jamk

Web-pohjaisen SaaS-palvelun kehittäminen

Case: Akeneo EasyCatalog InDesign Connector

Markus Haverinen

Opinnäytetyö, AMK
Huhtikuu 2022
Tietojenkäsittely ja tietoliikenne
Insinööri (AMK), tieto- ja viestintätekniikka

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Haverinen, Markus

Web-pohjaisen SaaS-palvelun kehittäminen Case: Akeneo EasyCatalog InDesign Connector

Jyväskylä: Jyväskylän ammattikorkeakoulu. Huhtikuu 2022, 34 sivua

Tietojenkäsittely ja tietoliikenne. Tieto- ja viestintätekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Verkkojulkaisulupa myönnetty: Kyllä

Tiivistelmä

Vaikka viime vuosina moni asia on digitalisoitu, on useilla yrityksillä edelleen tarvetta tuottaa erilaisia fyysisiä tuotekuvastoja, katalogeja, esitteitä sekä ohjeita. Näiden tuottamiseen tarvitaan erilaisia tuotetietoja, joissa on hyvin tärkeää se, että tiedot ovat helposti saatavilla sekä ajan tasalla. Näin ollen markkinoilla on kysyntää ratkaisulle, joka mahdollistaisi rikkaan tuotetiedon viennin taitto-ohjelmalle nopeasti ja helposti.

Opinnäytetyön toimeksiantaja oli Vincit Jyväskylä Oy, joka on osa Vincit Oyj:tä. Työn tavoitteena oli kehittää palvelu, joka mahdollistaa tuotetietojen hakemisen ja suodattamisen Akeneo PIM-tuotetiedonhallintajärjestelmästä. Tuotetiedoista muodostetaan XML-sanoma, joka viedään Adobe InDesign taitto-ohjelman EasyCatalog-lisäosalle.

Palvelu toteutettiin multi-tenant SaaS-palveluna, joka tarkoittaa sitä, että palvelusta olisi vain yksi instanssi ajossa, ja jota sitten käyttää useampi asiakas. Toteutuksessa hyödynnettiin React.js ja Node.js tekniikoita, Docker-konttiteknologiaa, PostgreSQL-tietokantaa sekä Google Cloud Platform-pilvialustaa.

Lopputuloksena saatiin täysin toimiva SaaS-palvelu, jossa käyttäjä voi itse luoda erilaisia konfiguraatioita yksinkertaisen käyttöliittymän avulla. Konfiguraatioiden avulla suodatetaan tuotetietoja Akeneo PIM-tuotetietojenhallintajärjestelmästä ja suodatetuista tuotetiedoista muodostetaan XML-sanoma, joka on tarkoitus viedä InDesign taitto-ohjelmalle. Sanomaa voidaan myös hyödyntää muissakin kohteissa tarpeen mukaan, eikä se ole lukittu pelkästään InDesignin käyttöön.

Opinnäytetyön tärkeimmät tavoitteet saavutettiin ja toteutettu sovellus saatiin käyttöön usealle asiakkaalle. Sovellus vietiin myös Akeneon kauppapaikalle tarjolle. Palvelun ansiosta erilaisten kuvastojen, katalogien, yms. tuottaminen nopeutui, kun XML-muotoista tuotesanomaa ei tarvitse enää muodostaa käsin.

Avainsanat (asiasanat)

Akeneo PIM, SaaS, tuotetietohallintajärjestelmä, web-sovellus

Muut tiedot (salassa pidettävät liitteet)

jamk

Description

Haverinen, Markus

Development of a web-based SaaS service: Case: Akeneo EasyCatalog InDesign Connector

Jyväskylä: JAMK University of Applied Sciences, April 2022, 34 pages.

Information and communication technology. Degree Programme in Information and Communication Technology. Bachelor´s thesis.

Permission for web publication: Yes

Language of publication: Finnish

Abstract

Although many things have gone digital in recent years, many companies still need to produce a variety of physical product catalogues, brochures, and instructions. These require a variety of product information, where it is very important that the information is easily accessible and up to date. Thus, there is a demand in the market for a solution that would allow exporting rich product information to a layout program in a quick and easy way.

The client of the thesis was Vincit Jyväskylä Oy, which is part of Vincit Oyj. The aim of the thesis was to develop a service that enables the retrieval and filtering of product data from the Akeneo PIM product information management system. The product data is converted to an XML message, which is then exported to the EasyCatalog add-on of the Adobe In-Design layout software.

The service was implemented as a multi-tenant SaaS service, which means that only one instance of the service would be running, and then used by multiple customers. The implementation made use of React.js and Node.js technologies, Docker container technology, PostgreSQL database and Google Cloud Platform cloud services.

The end result was a fully functional SaaS service that allows the user to create different configurations through a simple user interface. The configurations are used to filter product data from the Akeneo PIM product information management system. An XML message is then generated from the filtered product data which is then exported to the InDesign layout software. The message can also be used in other applications as needed and it is not locked to InDesign alone.

The main objectives of the thesis were achieved, and the implemented application was made available to several clients. The application was also made available on the Akeneo marketplace. Thanks to the service, the production of various catalogues, brochures, etc. was simplified, as the XML message no longer has to be generated manually.

Keywords/tags (subjects)

Akeneo PIM, SaaS, product information management system, web-application

Miscellaneous (Confidential information)

Sisältö

Kuviot

Lyhenteet

XML	Extensible Markup Language
JSON	JavaScript Object Notation
URL	Uniform Resource Locator
MIT	Massachusetts Institute of Technology
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure

1 Johdanto

1.1 Toimeksiantaja

Tämän opinnäytetyön toimeksiantajana toimii Vincit Jyväskylä Oy, joka on Vincit Oyj:n tytäryhtiö. Vincit on suomalainen vuonna 2007 Tampereen Hervantaan perustettu teknologiayritys, jonka keskeisiin arvoihin kuuluu työntekijöiden tyytyväisyys ja hyvinvointi, ja se että aamulla ei harmita mennä töihin. Vincitillä on Suomessa seitsemän toimipistettä, jotka sijaitsevat Tampereella, Turussa, Jyväskylässä, Kuopiossa, Oulussa ja kaksi Helsingissä. Toimipisteitä on myös neljä Yhdysvalloissa sekä yksi Sveitsissä. (Mikä ihmeen Vincit? n.d.)

Vincitin ydinosaamiseen kuuluu ohjelmistokehitys, palvelumuotoilu, pilvialustat, data-analytiikka, IoT ja sulautetut järjestelmät, digitaalisten palveluiden ylläpito ja niiden jatkokehitys, sekä tuotekokemuksen ratkaisut. Vincitin liikevaihto oli vuonna 2020 52,4 miljoonaa euroa, josta liikevoittoa oli 5,96 miljoonaa euroa. Työntekijöitä on yhteensä yli 500. Vincit on myös nimitetty kolmesti Suomen parhaaksi työpaikaksi sekä myös yhdesti Euroopan parhaaksi työpaikaksi. (Mitä me teemme n.d.)

1.2 Toimeksiannon tausta

Vaikkakin elämme vahvasti digitalisoituneessa maailmassa, on monilla yrityksillä edelleen tarvetta tuottaa tuotetietoja hyödyntämällä erilaisia fyysisiä painettuja esitteitä, tuotekuvastoja, katalogeja, yms. sekä myös PDF-tiedostoja moniin eri käyttötarkoituksiin. Näissä on hyvin olennaista, että tuotetieto on validia sekä ajan tasalla. On myös tärkeää, että tuotetiedot ovat helposti saatavilla ja päivitettävissä.

Vincit Jyväskylän projekteissa Akeneo PIM tuotetietohallintajärjestelmä ja Adobe InDesign taittoohjelman lisäosa EasyCatalog on yhdistetty toisiinsa itse kehitetyllä integraatiolla, joka muodostaa tuotetiedoista EasyCatalogille sopivia XML-sanomia. Tätä ideaa haluttiin viedä eteenpäin tuotteistuksen muodossa, jotta Vincitin tekemää konfiguroimista ja kustomointia saataisiin vähennettyä. Tätä varten lähdettiin kehittämään mahdollisimman geneeristä ratkaisua, joka kattaisi mahdollisimman monen asiakkaan tarpeet.

1.3 Työn tavoitteet ja vaatimukset

Työn tavoitteena oli toteuttaa web-pohjainen SaaS-palvelu, joka yhdistää Akeneo PIM tuotetietohallintajärjestelmän Adobe InDesign taitto-ohjelmalle, joka hyödyntää EasyCatalog-lisäosaa. Tavoitteena oli, että käyttäjä voisi palvelun avulla luoda itse erilaisia konfiguraatioita, joissa määritetään mitä tietoja tuotetietohallintajärjestelmästä haetaan. Haetuista tuotetiedoista muodostetaan XML-sanoma, joka on tarkoitus viedä InDesign taitto-ohjelmalle. Toteutuksen on oltava myös mahdollisimman yleiskäyttöinen, jotta sitä voidaan myös kaupata Akeneon kauppapaikalla. Palvelun generoima XML-sanoma ei ole myöskään lukittu pelkästään taitto-ohjelman käyttöön, vaan sitä on mahdollista hyödyntää muihinkin käyttötarkoituksiin.

Palvelu oli tarkoitus totuttaa ns. multi-tenant SaaS-palveluna eli palvelusta on ajossa vain yksi instanssi, joka tukee montaa eri asiakasta. Näin ollen palvelun ylläpito on helpompaa ja halvempaa, kun jokaiselle asiakkaalle ei tarvitse tehdä omaa ajoympäristöä. Vaatimuksena oli myös, että palvelu kehitetään verkkopalveluna, jolloin käyttäjän ei tarvitse asentaa mitään vaan sovellusta voidaan käyttää helposti verkkoselaimen avulla.

2 Tietoperusta

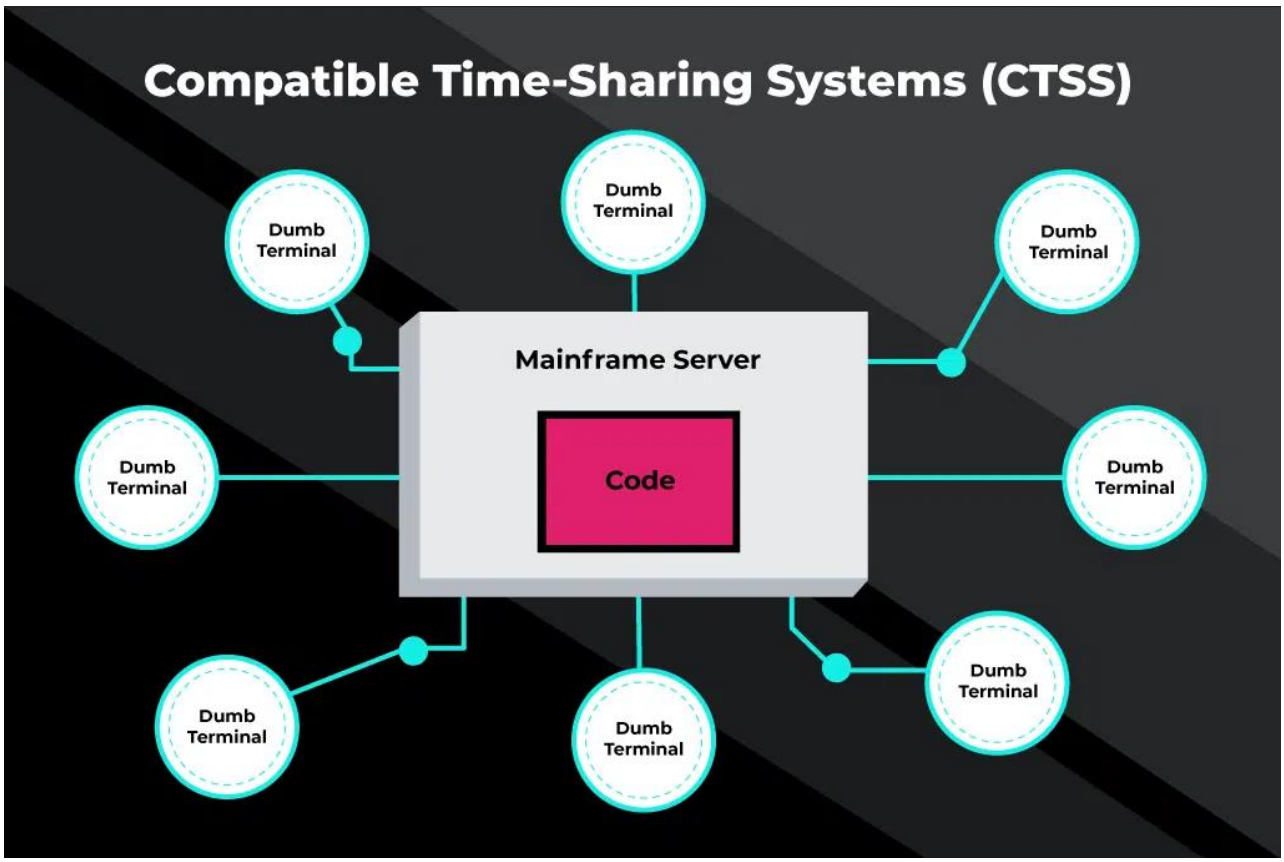
2.1 SaaS-palvelut

SaaS eli Software as a Service on käytännössä pilvipohjainen ohjelmisto, jossa palveluntarjoaja hoitaa ohjelmiston asennuksen, ylläpidon, tietoturvan sekä sen käyttökulut (Short & Spiller 2022.). Näin ollen SaaS-palvelun asiakas pääsee helpommalla, kun vastuu ohjelmiston asennuksesta ja päivittämisestä, ja sen ajoympäristöstä on kokonaan palveluntarjoajalla (Carey 2021.). SaaS-palveluiden hinnoittelu tapahtuu usein joko käytön mukaan tai esimerkiksi kuukausi- tai vuosimaksuilla (Short & Spiller 2022.).

SaaS-palveluita kutsutaan usein harhaanjohtavasti web-sovelluksiksi. Vaikka niillä voi ollakin monessa tapauksessa hyvin samankaltaisia piirteitä, on niillä kuitenkin eroavaisuuksia. Web-sovellus on aina käyttäjän laitteeseen asennetun verkkoselaimen avulla käytettävä sovellus, jolloin Internet yhteys on välttämätön. Kun taas SaaS-sovellus voi olla sen lisäksi myös työpöytäsovellus. (Ghosh 2019.)

Hyvä esimerkki SaaS-palvelusta on Microsoftin Office 365-sovellukset, joita on mahdollista käyttää verkkoselaimen tai käyttäjän koneelle asennettavan työpöytäsovelluksen avulla.

Vaikka SaaS on yleistynyt valtavasti Internetin aikakaudella, ei se kuitenkaan ole uusi asia. 1960-luvulla, kun tietokoneet olivat huoneen kokoisia ja laskentateho kallista, monet teknologiayritykset tarjosivat laskentatehoa sekä tallennustilaa eri organisaatioille käyttäen MITn kehittämää Compatible Time-Sharing System (CTSS)-teknologiaa (ks. Kuvio 1). (Brown 2021.)



Kuvio 1. Compatible Time-Sharing System (CTSS)-teknologia (Brown 2021.)

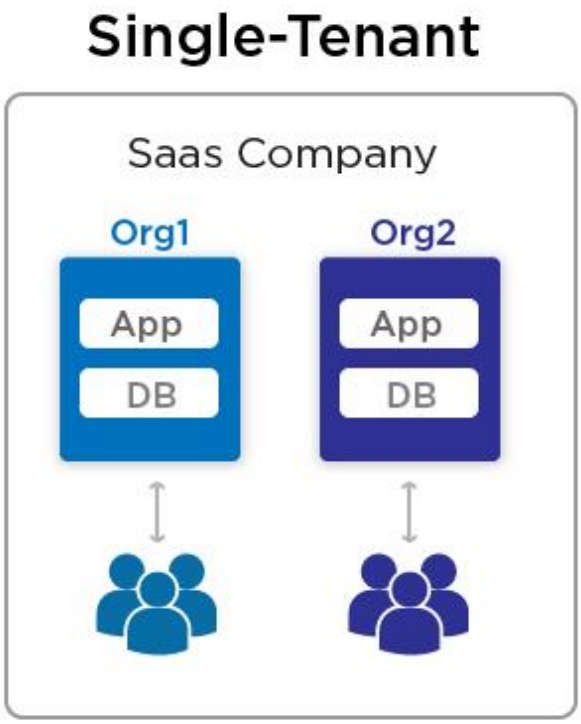
2.1.1 Single- ja multi-tenant SaaS arkkitehtuuri

SaaS-palvelut voidaan karkeasti jakaa kahteen eri malliin: multi-tenant ja single-tenant. Tässä luvussa tarkastellaan mitä ne ovat, mitä eroa niillä on ja mihin tarkoituksiin ne soveltuvat.

Single-tenant SaaS

Single-tenant SaaS tarkoittaa sitä, että palvelusta ajetaan yhtä instanssia asiakasta kohden. Yksi instanssi sisältää ajettavan ohjelmiston lisäksi myös sen ajoon vaaditut riippuvuudet sekä infrastruktuurin, esimerkiksi tietokannan. (Brook 2020.)

Kuviossa 1 on havainnollistettu kuinka kahden eri asiakkaan instanssit ovat täysin eristettyjä toisistaan, eivätkä instanssit jaa keskenään mitään.



Kuvio 2. Single-tenant SaaS arkkitehtuuri (Valdes 2020.)

Single-tenant arkkitehtuurin tärkeimpänä etuna voidaan pitää sen vahvaa tietoturvaa. Tässä arkkitehtuurissa on mahdollista, että palvelua ajetaan täysin omassa fyysisesti eristetyssä ympäristössä ilman mitään yhteyksiä muihin instansseihin, jolloin kaikki data mitä tietokantaan tallennetaan pysyvät eristyksissä muista. (Valdes 2020.)

Muita etuja ovat palvelun helppo kustomointi, eli palvelu voidaan kustomoida täysin asiakkaan tarpeisiin, ja ajoympäristön resurssien käyttö, eli koska instansseja on ajoympäristössä vain yksi, ei sen tarvitse jakaa saatavilla olevia resursseja muiden instanssien kanssa. (Brook 2020.)

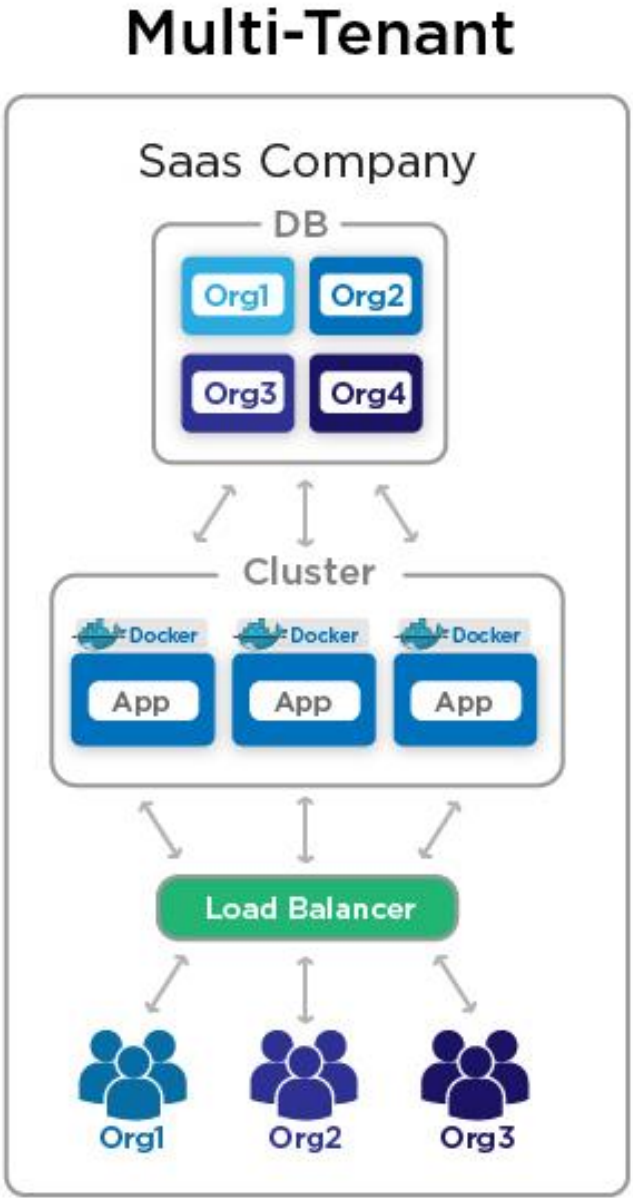
Tämä malli ei kuitenkaan ole täydellinen. Koska single-tenant arkkitehtuurissa jokaiselle asiakkaalle tehdään aina oma instanssi, sen käyttökustannukset ovat usein paljon suuremmat kuin multi-tenant arkkitehtuurissa, lisäksi ohjelmiston ja sen vaatiman infrastruktuurin pystyttämiseen voi kulua paljon aikaa. Lisäksi jos instanssille on allokoitu paljon palvelinresursseja, mutta käyttöä vähän, menevät resurssit tällöin hukkaan. (Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use? 2021.)

Hyvän tietoturvan takia, single-tenant arkkitehtuuri sopii erityisen hyvin tapauksiin, joissa tietoturva ja yksityisyys ovat erittäin tärkeitä. Tällaisia ovat esimerkiksi ohjelmistot, jossa käsitellään rahaliikennettä tai potilastietoja, ja ohjelmistot, jotka vaativan täysin oman eristetyn ajoympäristön muusta syystä. (Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use? 2021.)

Multi-tenant SaaS

Multi-tenant SaaS arkkitehtuurissa ohjelmistosta ajetaan yhtä instanssia, jota käyttää useampi asiakas. Instanssissa asiakkaat käyttävät samaa ohjelmistoa sekä sen ajamiseen vaadittua infrastruktuuria, esimerkiksi tietokantaa, mutta ovat kuitenkin eristetty toisistaan, niin etteivät tiedä toistensa olemassaolosta tai näe toistensa tietoja. (Brook 2020.)

Kuviossa 2 on havainnollistettu esimerkkiratkaisu multi-tenant arkkitehtuurista. Tässä esimerkissä ohjelmistoa ajetaan klusterissa, jonka edessä on kuormantasaaja. Tällainen järjestelmä kykenee käsittelemään suuren määrän verkkoliikennettä samanaikaisesti. Kuviosta näemme myös, että tietokantoja on käytössä vain yksi, johon on luotu kaikille asiakkaille omat skeemat.



Kuvio 3. Multi-tenant SaaS arkkitehtuuri (Valdes 2020.)

Multi-tenant arkkitehtuurin vahvimpana etuna voidaan pitää sen kustannustehokkuutta. Kun palvelua ajetaan vain yhdessä ajoympäristössä, käyttö- ja ylläpitokustannukset ovat huomattavasti pienemmät verrattuna single-tenant arkkitehtuuriin. Muita etuja ovat palvelinresurssien tehokkaampi käyttö, parempi ylläpidettävyys sekä skaalautuvuus usealle käyttäjälle. (Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use? 2021.)

Tässäkin arkkitehtuurissa on omat huonot puolensa. Toisin kuin single-tenant arkkitehtuurissa, jonka vahvuutena voidaan pitää hyvää tietoturvaa, tässä mallissa tietoturva on taas sen heikkous. Esimerkiksi jos kaikkia asiakastietoja säilytetään samassa tietokannassa, on hyvin mahdollista, että tietomurron yhteydessä kaikkien asiakkaiden tiedot ovat vaarantuneet. Tässä mallissa myös asiakaskohtaiset kustomoinnit ja muutokset ovat, ohjelmiston rakenteesta riippuen, paljon vaikeampia toteuttaa, tai ne eivät välttämättä ole edes vaihtoehto. (Brook 2020; Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use? 2021.)

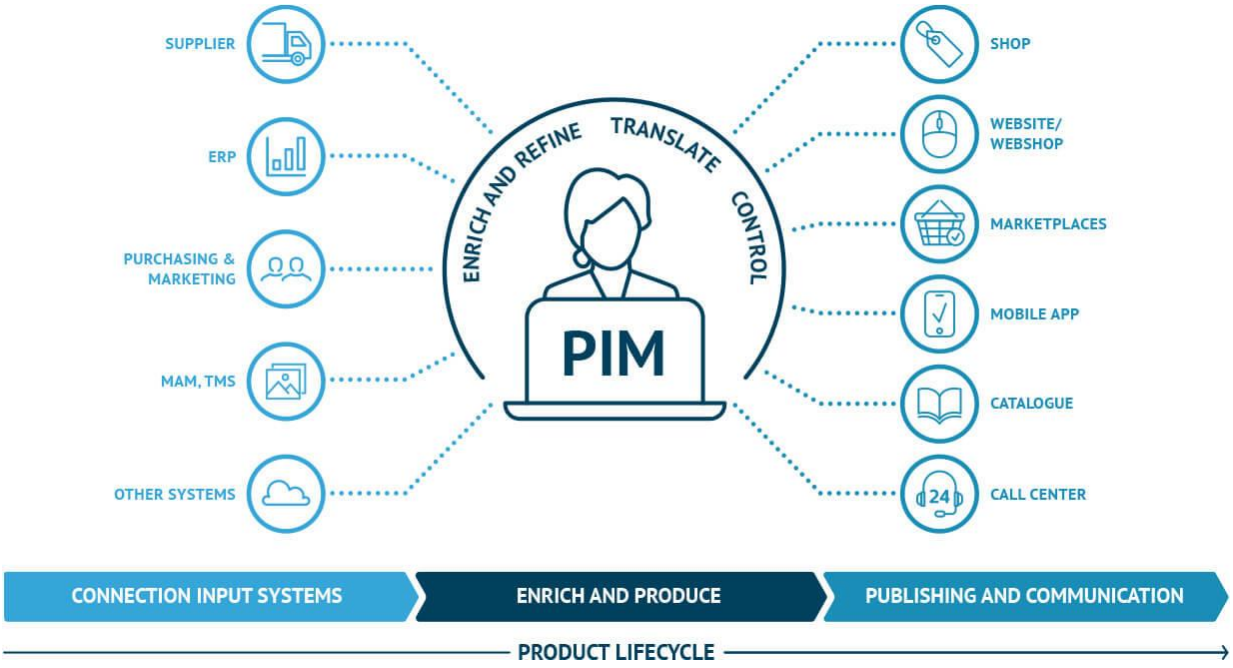
Kustannustehokkuutensa takia, multi-tenant arkkitehtuuri on usein paras valinta, kun lähdetään toteuttamaan SaaS-palvelua. Suurin osa kuluttajille suunnatuista SaaS-sovelluksista on rakennettu multi-tenant mallilla, niin kuin myös tässä työssä tehdään. Multi-tenant malli soveltuu siis käytännössä mihin vaan käyttötarkoitukseen, jossa asiakaskohtaisten tietojen eristäminen ei ole kriittistä. (Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use? 2021.)

2.2 Tuotetiedonhallinta (PIM)

Tuotetiedonhallinta eli PIM (engl. Product Information Management) tarkoittaa tuotetietojen keskittämis-, ja hallinnointiprosessia (What is PIM? n.d.). Yleensä kun puhutaan tuotetiedonhallinnasta, tarkoitetaan sillä tuotetiedonhallintajärjestelmää, joka keskittää yhteen kaiken yrityksen ulkopuolelle tarjottavan tuotetiedon (The Importance of the Product Information Management (PIM) 2019.). Tällaisia tietoja ovat muun muassa nimet, tuotekuvaukset, tuotekuvat ja medialiitteet, hinnat, tekniset tiedot ja käyttöohjeet.

Tuotetiedonhallintajärjestelmät mahdollistavat korkealaatuisen tuotetiedon tarjoamisen suoraan markkinointi- ja myyntialustoille, kuten verkkokauppoihin ja tuotekuvastoihin, joissa on tärkeää, että tuotetieto on validia ja korkealaatuista. Tuotetiedonhallintajärjestelmä nopeuttaa tuotteiden vientiä verkkokauppa-alustoille, jolloin myös myynti kasvaa. Myyntiä lisää myös korkealaatuinen tuotetieto, joka parantaa asiakaskokemusta sekä luottamusta, jolloin asiakas todennäköisemmin ostaa yritykseltä myös jatkossa. (Miksi tuotetiedonhallinta tekee liiketoiminnasta kannattavampaa 2018.)

Kuviossa 4 on visualisoitu tuotteen elinkaarta, josta näemme, kuinka tuotetiedonhallintajärjestelmä on tuotteen elinkaaren keskiössä.

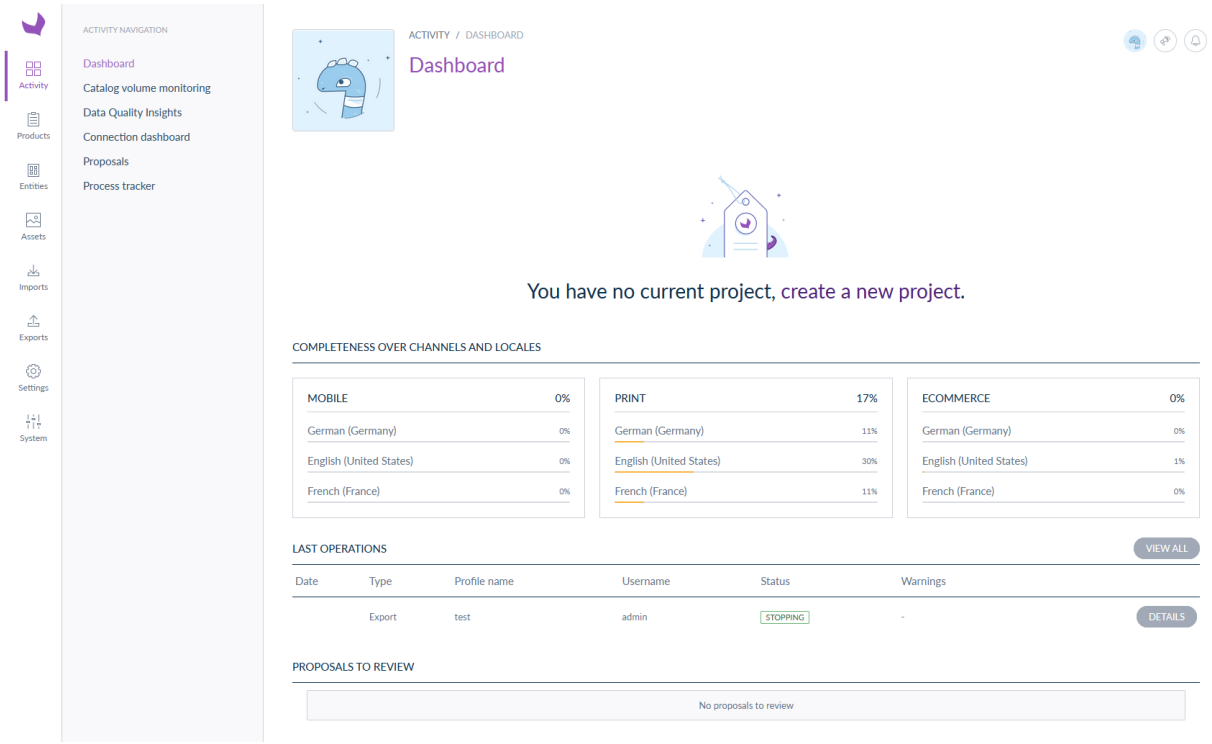


Kuvio 4. PIM-järjestelmä tuotteen elinkaaressa (Definition – What does Product Information Management mean? N.d.)

2.2.1 Akeneo PIM

Akeneo PIM on Ranskalaisen Akeneon kehittämä tuotetietohallintajärjestelmä. Akeneo PIM järjestelmästä on olemassa maksullinen Enterprise Edition sekä ominaisuuksiltaan rajallisempi avoimeen lähdekoodiin perustuva Community Edition. Avoimuutensa ansiosta Akeneo PIM on maailman eniten käytetyin PIM-järjestelmä ja vuonna 2019 järjestelmä oli käytössä yli 60 000 yrityksessä. (Dillet 2019.)

Akeneo PIM on ulkoasultaan sekä käytettävyydeltään yksinkertainen, joka nopeuttaa järjestelmän haltuunottoa. Kuviossa 5 on esitelty Akeneo PIM järjestelmän käyttöliittymää.



Kuvio 5. Akeneo PIM järjestelmän aloutusnäkymä

3 Suunnittelu

Tässä luvussa tutustutaan Akeneo EasyCatalog InDesign Connectorin suunnitteluprosessiin sekä käytettyihin menetelmiin. Luvussa käsitellään aluksi yleistä suunnitteluprosessia, jonka jälkeen syvennytään sovelluksen arkkitehtuuriin ja lopuksi tarkastellaan valittuja teknologioita sekä tietokannan rakennetta.

3.1 Yleistä

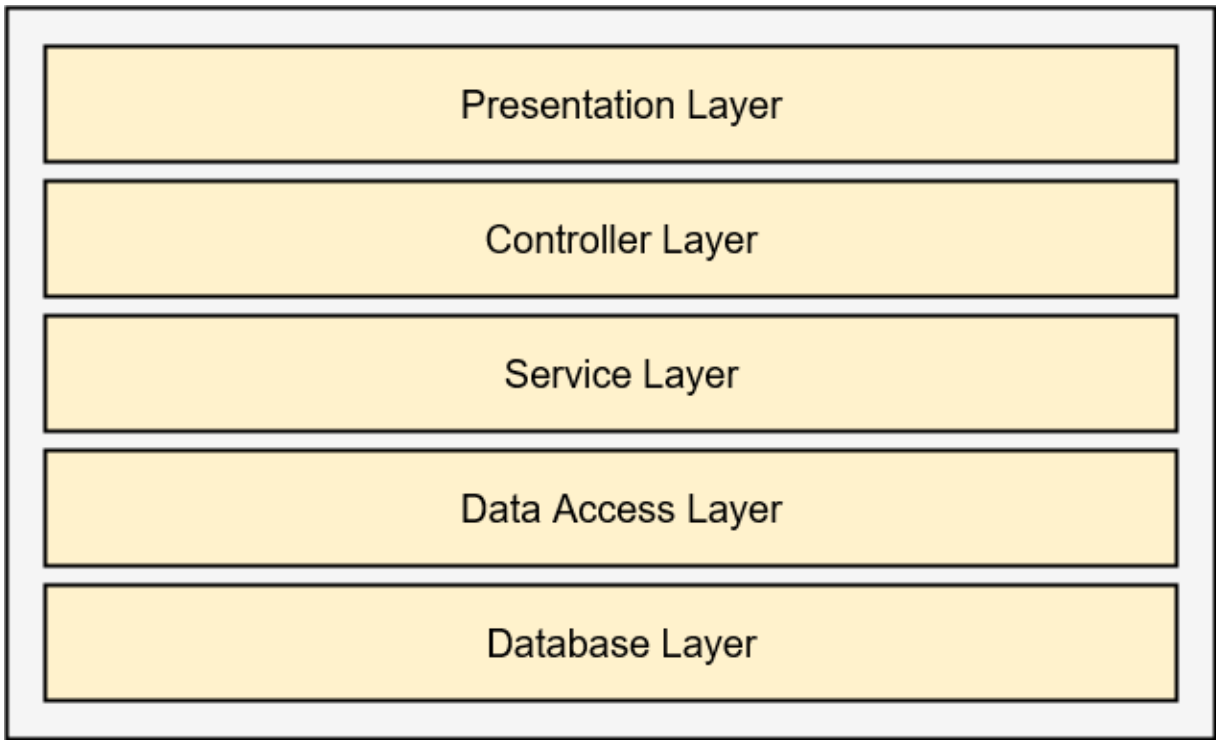
Ennen kuin varsinaista kehitystyötä voidaan lähteä aloittamaan, on työn onnistumisen kannalta olennaista, että työ suunnitellaan ensin perusteellisesti. Työn suunnittelu lähti liikkeelle sovelluksen toiminnallisuuksien ja tietomallin määrityksistä. Tätä vaihetta helpotti huomattavasti jo olemassa olevat saman tyyppiset sovellukset, joidenka rakennetta tutkimalla sai perustavanlaatuisen kuvan, mitä kaikkea ottaa huomioon niin suunnittelu kuin totutusvaiheessa. Suunnittelua helpotti myös toteutuksesta olevat alustavat määritykset, joidenka pohjalle oli helppo lähteä rajaamaan sovelluksen ensimmäisen iteraation toteutusta.

3.2 Arkkitehtuuri

3.2.1 Yleistä

Richardsin (2015, 6) mukaan ohjelmiston arkkitehtuurin suunnittelussa on tärkeää ottaa huomioon sen skaalautuvuus, suorituskyky, kyky sopeutua tarpeen muuttuessa ja minkälaisia rajoitteita sen kehittämiseen ja tuotantoon vientiin liittyy. Richards (2015, 6) myös painottaa, että ilman selkeää ja hyvin perusteltua arkkitehtuuria monet sovellukset päätyvät hauraiksi ja vaikeasti ylläpidettäviksi kokonaisuuksiksi ilman selvää päämäärää tai tavoitetta. Nämä tiedot antoivat hyvän kuvan siitä, mitä ottaa huomioon, kun halutaan onnistua ohjelmistoarkkitehtuurin valinnassa.

Ottaen huomioon sovelluksen vaatimukset, laajuuden sekä mahdollisten käyttäjien määrän samanaikaisesti, sovelluksen arkkitehtuuriksi valittiin kerrosarkkitehtuuri, jossa tulee olemaan yhteensä viisi kerrosta kuvion 6 mukaisesti. Valintaan vaikutti myös se, että kerrosarkkitehtuuri on hyvin yksinkertainen ja sen kanssa on hyvin helppo ja nopea päästä alkuun. Se on myös hyvin yleisesti käytetty malli, joten siihen löytyy lukuisia eri esimerkkejä, joita voi käyttää hyödyksi kehitystyössä.



Kuvio 6. Sovelluksessa käytetty viisikerroksinen kerrosarkkitehtuuri

3.2.2 Kerrosarkkitehtuuri

Tässä luvussa tarkastellaan sovelluksessa käytettävän kerrosarkkitehtuurin eri kerroksia.

Käyttöliittymäkerros (Presentation layer)

Käyttöliittymäkerros tarkoittaa sovelluksen selainpuolta eli sovelluksen osaa, joka näkyy loppukäyttäjälle verkkoselaimessa. Käyttöliittymäkerros koostuu enimmäkseen HTML-merkintäkielestä, CSS-tyylimäärityksistä sekä JavaScript-koodista, jolla saadaan luotua käyttöliittymälle jonkinlaista sovelluslogiikkaa. Käyttöliittymäkerros ei tule sisältämään paljoa sovelluslogiikkaa, vaan se lähettää pyynnön ohjauskerrokselle HTTP-protokollaa käyttäen.

Ohjauskerros (Controller layer)

Ohjauskerroksen vastuulla on kaikkien HTTP-pyyntöjen vastaanottaminen, niiden esikäsittely ja niihin vastaaminen. Tässä kerroksessa määritetään sovelluksen palvelinpuolen (engl. backend) päätepisteet (engl. endpoint) sekä niiden väliohjelmistot (engl. middleware). Väliohjelmistot esikäsittelevät HTTP-pyynnön ennen kuin se ohjataan seuraavalle kerrokselle eli palvelukerrokselle. Väliohjelmisto voisi olla esimerkiksi HTTP-pyynnön autorisointiohjelmisto, joka varmistaa, että pyynnön lähettäjällä on oikeudet johonkin resurssiin.

Palvelukerros (Service layer)

Palvelukerros tai logiikkakerros on kerros, jossa suurin osa sovelluksen palvelinpuolen logiikasta sijaitsee. Tässä kerroksessa käsitellään kaikki sovelluksen kriittiset toimenpiteet, kuten esimerkiksi uusien käyttäjien luonti ja niiden autentikoiminen, sekä tietojen hakemisen tietokannasta. Palvelukerros ei kuitenkaan hae tietoa suoraan tietokannasta vaan käyttää datankäsittelykerrosta tietokanta operaatioihin.

Datankäsittelykerros (Data Access layer) ja Tietokantakerros (Database layer)

Datankäsittelykerros on kerros, jonka tehtävä on tarjota palvelukerrokselle kaikki sen vaatimat tietokantaoperaatiot. Näitä ovat muun muassa tietojen hakeminen, lisääminen, muokkaaminen ja poistaminen. Tietokantakerros taas koostuu nimensä mukaan pelkästä tietokannasta ja sen tarjoamista toiminnallisuuksista.

3.3 Teknologioiden valinta

Koska työ oli tarkoitus toteuttaa web-pohjaisena SaaS-palveluna, oli tähän tarkoitukseen soveltuvia teknologioita ja tekniikoita olemassa paljonkin erilaisia. Valintoja tehdessä oli olennaista ottaa huomioon jo olemassa oleva osaaminen eri teknologioista sekä toimeksiantaja yrityksessä yleisesti käytetyt teknologiat, jolloin kehitystyössä pääsisi nopeasti alkuun ja tulevaisuudessa mahdollinen sovelluksen jatkokehitys jonkun muun toimesta olisi helpompaa.

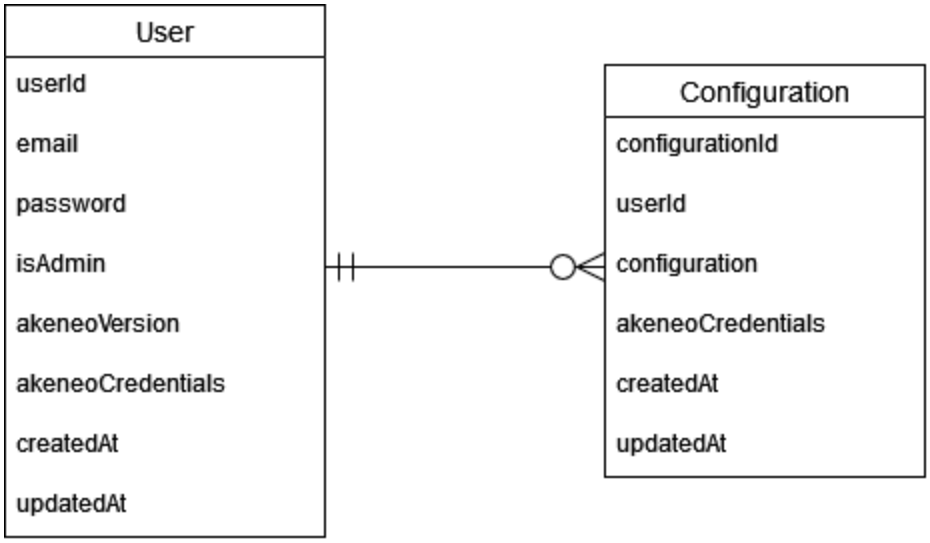
Näin ollen valinnoissa päädyttiin käyttämään React.js -kirjastoa sovelluksen selainpuolella. React.js on hyvin suosittu JavaScript-kirjasto erilaisten käyttöliittymien tekemiseen. Palvelinpuoli taas toteutettiin Node.js ajoalustaa sekä Express.js kirjastoa hyödyntämällä. Node.js on JavaScript ajoalusta, joka mahdollistaa JavaScript koodin ajamisen myös palvelinpuolella (About Node.js n.d.). Express.js on hyvin minimalistinen reititys- sekä väliohjelmistokirjasto Node.js alustalle, joka tarjoaa hyvät työkalut erilaisten rajapintojen toteutukseen (Express Fast, unopinionated, minimalist web framework for Node.js n.d.). Tietokannaksi valittiin perinteinen relaatiotietokanta, joka oli tässä tapauksessa PostgreSQL.

Sekä käyttöliittymä että palvelinpuoli toteutettiin TypeScript-kielellä, jolloin niiden välillä työskentely onnistui vaivattomasti. TypeScript itsessään ei ole ohjelmointikieli, vaan se tuo lisäominaisuuksia JavaScript-kieleen, kuten vahvan tyypityksen ja selkeämmän syntaksin.

Valittuihin teknologioihin päädyttiin juuri edellä mainituista syistä, eli niistä oli valmiiksi jo hyvin kokemusta sekä ne olivat myös yleisesti käytössä toimeksiantajalla. Etuna voidaan pitää myös sitä, että valitut teknologiat olivat myös hyvin suosittuja, jolloin erilaisia ohjeita, esimerkkejä ja hyviä käytänteitä oli saatavilla hyvinkin laajasti. Teknologiavalinnat olivat jo heti alussa hyvin selviä, eikä siihen mennyt paljoakaan aikaa.

3.4 Tietokannan suunnittelu

Koska sovelluksessa tultaisiin käyttämään relaatiotietokantaa, vaatisi se jonkinlaisen alustavan ER-kaavion suunnittelun, jotta sovelluksen toteutus olisi selkeämpää. Kuviossa 7 on kuvattu sovelluksen tietokannan rakennetta ER-kaavion avulla. Tietokannassa on kaksi taulua. Käyttäjä- ja konfiguraatiotaulu. Yhdellä käyttäjällä voi olla useita konfiguraatioita, mutta yksi konfiguraatio voi kuulua vain yhdelle käyttäjälle. Jos käyttäjä poistetaan, poistuu samalla myös kaikki käyttäjän luomat konfiguraatiot.



Kuvio 7. Tietokannan ER-kaavio

4 Tekninen toteutus

Tässä luvussa syvennytään Akeneo EasyCatalog InDesign Connectorin toteutukseen ja sen aikana vastaan tulleisiin haasteisiin ja ratkaisuihin. Luvussa tutustutaan aluksi itse sovelluksen kehitystyöhön, jonka jälkeen tarkastellaan sovelluksen ajoympäristön infrastruktuuria.

4.1 Yleistä

Toteutuksen tavoitteena oli luoda sovellus, jossa käyttäjä voisi luoda erilaisia konfiguraatioita käyttöliittymän avulla. Luotua konfiguraatiota käytetään sen jälkeen suodattimena tuotetietojen hakemiseen, jonka jälkeen haetuista tuotetiedoista muodostetaan XML-sanoma. Konfiguraatio on käytännössä kuvion 8 mukainen JSON-objekti, joka koostuu erilaisista arvoista, joita käytetään

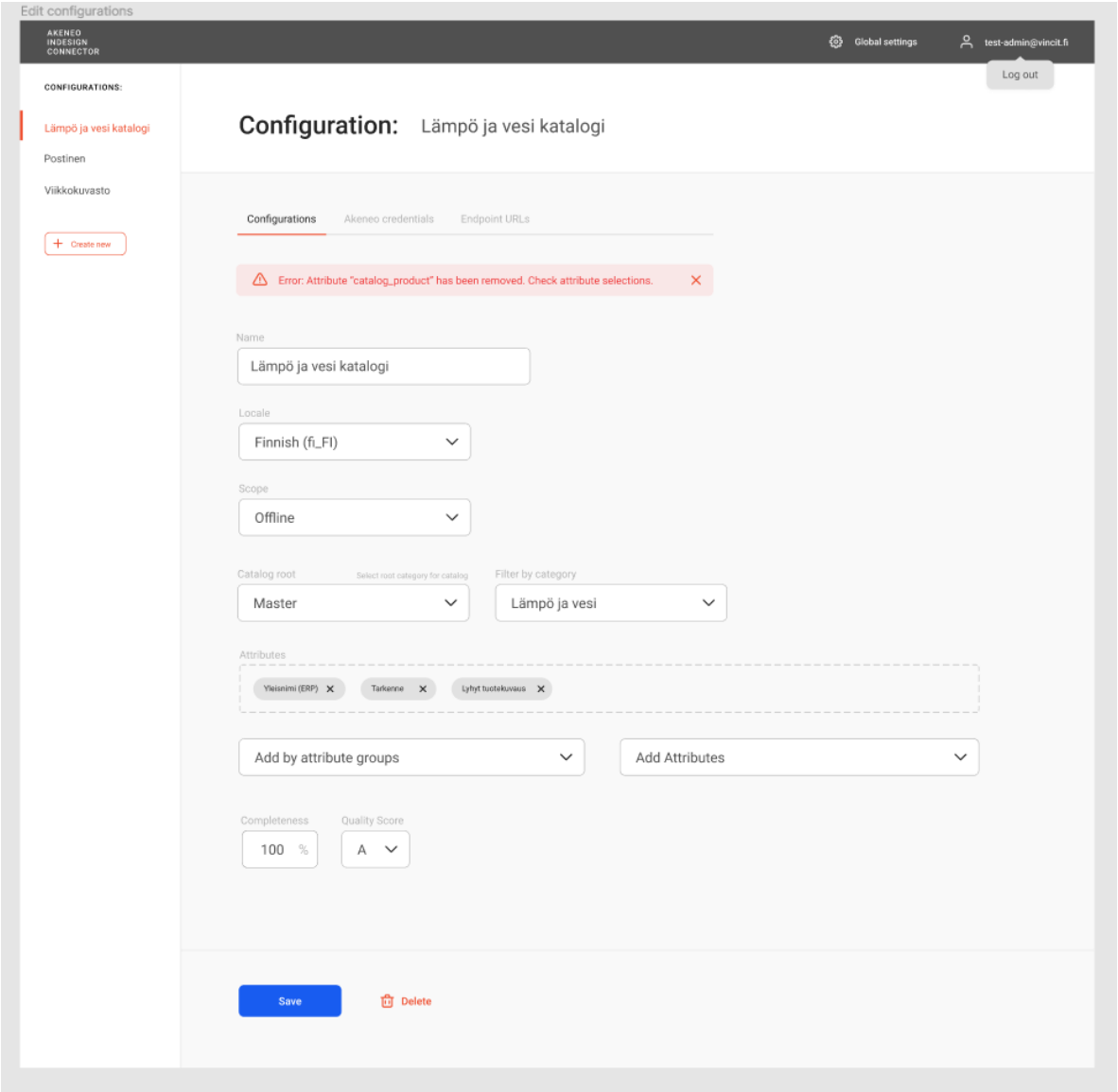
tuotetietojen suodattamiseen. Näitä arvoja ovat tuotteen kanava, kieli, kategoria, attribuutit, kategoriapuu, valmiusaste prosentti sekä laatupisteytys.

Kuvio 8. Esimerkki konfiguraatio JSON muodossa

Toteutuksessa oli käytännössä kolme osaa. Sovelluksen käyttöliittymä (engl. frontend), palvelinpuoli (engl. backend) sekä infrastruktuuri, joka vaaditaan sovelluksen ajamiseen. Toteutuksen alussa projektille luotiin uusi GitHub-repositorio, jossa lähdekoodia sekä dokumentaatiota tultaisiin ylläpitämään. Koodieditorina oli käytössä Visual Studio Code, joka soveltui erittäin hyvin TypeScript kehitykseen. Toteutusta varten asennettiin paikallinen Akeneo PIM järjestelmä, jotta sovelluksen testaus olisi helpompaa.

4.2 Sovelluksen tekninen toteutus

Sovelluksen toteutus alkoi alustavan käyttöliittymän toteuttamisella. Tätä vaihetta helpotti huomattavasti jo olemassa oleva käyttöliittymä prototyyppi, joka oli tehty Figma-suunnittelutyökalulla (ks. Kuvio 9). Prototyypin ansiosta erilaiset käyttöliittymäelementit oli paljon helpompi toteuttaa, kuin jos olisi lähtenyt kehittämään käyttöliittymää tyhjästä.



Kuvio 9. Sovelluksen käyttöliittymän Figma-prototyyppi

Sovellukseen luotiin aluksi kirjautumisnäkymä sekä siihen tarvittava palvelinpuolen logiikka. Näin ollen muut ominaisuudet oli helpompi toteuttaa sen päälle. Kirjautuminen toteutettiin Json-WebTokeneita (JWT) hyödyntämällä, eli kun käyttäjä kirjautuu sisään, sovelluksen palvelinpuoli palauttaa selaimelle JsonWebTokenin, joka sitten tallennetaan selaimen muistiin. JWT on voimassa 4 tuntia, jonka jälkeen käyttäjän täytyy kirjautua sisään uudestaan. Jos käyttäjä kirjautuu ulos sovelluksesta, JWT poistetaan. JsonWebToken lähetetään aina HTTP-pyynnön mukana sovelluksen palvelinpuolelle, joka varmistaa, että JWT on validi sekä ajan tasalla. Jos JWT on vanhentunut tai se ei ole validi, palautetaan käyttöliittymälle 401 HTTP-virheviesti, joka kertoo käyttäjälle

epäonnistuneesta validoinnista. Käyttäjä kirjataan sitten ulos ja vanha JWT poistetaan selaimen muistista. JWT kertoo myös käyttäjän roolin, eli onko käyttäjä perus- vai ylläpitokäyttäjä.

Sovellukseen kirjaudutaan sähköpostin ja salasanan avulla. Uusia käyttäjiä ei voi luoda kuka vain, vaan ainoastaan sovelluksen ylläpitokäyttäjät voivat luoda uusia käyttäjiä. Ylläpitokäyttäjät pystyvät myös muokkaamaan tai poistamaan olemassa olevia käyttäjiä sovelluksen hallintapaneelin avulla. Tämä on tarpeen, mikäli jonkun käyttäjän salasana pitää vaihtaa tai vanha käyttäjä halutaan poistaa. Kun sovellus asennetaan ajoympäristöön ja käynnistetään ensimmäisen kerran, järjestelmä luo automaattisesti uuden ylläpitokäyttäjän järjestelmään, jos sitä ei ole vielä olemassa. Käyttäjien salasanoja ei tallenneta tietokantaan sellaisenaan, vaan niistä tehdään ensin tiiviste käyttäen argon2-hajautusalgoritmia, jonka jälkeen tiiviste tallennetaan tietokantaan. Tietokantaan tallennettua tiivistettä ei ole tarkoitus purkaa, vaan käyttäjän kirjautuessa tiivistetään käyttäjän antama salasana uudestaan ja verrataan sitä tietokannassa olevaan tiivisteeseen.

Tämän jälkeen luotiin konfiguraatiolomake, jonka avulla uusia konfiguraatioita voitaisiin luoda sekä vanhoja muokata. Tätä varten tehtiin yksi geneerinen lomakekomponentti, jota oli mahdollista käyttää niin konfiguraation luomiseen kuin niiden muokkaamiseen. Lomakkeeseen tarvittiin eri tietoja Akeneo PIM järjestelmän rajapinnasta, jota varten palvelinpuolen rajapintaan luotiin options-niminen resurssi, jota hyödyntämällä käyttöliittymä sai haettua kaikki tarvittavat tiedot.

Jotta sovellus saisi yhteyden Akeneo PIM järjestelmään, täytyi käyttäjän ensin luoda uudet Akeneo PIM tunnukset ja sen jälkeen asettaa ne sovelluksen asetuksissa kohdilleen. Koska Akeneo PIM järjestelmän tunnukset ovat arkaluonteista tietoa, täytyy ne ensin salata ennen niiden tallentamista tietokantaan. Tässä tapauksessa salausmenetelmänä käytettiin AES-256-salausta, joka todettiin riittävän vahvaksi. Salaukseen tarvittava salausavain tarjottiin sovellukselle ympäristömuuttujien avulla, jolloin se olisi helposti vaihdettavissa, jos tilanne sen vaatii.

Toteutuksen yksi työläimmistä osuuksista oli JSON-muotoisten tuotetietojen muuntaminen XML-muotoon. Akeneo PIM järjestelmän rajapinnasta saatava tuotetieto tulee JSON-muodossa, jossa yksittäisen tuotteen attribuuttien tietorakenne vaihteli aina attribuuttityypin mukaan, jolloin eri tietorakenteille tarvittiin aina erilainen ohjelmallinen käsittely. Esimerkiksi tekstityyppiselle attri-

buutille riitti hyvin, että siitä tehdään yksi XML-elementti, jonka arvoksi asetetaan kyseisen attribuutin arvo. Mutta jos attribuutti olisi esimerkiksi monivalintatyyppinen, olisi sen tietorakenne taulukkomuotoinen, jolloin se vaatisi hyvinkin erilaisen käsittelyn.

Attribuuttityyppi oli kuitenkin mahdollista saada selville Akeneo PIM-järjestelmän rajapinnasta, joka helpotti attribuuttien käsittelyä huomattavasti. Tämä ratkaisu kuitenkin hidasti merkittävästi XML-sanoman generoimista, koska jos tuotteita käsitellään monia tuhansia ja jokaisella tuotteella on esimerkiksi kymmenen attribuuttia, joudutaan rajapinnasta hakemaan jokainen attribuutti aina uudestaan. Tätä prosessia saatiin optimoitua merkittävästi niin, että rajapinnasta haetut attribuutit tallennetaan välimuistiin, jolloin niitä ei tarvitse hakea Akeneon rajapinnasta kuin kerran sanoman generoinnin aikana.

Myöhemmissä testeissä huomattiin, että erittäin suurilla tuotemäärillä XML-sanoman generoiminen vei niin paljon palvelinresursseja, että koko sovelluksen palvelinpuoli kaatui resurssien puutteen takia. Tähän ongelmaan kehitettiin tehokas ratkaisu, jossa XML-luotiin pienissä erissä ja palautettiin käyttäjälle datavirtaa (engl. data stream) hyödyntämällä. Tämän ansiosta sovellus ei vaatinut palvelimelta kovinkaan paljoa resursseja.

4.3 Sovelluksen infrastruktuuri

Sovelluksen ensimmäisen version valmistuttua, oli sovellus sen jälkeen tarkoitus asentaa testiympäristöön testausta varten. Testiympäristön tulisi olla samanlainen kuin mitä tuotantoympäristö tulisi olemaan, jotta mahdollisimman moni ongelma saataisiin ratkaistua ennen tuotantoon vientiä. Sovelluksen ajoympäristö tulisi käytännössä olemaan virtuaalikone, jossa sovellusta ajetaan konteissa Dockerin avulla. Jokaista sovelluksen osaa ajetaan omassa Docker kontissaan ja niiden hallintaan käytetään Docker Compose-työkalua, jolla on mahdollista luoda ja hallita useasta Docker kontista muodostuvia sovelluksia helposti ja tehokkaasti.

Kaikki verkkoliikenne kulkee NGINX-ohjelmalla toteutetun käänteisen välityspalvelimen (engl. reverse proxy) läpi, jolloin kontit eivät ole suoraan näkyvissä ulkoverkkoon eikä palomuurista tarvitse avata kuin välityspalvelimen käyttämät portit. Koska kaikki verkkoliikenne kulkee käänteisen välityspalvelimen läpi, mahdollistaa se myös paremman lokituksen. Välityspalvelimen lokitiedostoista

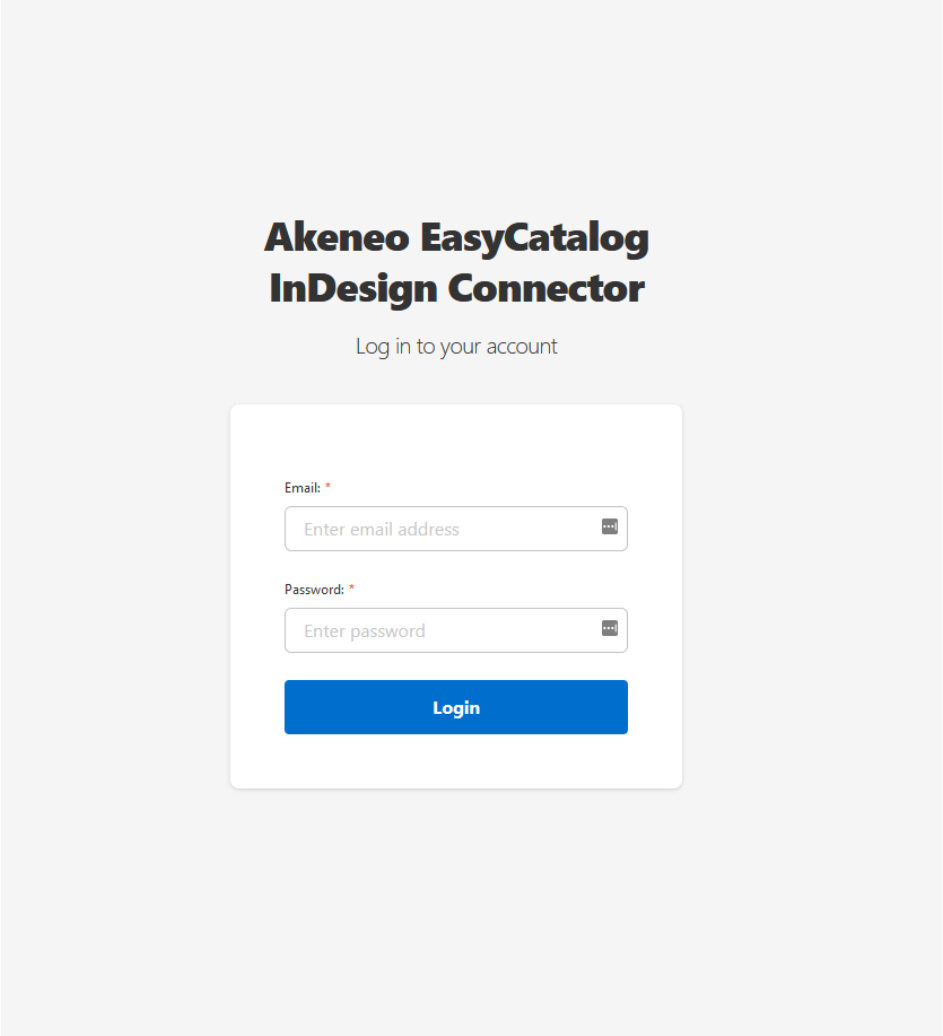
oli mahdollista nähdä muun muassa mistä IP-osoitteesta sovellukseen yhdistettiin. NGINX käänteisen välityspalvelimen avulla sovellukselle saatiin myös asennettua HTTPS-sertifikaatti Let's Encryptin ilmaista Certbot-ohjelmaa hyödyntämällä. HTTPS-sertifikaatti mahdollistaa kaiken verkkoliikenteen salaamisen, jolloin sovelluksen tietoturva paranee ja palvelusta saa myös luotettavamman kuvan loppukäyttäjän näkökulmasta.

Sovelluksen pilvialustaksi valittiin Google Cloud Platform-pilvipalvelu. Tähän päädyttiin, koska sen käytöstä oli jo valmiiksi paljon kokemusta sekä sitä myös käytettiin yleisesti toimeksiantaja yrityksessä. Google Cloudiin luotiin uusi virtuaalikone, jolle määritettiin staattinen IP-osoite. Ilman staattista IP-osoitetta, virtuaalikoneen IP-osoite voi muuttua, jolloin palvelun domain-nimi ja HTTPS-sertifikaatti eivät enää toimisi. Virtuaalikoneelle tarvittiin asentaa vain Docker ja Docker Compose, jonka jälkeen sovellus oli jo mahdollista laittaa ajoon.

5 Tulokset

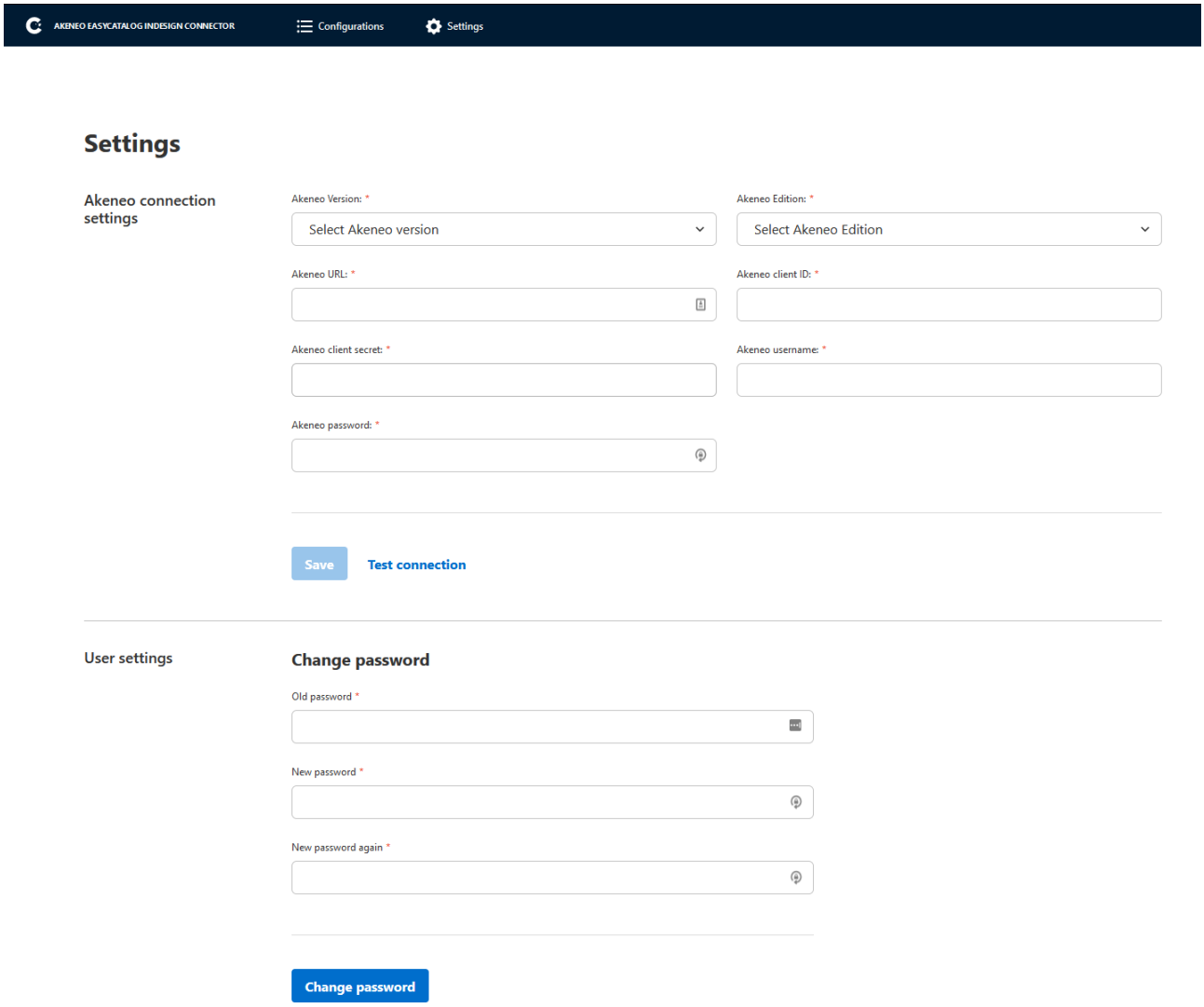
Tässä luvussa tarkastellaan kehitystyöstä syntyneitä tuloksia sekä tutustutaan tarkemmin Akeneo EasyCatalog InDesign Connectorin käyttöön.

Sovelluksen ensimmäinen näkymä on sen kirjautumisnäkymä (ks. Kuvio 10). Jotta sovellusta voidaan käyttää, on käyttäjän ensin kirjauduttava sisään. Sovellukseen kirjaudutaan sähköpostilla ja salasanalla. Käyttäjä ei voi luoda itselleen tunnuksia, vaan ylläpitokäyttäjän täytyy ensin luoda ne valmiiksi.



Kuvio 10. Akeneo EasyCatalog InDesign Connectorin kirjautumisnäkymä

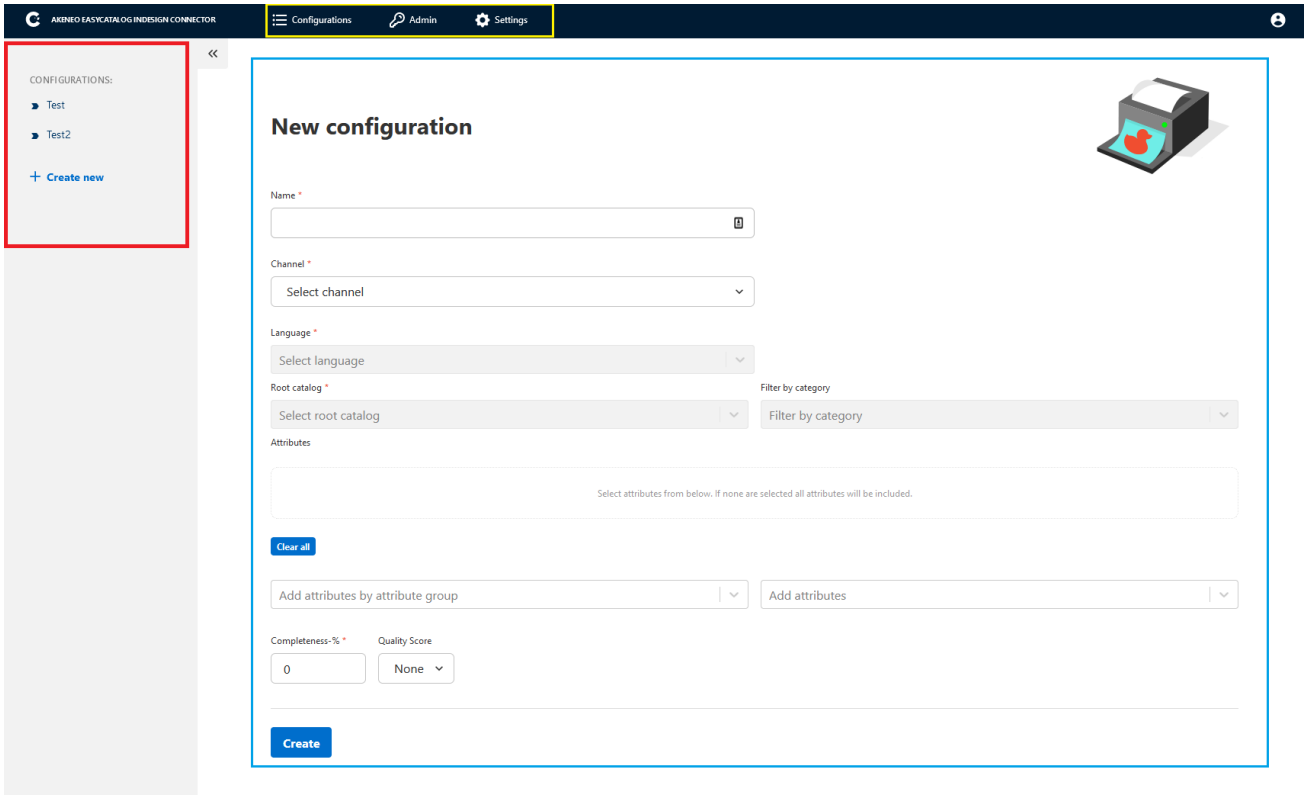
Ensimmäisen kirjautumisen yhteydessä käyttäjälle avautuu asetukset näkymä (ks. Kuvio 11), johon käyttäjän täytyy syöttää Akeneo PIM järjestelmän tunnukset. Tunnusten avulla sovellus pystyy hakemaan tarvittavat tiedot Akeneo PIM järjestelmän rajapinnasta. Ilman tunnuksia sovellusta ei ole mahdollista käyttää. Sovellukselle täytyy myös kertoa mikä versio Akeneo PIM järjestelmästä on käytössä. Asetuksista käyttäjä voi myös vaihtaa salasanansa.



Kuvio 11. Asetukset näkymä ensimmäisen kirjautumisen yhteydessä

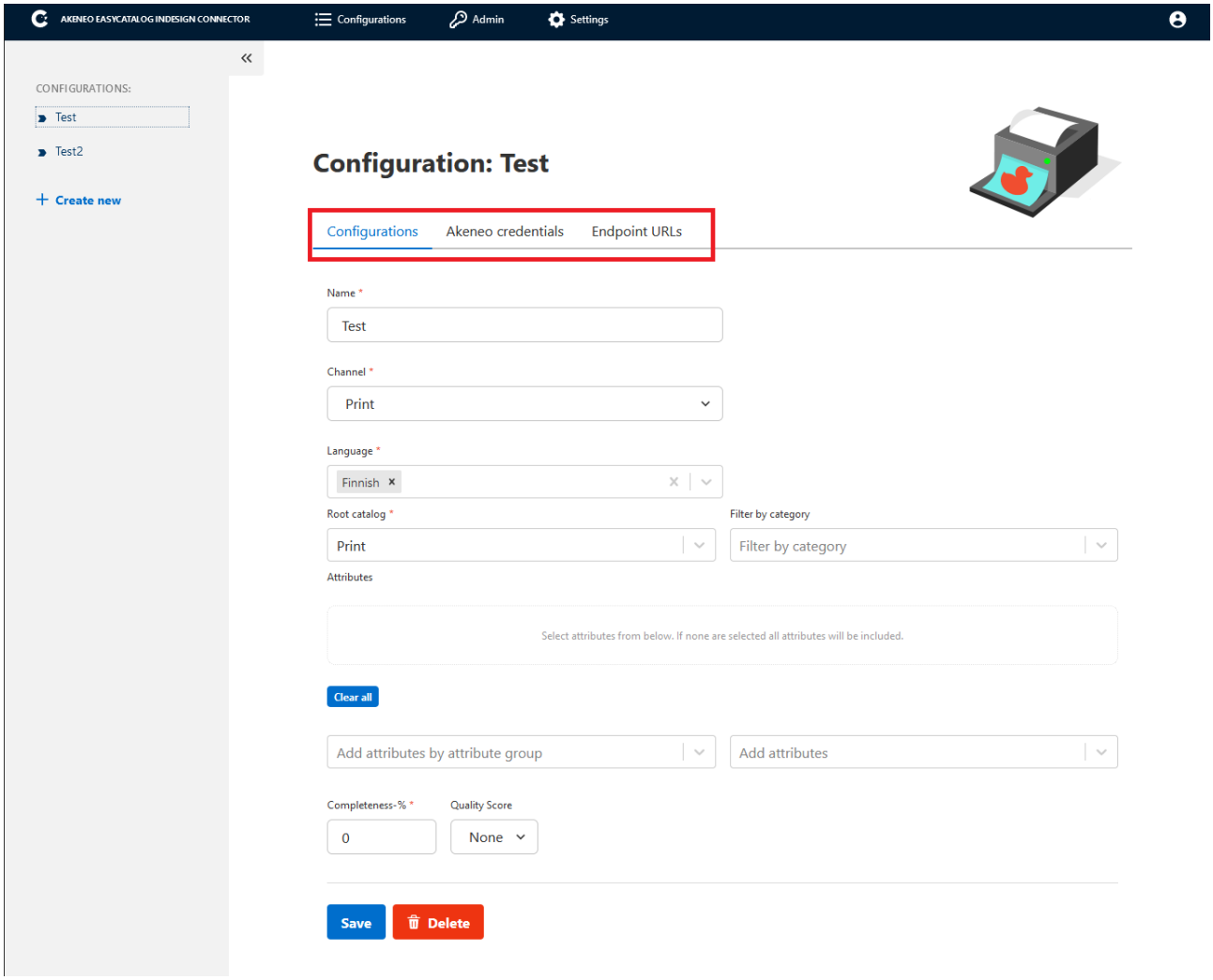
Tunnusten asettamisen jälkeen sovellus on käyttövalmiina ja käyttäjälle avautuu konfiguraationäkymä, jossa käyttäjä voi lisätä tai muokata konfiguraatioita (ks. Kuvio 12). Kuviossa 12 punaisella värillä on korostettu lista kaikista käyttäjän luomista konfiguraatioista, joita klikkaamalla käyttäjä voi joko muokata tai poistaa valitun konfiguraation. Keltaisella värillä on korostettu navigointipalkin eri painikkeet, joita klikkaamalla käyttäjä voi siirtyä eri näkymien välillä. Sovelluksen eri näkymiä ovat konfiguraationäkymä, asetukset sekä vain ylläpitokäyttäjille näkyvä hallintapaneeli. Sinisellä värillä on korostettu uusien konfiguraatioiden luontiin käytettävä lomake. Jotta konfiguraation saa luotua, täytyy sille määrittää ainakin nimi, kanava, kieli sekä kategoriapuu. Loput vaihtoehdot ovat valinnaisia.

Kuviosta 12 myös huomaamme, että konfiguraatiota luodessa jotkut vaihtoehdot eivät ole valittavissa ennen kuin käyttäjä on valinnut edellisestä valikosta jonkun vaihtoehdon. Esimerkiksi kieltä tai kategoriapuuta ei voi valita ennen kuin käyttäjä on valinnut kanavan. Tämä johtuu siitä, että Akeneo PIM järjestelmässä eri kanavilla on aktivoituna eri kielet, jolloin kieli voidaan valita vasta kun kanava on valittu.



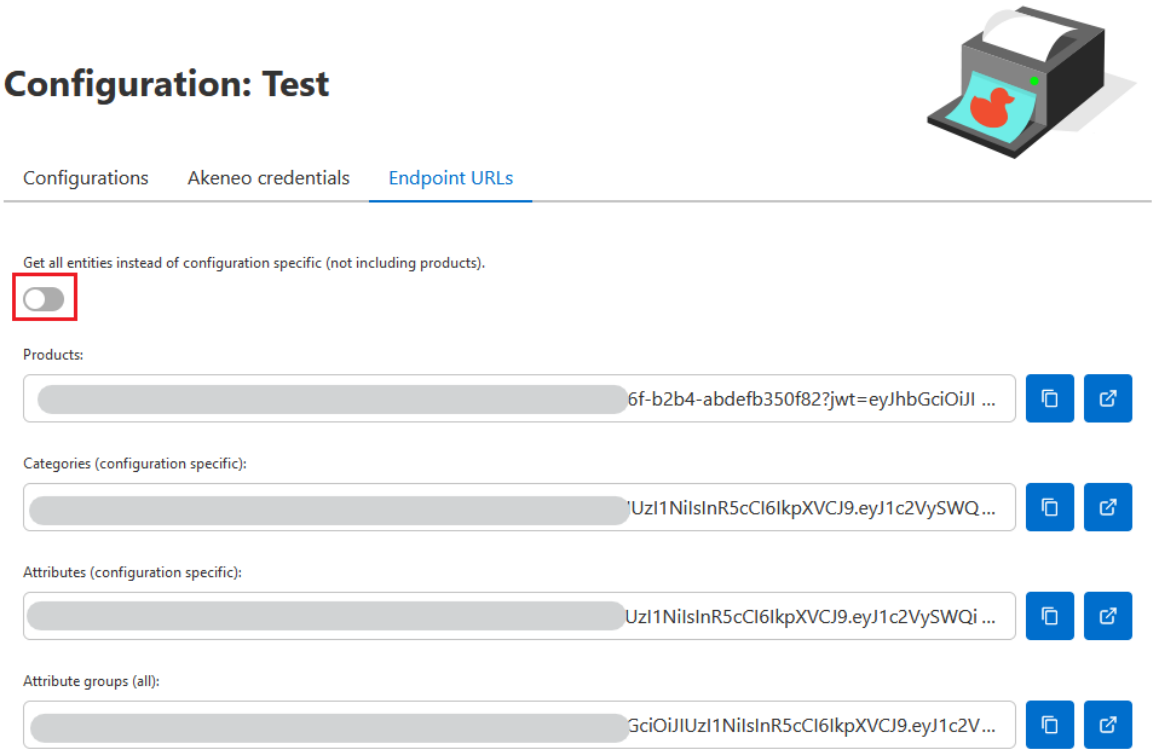
Kuvio 12. Akeneo EasyCatalog InDesign Connectorin konfiguraationäkymä

Kuviossa 13 on esitelty konfiguraation muokkausnäkymää, jossa valittua konfiguraatiota on mahdollista muokata tai se voidaan poistaa kokonaan. Tässä näkymässä on kolme välilehteä. Konfiguraation muokkausvälilehti, Akeneo tunnukset-välilehti, jossa konfiguraatiolle voidaan määrittää konfiguraatio kohtaiset Akeneo PIM tunnukset sekä XML-osoitteet välilehti, josta löytyy URL-osoitteet eri XML-sanomille (ks. Kuvio 14).



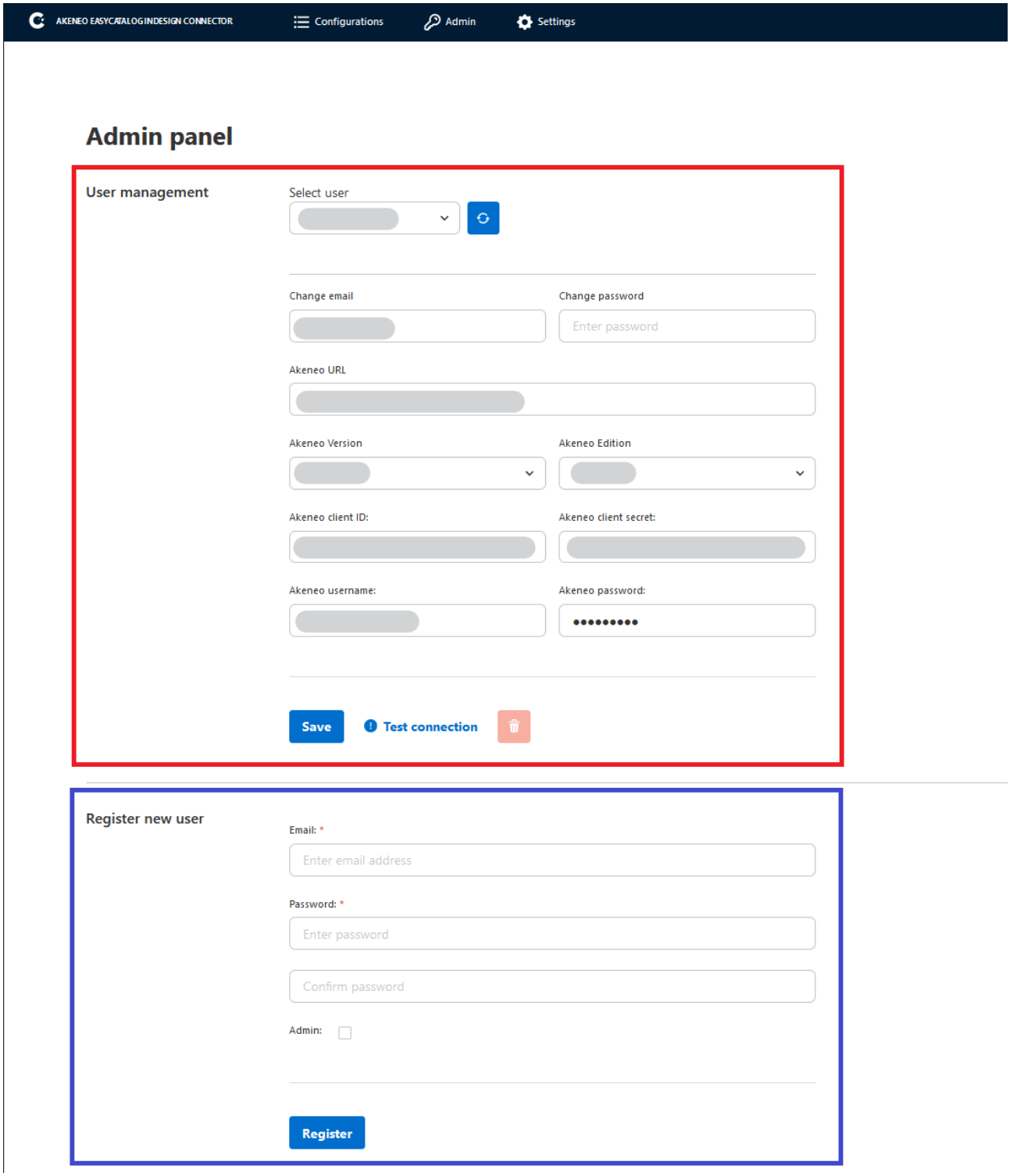
Kuvio 13. Akeneo EasyCatalog InDesign Connectorin kofiguraatio muokkausnäkymä

Konfiguraation XML-sanomien osoitenäkymästä (ks. Kuvio 14) saadaan URL-osoitteet XML-sanomille, joita voidaan sitten hyödyntää käyttökohteen mukaan. Valittavissa on tuote-, kategoria-, attribuutti- sekä attribuuttiryhmäsanomat. Käyttäjä voi joko tallentaa osoitteen suoraan leikepöydälle tai avata sen selaimessa uuteen välilehteen tarkastelua varten. Punaisella värillä korostetun napin avulla käyttäjä voi valita tuleeko XML-sanoman mukana kaikki Akeneo PIM-järjestelmässä saatavilla olevat tiedot vai pelkästään konfiguraatiokohtaiset tiedot. Tämä ei koske tuotesanomaa, missä haetaan aina vain konfiguraation mukaiset tiedot, eikä attribuuttiryhmäsanomaa, missä haetaan kaikki saatavilla olevat attribuuttiryhmät.



Kuvio 14. Konfiguraation XML-sanomien osoitenäkymä

Hallintapaneelissa (ks. Kuvio 15) sovelluksen ylläpitokäyttäjät voivat tarvittaessa hallita sovellusta monin eri tavoin. Kuviossa punaisella värillä on korostettu käyttäjienhallintakomponentti, jossa ylläpitokäyttäjät voivat muuttaa toisten käyttäjien salasanoja, muokata käyttäjälle asetettuja Akeneo PIM tunnuksia sekä myös tarvittaessa poistaa käyttäjiä. Käyttäjä ei voi kuitenkaan poistaa itseään, koska muuten sovellus voisi jäädä ilman mitään käyttäjää, jolloin palvelun käyttäminen olisi mahdotonta. Kuviossa sinisellä värillä on korostettu käyttäjienluontilomake. Lomakkeen avulla voidaan luoda uusia peruskäyttäjiä sekä myös ylläpitokäyttäjiä.



Kuvio 15. Akeneo EasyCatalog InDesign Connector hallintapaneeli

6 Pohdinta

Opinnäytetyön tavoitteena oli toteuttaa web-pohjainen SaaS-palvelu, jonka avulla asiakas voisi luoda erilaisia konfiguraatioita, joita käytetään tuotetietojen suodattamiseen Akeneo PIM tuotetiedonhallintajärjestelmästä. Suodatetuista tuotetiedoista luotaisiin XML-sanoma, joka olisi tarkoitus viedä Adobe InDesign taitto-ohjelman EasyCatalog-lisäosalle. Lisäosan ja XML-sanoman avulla voidaan luoda muun muassa erilaisia tuotekuvastoja, esitteitä ja käyttöohjeita. Sovelluksen keskeisenä tavoitteena oli helpottaa printtiautomaation kanssa työskentelevien asiakkaiden arkea nopeuttamalla rikkaan tuotetiedon viemistä taitto-ohjelmalle.

Sovellus toteutettiin multi-tenant SaaS-palveluna, joka olisi jatkuvasti kaikkien asiakkaiden saatavilla ja käytettävissä. Sen kehitystyöhön sovellettiin viisikerroksista kerrosarkkitehtuuria ja toteutukseen käytettiin React.js ja Node.js tekniikoita, joista oli jo valmiiksi hyvin kokemusta. Tuloksena saatiin täysin toimiva ratkaisu, jossa käyttäjä voi helposti luoda erilaisia konfiguraatioita yksinkertaisen käyttöliittymän avulla. Konfiguraatiota hyödynnetään tuotetietojen suodattamiseen Akeneo PIM tuotetietohallintajärjestelmästä, jonka jälkeen suodatetuista tuotetiedoista generoidaan XML-sanoma. Työssä päästiin siis täysin tavoitteisiin, eikä toteutuksesta jäänyt puuttumaan mitään oleellista.

Sovelluksen toteuttamisen aikana tuli vastaan useita ongelmia. JSON-muotoisten tuotetietojen muuttaminen oikeanlaiseksi XML-sanomaksi oli hyvin työläs prosessi, jossa kohdattiin useita ongelmia. Aluksi XML-sanomien generoiminen oli erittäin hidasta, kun esimerkiksi tuhannen tuotteen prosessointiin meni yli 20 minuuttia. Tämä johtui suuresta HTTP-pyyntöjen määrästä generoinnin aikana, joka saatiin ratkaistua välimuistia hyödyntämällä. Ratkaisun avulla tuhannen tuotteen prosessointi vei vain seitsemän sekuntia, joka on huomattava parannus aiempaan 20 minuuttiin. Toisena isona ongelmana oli suurien tuotemäärien prosessointi, joiden käsittelyyn sovelluksella ei ollut riittävästi resursseja. Ratkaisuna XML-sanoma generoidaan ja palautetaan käyttäjälle osissa, jolloin sovelluksen suorituskyky parani huomattavasti. Nämä ongelmat olivat kuitenkin hyvin opettavaisia ja niistä saaduista opeista on varmasti hyötyä tulevaisuudessa.

Työn tuloksena kehitettyä palvelua tullaan hyödyntämään useiden eri asiakkaiden tarpeissa sekä sitä tullaan myös kauppaamaan Akeneon kauppapaikalla. Sovelluksen generoimaa XML-sanomaa voidaan hyödyntää moneen eri käyttökohteeseen, mutta se on lähtökohtaisesti suunniteltu Adobe

InDesign taitto-ohjelman EasyCatalog-lisäosalle. Ratkaisu tulee helpottamaan monia asiakkaita, joilla on tarvetta viedä XML-muotoista tuotedataa esimerkiksi Adobe InDesign taitto-ohjelmalle tai johonkin muuhun käyttökohteeseen.

Opinnäytetyö aiheena onnistui erittäin hyvin, sillä tuotetiedonhallinta sekä SaaS-palvelut ovat jatkuvasti kasvava aihe ja tällaisille ratkaisulle tulee olemaan kysyntää nyt ja tulevaisuudessa. Vaikka samantyylisiä ratkaisuita oli olemassa, ei kuitakaan täysin samanlaista ratkaisua löytänyt, mitä tässä opinnäytetyössä käsitellään.

Tämä toteutus oli vasta sovelluksen ensimmäinen versio, ja tätä kirjoittaessa on jo suunnitteilla useita uusia ominaisuuksia eri asiakkaiden tarpeisiin. Tulevaisuudessa sovellukseen olisi hyvä lisätä vaihtoehto JSON- ja CSV-sanomille XML-sanoman lisäksi. Tälle ominaisuudelle ei ollut kuitenkaan tarvetta ensimmäisessä versiossa.

Lähteet

About Node.js. N.d. Esittelyteksti Nodejsn verkkosivuilla. Viitattu 5.4.2022. https://nodejs.org/en/about/

Brook, C. 2020. SaaS: Single Tenant vs Multi-Tenant - What's the Difference?. Blogi Digital Guardianin verkkosivuilla. Julkaistu 1.12.2020. Viitattu 20.1.2022. https://digitalguardian.com/blog/saas-single-tenant-vs-multi-tenant-whats-difference

Brown, N. 2021. The History of SaaS | The New Digital Revolution, Explained. Artikkeli Accelerate Agencyn verkkosivuilla. Julkaistu 6.12.2021. Viitattu 20.2.2022. https://accelerateagency.ai/the-history-of-saas

Carey, S. 2021. What is SaaS? Software as a service defined. Artikkeli InfoWorldin verkkosivuilla. Julkaistu 22.7.2021. Viitattu 20.1.2022. https://www.infoworld.com/article/3226386/what-is-saas-software-as-a-service-defined.html

Definition – What does Product Information Management mean?. N.d. Artikkeli PIM-Auswahlin verkkosivuilla. Viitattu 26.2.2022. https://pim-auswahl.de/en/what-is-pim/

Dillet, R. 2019. Akeneo raises $46 million for its product information management service. Artikkeli TechCrunchin verkkosivuilla. Julkaistu 12.9.2019. Viitattu 13.2.2022. https://techcrunch.com/2019/09/12/akeneo-raises-45-million-for-its-product-information-management-service/

Express Fast, unopinionated, minimalist web framework for Node.js. N.d. Esittelyteksti Expressin verkkosivuilla. Viitattu 5.4.2022. https://expressjs.com/

Ghosh, A. 2019. What is the Difference Between a Web Application and a SaaS Application?. Artikkeli The Customize Windowsin verkkosivuilla. Julkaistu 2.4.2019. Viitattu 20.2.2022. https://thecustomizewindows.com/2019/04/what-is-the-difference-between-a-web-application-and-a-saas-application/

Miksi tuotetiedonhallinta tekee liiketoiminnasta kannattavampaa. 2018. Blogi Canterin verkkosivuilla. Julkaistu 10.4.2018. Viitattu 13.2.2022. https://www.canter.fi/tuotetiedonhallinta-uutisia/tuotetiedonhallinta-tekee-liiketoiminnasta-kannattavampaa/

Mikä ihmeen Vincit?. N.d. Yritysesittely Vincitin verkkosivuilla. Viitattu 29.1.2022. https://www.vincit.fi/fi/medialle/

Mitä me teemme. N.d. Julkaisu Vincitin verkkosivuilla. Viitattu 29.1.2022. https://www.vincit.fi/fi/mita-me-teemme/

Richards, M. 2015. Software Architecture Patterns. Sebastopol, CA: O'Reilly Media.

Short, T. & Spiller, L. 2022. What is SaaS? 10 FAQs About Software-as-a-Service. Artikkeli Software Advicen verkkosivuilla. Julkaistu 5.1.2022. Viitattu 20.2.2022. https://www.softwareadvice.com/resources/saas-10-faqs-software-service/

Single-Tenant Vs. Multi-Tenant Cloud: Which Should You Use?. 2021. Blogi CloudZeron verkkosivuilla. Julkaistu 11.6.2021. Viitattu 30.1.2022. https://www.cloudzero.com/blog/single-tenant-vs-multi-tenant

The Importance of the Product Information Management (PIM). 2019. Artikkeli TechBullionin verkkosivuilla. Julkaistu 3.9.2019. Viitattu 13.2.2022. https://techbullion.com/the-importance-of-the-product-information-management-pim/

Valdes, A. 2020. Single Tenant vs Multi Tenant: SaaS Architecture. Artikkeli DZonen verkkosivuilla. Julkaistu 17.9.2020. Viitattu 30.1.2022. https://dzone.com/articles/single-tenant-vs-multi-tenant-saas-architecture

What is PIM?. N.d. Julkaisu Akeneon verkkosivuilla. Viitattu 13.2.2022. https://www.akeneo.com/what-is-a-pim/