauttaa vain tuotteet, joissa tämä termi esiintyy tarkasti. Esim. ei tuotetta “ZOF-2267”.

Tässä testitilanteessa haluttiin selvittää, voidaanko tehdä hakujärjestelmä, jolle voi antaa termiksi tismalleen tuotekoodin. Tapaus saatiin onnistumaan ilman suurempia ongelmia. Toiminnan saavuttamiseen käytettiin ”edge_ngram”-tyyppistä tokenisoijaa, joka määriteltiin tapaukselle sopivaksi. Tämä tokenisoija määriteltiin siten, että se ei katkaise tokenia välimerkeillä kuten väliviivalla, sillä nämä ovat yleisiä tuotekoodeissa. Tuotteita on ainoastaan järkevä etsiä täsmälleen osuvalla koodilla, sillä tuotteet, joiden tuotekoodi eroaa yhdelläkin merkillä, voivat olla eri tuotteita eri tuoteryhmistä. Siitä huolimatta testauksessa lisättiin aakkosten pienennys, sillä harvoin tuotekoodit eroavat pelkästään aakkosten koon mukaan.

Testitapausta varten määriteltiin myös eroava hakuajan analysaattori. Kuvassa 6 näkyy erikseen määritelty hakuajan analysaattori. Hakuajan analysaattori pienentää aakkoset samoin kuin indeksointiajan analysaattori mutta hakuaikana analysaattori ainoastaan katkaisee tokenit välilyöntien kohdalta. Tuotekoodeissa ei koskaan ole välilyöntejä, joten on turvallista olettaa tokenin katkeavan siitä. Hakuaikana haetaan tuotekoodia sellaisenaan kuin se on hakuterminä annettu, joten sille ei tehdä muita muutoksia. Viimeisenä varmistetaan, että kun tuotekoodin hakukentässä löytyy tulos, tulisi tulosta vastaava tuote ensimmäiseksi tulokseksi listaukseen. Se toteutettiin antamalla tuotekoodi hakukentälle ”boost”-arvoksi 10.

Tässä on huomioitava yksi seikka toiminnasta. Koska tuotekoodi ”ZOF-2265” indeksoidaan tokeneilla ”zof-22”, ”zof-226” ja ”zof-2265”, niin tuote, jonka tuotekoodi on ”ZOF-226”, tulkitaan hakutilanteessa saman arvoiseksi kuin tuote ”ZOF-2265” haettaessa termillä ”ZOF-226”. Tästä johtuen tämän tyyppinen ratkaisu ei välttämättä ole kannattavaa tuotantokäytössä tai sitä täytyy jotenkin muuttaa, jotta saataisiin toivottu toiminnallisuus. Tuotekoodit indeksoidaan tässä testauksessa näin, jotta hakutuloksiin tulee myös tuote ”ZOF-2265”, kun haetaan termillä ”ZOF-226”.

- Mainostettavaksi merkitty tuote nousee hakutuloksissa normaalia sijoitustaan korkeammalle.

Tässä on selkeä testitapaus. Toivottu toiminnallisuus saatiin toteutettua ”boost”-arvolla. Kyselyssä ”featured”-kentältä ei kysytä hakutermiä vaan kysytään ainoastaan, onko se ”1”. Mikäli ”featured”-kenttä on ”1”, silloin se on mainostettava ja dokumentti saa ”boost”-arvoksi 0.4. Arvoa voi tuotantokäytössä helposti muuttaa vaikka kauppakohtaisesti siten, että jokainen kauppa saa toivotun tuloksen. Käytössä olevalla testidatalla 0.4 oli tarpeeksi, jotta mainostettava tuote tuli muiden edelle.

- Varastosta loppunut tuote laskee hakutuloksissa normaalia sijoitustaan matalammalle.

Tämä tapaus toteutettiin samalla tavalla kuin edellinen. Kyselyssä ”in_stock”-kentältä kysyttiin, onko sen arvo 1, ja mikäli on, se tarkoittaa, että sitä on saatavilla varastossa. Siinä tapauksessa tälle tuotteelle annettiin suurempi arvo käyttäen ”boost”-ominaisuutta. Elasticsearchissa ei voida antaa negatiivista ”boost”-arvoa, joten emme voi laskea tuotteiden pisteitä, joita ei ole varastossa. Nostamalla varastossa olevien tuotteiden arvoa saamme aikaan halutun toiminnallisuuden.

7 Tulokset

Tutkimuksen tarkoituksena oli selvittää, voiko toimeksiantaja toteuttaa sisäisesti toimivan tuotehakujärjestelmän, jossa toimii luotettavasti halutut ominaisuudet. Olennaista tässä tutkimuksessa oli se, onko sisäisen tuotehaun toteuttaminen kannattavaa resurssien sekä taloudellisuuden kannalta.

Kannattavuutta täytyy pohtia huolellisesti monelta eri kannalta, ennen kuin sijoittaa johonkin huomattavan määrän resursseja. Toimeksiantajan yrityksessä halutaan korvata Klevun tuotehakujärjestelmä Elasticsearchilla, mutta kun katsomme kannattavuutta yrityksen näkökulmasta, niin pelkkä Klevun kulujen vertaaminen Elasticsearchin kuluihin ei riitä. Täytyy ottaa huomioon, että Elasticsearchilla on paljon muita käyttömahdollisuuksia ja toimeksiantajalla on suunnitelmia Elasticsearchin hyödyntämiselle muissa projekteissa. Elasticsearchin käyttöönotto tuotehakualustana ei siis pelkästään leikkaa Klevun järjestelmän kuluja kokonaan pois, vaan myös tuo paljon arvoa yritykselle lisääntyneen osaamisen kautta. Näin ollen Elasticsearchista saatavat hyödyt eivät ole suoraan selvitettävissä. Vielä ei voida tietää, mihin Elasticsearchia tullaan hyödyntämään, mutta kun tiedetään, että sitä aiotaan hyödyntää, niin tuotehakujärjestelmän toteutus käyttäen Elasticsearchia olisi yksi keino tuoda lisää osaamista yritykseen. Kappaleessa 5 tähän liittyvät kuvat.

Tuloksia pyrittiin saamaan kartoittamalla Klevun tarjoamat ominaisuudet ja toiminnallisuus sekä selvittämällä Elasticsearchilla toteutettavissa olevia ominaisuuksia. Toimivalta hakujärjestelmältä toivottiin vähintään sama toiminnallisuus kuin Klevulla saadaan tällä hetkellä. Erityisesti toivottiin hakuanalytiikkaa, mikä ei Klevulla yrityksen työntekijöiden kokemusten mukaan toiminut ja se on yksi syy, miksi Elasticsearchia lähdettiin tutkimaan.

Tutkimuksessa selvisi Klevun tarjoamat ominaisuudet sekä puutteet. Klevun tarjoamat ominaisuudet sekä puutteet ovat toteutettavissa Elasticsearchilla. Tämä on todettu tutkimalla sekä osittain toteuttamalla soveltuvuusselvityksellä. Tutkimuksessa selvisi, että Elasticsearch-alusta sisältää sisäänrakennetun skaalautuvuusominaisuuden, jota ei ole samanlaista muissa hakujärjestelmissä. Toimeksiantajan kokemusten kautta selvisi, että Klevulla on skaalautuvuushaasteita. Etenkin indeksointiaikana muu järjestelmä hidastelee. Soveltuvuusselvitys auttaa kartoittamaan yritykselle vaadittujen ominaisuuksien toteutuksen haastavuuden sekä toteutukseen vaadittavien resurssien tarpeen.

Soveltuvuusselvityksessä selvisi, että osa ominaisuuksista saatiin toimimaan helposti ja yksinkertaisesti. Kaikkia ominaisuuksia ei saatu toteutettua selvityksessä. Nekin tapaukset, joita ei saatu toteutettua, auttavat toimeksiantajaa kartoittamaan toteutukseen vaadittavia resursseja. Yksi ominaisuus, mitä ei saatu toimimaan testiympäristössä, oli yhdyssanat kuten tummansininen tai polkupyörä. Tämä ominaisuus ei kykene erottamaan yhdyssanoja ja sen vuoksi haun tarkkuus heikkenee. Tutkimuksessa selvisi, että Elasticsearchiin on mahdollista saada yhdyssanakirjasto, joka mahdollistaa yhdyssanojen erotuksen myös suomen kielelle. Tämä vaatii lisää tutkimusta mutta tieto tästä mahdollisuudesta auttaa arvioimaan lopullisen toteutuksen vaativuutta.

Toimenpiteet

Toimeksiantaja tekee tämän tutkimuksen perusteella päätöksen toimenpiteistä. Tällä hetkellä toimenpiteistä ei ole vielä tietoa. Tutkimuksen tuomia tuloksia ja ymmärrystä kuitenkin on tarkoitus hyödyntää yrityksen erilaisissa Elasticsearch-projekteissa.

Tutkimuksen tulosten perusteella toimeksiantajan on kannattavaa kehittää tuotehakujärjestelmä käyttäen Elasticsearch alustaa. Elasticsearchin kehitystä voisi aloittaa toisesta projektista, joka ei olisi niin vaativa toteutus kuin tuotehaku. Elasticsearch projekteissa kannattavaa olisi keskittyä täyden tekstin haun, uudelleenindeksoinnin sekä tulosten oleellisuuden säätämisen tuomiin haasteisiin ottaen huomioon, kuinka paljon kehitysmahdollisuuksia menee hukkaan ilman toimivaa hakuanalytiikkaa. Huomioitava on myös toimeksiantajan suunnitelmat Elasticsearchin käytölle muissakin projekteissa. Nämä seikat huomioiden tutkimuksen toimenpide-ehdotus on jatkaa Elasticsearchin toteutuksen suunnittelua.

8 Pohdinta

Tutkimuksessa haastavaa oli selvittää nykyisen järjestelmän, Klevun, toimintaa sillä Klevusta ei ole paljoa tietoa. Eniten tietoa nykyisen toteutuksen toiminnasta on haettu toimeksiantajan kautta. Elasticsearchin ja Klevun rehellinen vertailu on haastavaa, sillä useat Elasticsearchin ominaisuudet vaativat kehitystyötä toimiakseen. Sitä helpottaa soveltuvuusselvitys, jossa ominaisuuksien käyttöönoton haastavuutta kyettiin selkeyttämään.

Tutkimus on kokonaisuutena hyvin avaava haasteille, joita kohtaa kehittäessä hakujärjestelmää. Esimerkkinä kappaleessa 6.5 kerrotaan hakujärjestelmän uudelleenindeksointitarpeesta. Tarpeen olemassaolo ei ole salaisuus, mutta se ei ole sanomatta heti selkeää. Uudelleenindeksointi on tarpeellista aina, kun muutetaan kenttien analysointia ja se tulee lisäämään haasteita.

Tutkimuksen loppupuolella havaittiin vika soveltuvuusselvityksen testitapauksessa, jossa haettiin tuotekoodeja. Vika tulee, kun tuotekoodin mitta on yli 20 merkkiä. Tämä on huolimattomuudesta johtunut vika, mutta tapaukseen on yksinkertainen ratkaisu, joka parantaa tulosten olennaisuutta. Vika ratkaistaan siten että tuotekoodi indeksoidaan kahdesti: kerran testitapauksen osoittamalla tavalla ja toisen kerran ilman ”egde_ngram” -suodatinta. Tuloksena tuotekoodi on täsmällisesti haettavissa, oli se minkä mittainen tahansa ja lyhyempiin tuotekoodeihin tulosten olennaisuus paranee. Toimeksiantajan suunniteltavaksi jää, miten tapaus toteutetaan ja käytetäänkö ”edge_ngram” -suodatinta ollenkaan.

Tutkimus onnistui tavoitteessaan selvittää nykyisen järjestelmän ominaisuudet sekä selvittää, mitä hakujärjestelmän toteutus vaatii. Toimeksiantajalle jää päätettäväksi kappaleessa 6.6 mainittu synonyymisuodattimen toteutus. Synonyymisuodattimen toteutuksen voi hoitaa eri tavoin, mutta toimeksiantajan täytyy suunnitella omaan ratkaisuun sopiva toteutus. Toimeksiantaja päättää toimenpiteistä tutkimuksen pohjalta myöhemmin, mutta tutkimuksen tulosten perusteella Elasticsearchin käyttöönotto vaikuttaa kannattavalta.

Toimeksiantaja voi tyytyä Klevuun nojaten siihen, että se toimii mutta heikosti. Klevun kanssa toimiessa kuitenkin usein herää toiveita hakujärjestelmän kehityksestä, eikä se ole mahdollista Klevun kanssa. Klevun toteutuksen suurin ongelma tällä hetkellä on hakuanalytiikan puute, ja ilman hakuanalytiikkaa on hankala arvioida edes, kuinka nykyistä hakujärjestelmää käytetään. Lisäksi, kuten jo tutkimuksessa on mainittu, toimeksiantajalla on kokemusta Elasticsearchin

käytöstä sekä suunnitelmia muihinkin käyttötarkoituksiin. Nämä huomioiden mielestäni on selkeää, että Elasticsearchin käyttöönottoa kannattaa suunnitella.

Tutkimuksessa olen pyrkinyt huomioimaan työn merkitsevyyttä myös muille yrityksille, jotka suunnittelevat hakujärjestelmää käyttäen Elasticsearchia alustana. Vaikka tutkimuksessa puhutaan ja keskitytään verkkokauppoihin ja verkkokauppojen tuotteisiin, niin etenkin soveltuvuusselvityksessä esitetyt testitapaukset voivat olla avuksi myös muunlaisissa projekteissa. Teoriaosuudet Elasticsearchin toiminnasta voivat myös auttaa muita ymmärtämään paremmin Elasticsearchin toimintaa.

Lähteet

(1) Haastattelu Ville Kervisen, Ismo Ruotsalaisen ja Timo Korhosen kanssa. 22.02.2019.

(2) Turnbull D, Berryman J. Relevant search. Shelter Island: Manning; 2016:12-15.

(3) Klevu. Saatavilla 24.03.2019. https://www.klevu.com/

(4) Klevu Languages. Saatavilla 24.03.2019. https://support.klevu.com/faq/faqs/what-languages-are-supported-by-klevu-search/

(5) Manda Sai Divya, Shiv Kumar Goyal. ElasticSearch: An advanced and quick search technique to handle voluminous data. Compusoft 2013:171-174

(6) Gormley C, Tong Z. Elasticsearch: The Definitive Guide. 1st ed. Sebastopol: O'Reilly Media, Incorporated; 2015.

(7) NASA: Unlocking Interplanetary Datasets with Real-Time Search. Saatavilla 24.03.2019. https://www.elastic.co/elasticon/2015/sf/unlocking-interplanetary-datasets-with-real-time-search

(8) Marr B. Amazing Big Data At NASA: Real Time Analytics 150 Million Miles From Earth. Forbes 14.04.2016.

(9) Samuel Scott. These 15 Tech Companies Chose the ELK Stack Over Proprietary Logging Software. 17.02.2016.

(10) Klevu Features. Saatavilla 24.03.2019. https://www.klevu.com/features/

(11) Elasticsearch. Saatavilla 24.03.2019. https://www.elastic.co/products/elasticsearch.

(12) Elasticsearch Getting started Basic concepts. Saatavilla 24.03.2019. https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started-concepts.html

(13) Install Elasticsearch with Debian package. Saatavilla 24.03.2019. https://www.elastic.co/guide/en/elasticsearch/reference/current/deb.html#