OULUN AMMATTIKORKEAKOULU

Mika Ylitalo

CMS-NAVIGAATIOMODUULI MAGENTO-VERKKOKAUPPA-ALUSTALLE

OULUN AMMATTIKORKEAKOULU

CMS-NAVIGAATIOMODUULI MAGENTO-VERKKOKAUPPA-ALUSTALLE

Mika Ylitalo
Opinnäytetyö
Kevät 2016
Tietotekniikka
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikka, ohjelmistokehitys

Tekijä(t): Mika Ylitalo
Opinnäytetyön nimi: CMS-navigaatiomoduuli Magento-verkkokauppa-alustalle
Työn ohjaaja: Eino Niemi
Työn valmistumislukukausi- ja vuosi: Kevät 2016
Sivumäärä: 28

Työn taustalla ovat Piimega Oy:n verkkokauppatoiminnan tarpeet nopeuttaa uusien ja jo olemassaolevien verkkokauppasivustojen luontia sekä hallinnointia.

Tavoitteena oli luoda moduuli, jonka avulla kaupan sisältösivujen luonti sujuu luontevammin Magenton alkuperäiseen muokkaamattomaan tapaan nähden tarjoamalla yhden hallintasivun kautta useamman sisältösivun luomisen sekä näiden linkkien asettamisen kaupan tai sivuston navigaatioon samassa prosessissa. Moduulin nimeksi tuli CMSNav.

Työn ohjelmisto perustuu yleisiin Magenton kehitysmenetelmiin. Työssä on käytetty muualla lähdekoodissa löytyviä moduulien kehitystapoja sekä yleisiä Magenton kehityskäytäntöjä. Useammat ohjelmistotekniset lähdesivustot ovat toimineet Magenton kehitystapojen sekä yleisten ohjelmistokäytäntöjen tietolähteenä.

Moduulin kehityksessä saavutettiin sen perustoimintojen loppuunvienti eli Magento-verkkokaupan sisältösivujen luonti, muokkaus ja poisto. Löydettyjen, tämän kirjoituksen aikana vielä korjaamattomien virheiden ratkaisu jäi jatkokehitykselle. Koska moduuli on vielä kehitysvaiheessa, ovat olemassa olevat virheet, testaus sekä loppuviimeistely vielä suorittamatta. Mainittakoon vielä, että moduuli on luotu Magenton versio 1 alustalle, eikä versiolle 2.

Asiasanat: verkkokauppa, ohjelmistomoduulit, sisällönhallinta.

ABSTRACT

Oulu University of Applied Sciences
Information technology, software development

Author(s): Mika Ylitalo
Title of thesis: CMS Navigation Module For Magento
Supervisor(s): Eino Niemi
Term and year when the thesis was submitted: Spring 2016
Number of pages: 28

Thesis' background lies in Piimega Oy's e-commerce branch's needs to speed up the creation and management of new web stores and their content pages.

The goal was to create a module which eases the creation of a store's content pages compared to Magento's native approach by offering a single management page for creating multiple static pages for the store where the module is installed. The links of these created pages would also be included in the main site navigation bar within the same creation process.

Module's code is based on common Magento development practices. Magento's own core modules have acted as a base and reference for creating the logic and functionality of this work alongside numerous external resources.

Apart from completing the module's base functionality it still has numerous faults to fix with testing yet to be done. Because of these incomplete tasks the module is still in the development stage.

Keywords: Magento, e-commerce, content management.

SISÄLLYSLUETTELO

ALKULAUSE

Kiitokset Piimega Oy:lle ensimmäisistä askeleistani ohjelmistotekniikan alalla työelämän näkökulmasta. Erityiskiitos Piimega Oy:n verkkokauppaliiketoiminnan johtaja Mika Myllerille työn aiheesta.

3.4.2016

Mika Ylitalo

1 JOHDANTO

Magento on PHP-ohjelmointikieleen, Zend-viitekehykseen ja MVC-arkkitehtuuriin perustuva avoimen lähdekoodin sisällönhallintajärjestelmä, joka on suunnattu verkkokauppaliiketoimintaan pienistä isoihin yrityksiin (1). Tämän työn pääasiallisena ohjelmointikielenä toimi PHP.

Alustalla on kaksi versiota: ilmainen Community-versio sekä maksullinen Enterprise-versio. Kummankin version alustan kehitystyö tapahtuu moduulipohjaisesti niin, että kehittäjä kauppaa laajentaessaan luo erilaisia moduuleja haluamansa toiminnallisuuden saavuttamiseksi.

Työn toimintaympäristö kohdistuu verkkokaupan hallintanäkymän käyttäjän päivittäiseen työhön. Jotta kaupan asiakasnäkymän sisältösivujen luonti onnistuisi vaivattomammin Magenton oman sisällönhallinnan sijasta, oli tähän keksittävä Magenton perusversion ulkopuolinen ratkaisu. Tämän opinnäytetyön aiheena on tarjota ratkaisuksi Magento-moduuli nimeltä CMSNav.

Pelkkään perusversioon pohjautuvan Magento-verkkokaupan hallinnoinnissa käyttäjän täytyisi luoda jokainen asiasnäkymän sisältösivu yksitellen käyden luomisprosessin läpi kerta toisensa jälkeen. Työssä luodulla moduulilla CMSNav hallinnoitsija voi luoda useamman sisältösivun yhdeltä hallintasivulta yhden napin painalluksella. Tämän lisäksi moduuli lisää hallinnoitsijan luomat sivut Magenton asiakasnäkymän päänavigaatioon.

2 TYÖKALUT JA TEKNOLOGIAT

2.1 Magento-verkkokauppa-alusta

Magento on verkkokauppayrittäjille suunnattu modulaarinen verkkokauppa-alusta. Tämä tarkoittaa sitä, että verkkokauppaa voidaan laajentaa erilaisilla Magento-moduuleilla. Tällaisia moduuleja ovat esimerkiksi kehittyneempi etsi-toiminto, jolla loppukäyttäjä voi etsiä kaupasta esimerkiksi tuotetta sen nimellä. Tämän opinnäytetyön aiheena on nimenomaan moduuli Magento-verkkokauppa-alustalle.

2.2 Sisällönhallintajärjestelmä

Sisällönhallintajärjestelmä, englanniksi Content Management System (CMS) on ohjelmistojärjestelmä, jonka tarkoituksena on antaa pääkäyttäjälleen valtaa määritellä järjestelmän sisältöä. Magenton päivittäisessä käytössä tämä nähdään tilanteissa, missä verkkokaupan hallinnoitsija eli Magenton hallintanäkymän käyttäjä haluaa laatia omia sisältösivujaan ja muokata näitä mielensä mukaan. Tällaisia esimerkkejä ovat esimerkiksi yhteystietosivut ja verkkokauppojen tuotteita listaavat sisältösivut.

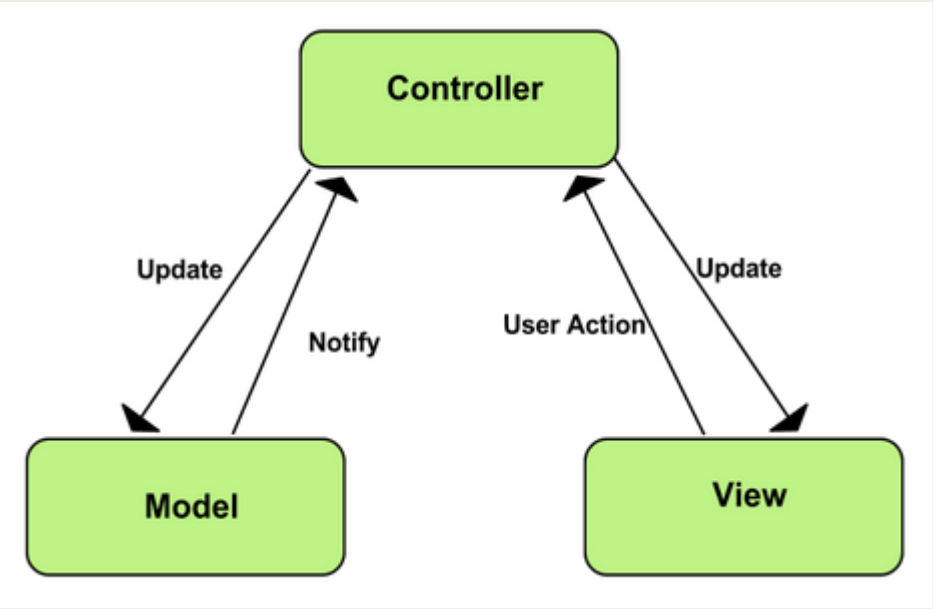
2.3 Avoin lähdekoodi

Avoin lähdekoodi (open source) on Open Source Initiativen määrittelemä termi lähdekoodille, joka määrittelee ohjelmiston lähdekoodin vapaan jakelun säännöt ja lisenssit (2). Magenton Community-versiolla on käytössä Open Source Initiativen alaisuudessa oleva Open Software License (OSL 3.0), mutta Enterprise-versiossa noudatetaan EULA:n (End-user License Agreement) ehtoja (3).

2.4 MVC-arkkitehtuuri

MVC (Model-View-Controller) arkkitehtuurissa (kuva 1) ohjelmisto jaetaan kolmeen, nimensä mukaiseen kategoriaan, missä näkymä (View) määrittelee mitä loppukäyttäjä näkee, Kontrolleri (Controller) käsittelee loppukäyttäjän käyttäytymisen ja näkymän toimintojen logiikan. Lopuksi datamalli (Model) sisältää jonkun tietokokonaisuuden, esimerkiksi yksittäisen verkkokaupan tuotteen datamallinnuksen (4).

Esimerkkitapauksessa käyttäjä voi Magentoon pohjautuvassa verkkokaupassa valita tuotekategorialistan näkymältä tuotteen, jonka alapuolella olevaa osta-painiketta klikkaamalla hän siirtää tuotteen ostoskoriin. MVC-arkkitehtuurin mukaisesti näkymä näyttää tuotteen pääkäyttäjälle. Tuotteella on tietynlaista tietoa datamallinsa mukaisesti (esimerkiksi kenkien koko). Ostajan painaessa Osta-painiketta näkymän Kontrolleri ohjelmallisen logiikkansa mukaisesti siirtää tuotteen datamallin mukaisen objektin ostoskoriin ja ilmoittaa näkymälle ja samalla käyttäjällensä tapahtumasta luomalla esimerkiksi ilmoituksen ostoskorin päivittymisestä.



KUVA 1. MVC-Arkkitehtuuri (5)

2.5 Zend Framework -viitekehys

Zend Framework on PHP-ohjelmointikieleen perustuva avoimen lähdekoodin oliopohjainen viitekehys web-sovellusten kehittämiseen. Viitekehys on maininnan arvoinen sillä Magento perustuu pääosin Zend Frameworkiin (6).

2.6 Ext JS -viitekehys

Ext JS on Sencha-nimisen yrityksen ylläpitämä MVC/MVVM (Model View Controller / Model View View Model) Javascript-viitekehys web-sovellusten kehittämiseen (7). Tämän opinnäytetyön ohjelmistossa Ext JS tulee esiin moduulin hallinta-asetuksissa sivuhierarkian visualisoinnissa ja sivuhierarkian muokkaamisessa (kuva 6).

2.7 PhpStorm-ohjelmointiympäristö

PhpStorm on PHP-kehitykseen räätälöity IntelliJ IDEAan perustuva IDE (Integrated Development Environment), mistä PhpStormiin on tuotu sen web-kehitykseen liittyvä toiminnallisuus.

2.8 Git-versionhallintajärjestelmä

Git (8) on avoimen lähdekoodin versionhallintajärjestelmä ohjelmistokehitykseen. Versionhallintajärjestelmän hyötynä saadaan askel askeleelta käytävä kehityskäyrä, jossa ohjelmistoa voidaan lokaalissa kehityksessä vaikka palauttaa aikaisempaan tilaan syystä tai toisesta.

Modulaarisessa kehityksessä Git on hyödyllinen, kun ohjelman eri osia ja toiminnallisuuksia voidaan kehittää yksitellen muun muassa kehitystiimin kanssa yhteistyössä. Tämän johdosta, kun esimerkiksi kehittäjä A tekee muutoksia samaan ohjelmistoon, hänen muutoksensa eivät sotke alkuperäistä ohjelmakoodia. Tämän Magento moduulin kehityksessä käytettiin apuna Git-versionhallintaa ja Bitbucketia (https://bitbucket.org).

3 TOTEUTUS

3.1 Katsaus Magenton moduulikehitykseen

Magentolle kehittäminen vaatii tiukkaa sen asettaman arkkitehtuurin noudattamista. Moduulin oma tiedostohierarkia täytyy olla oikeaoppisesti aseteltu moduulin toiminnan varmistamiseksi (9). Jokaisen luodun moduulin tulisi noudattaa samaa tiettyä tiedostohierarkiaa (kuva 2).

```
app/code/local/Modulenamespace/Modulename/Block
app/code/local/Modulenamespace/Modulename/controllers
app/code/local/Modulenamespace/Modulename/etc
app/code/local/Modulenamespace/Modulename/Helper
app/code/local/Modulenamespace/Modulename/Model
app/code/local/Modulenamespace/Modulename/sql
```

KUVA 2. Magento-moduulin funktionaalisen ohjelmiston tiedostohierarkia

Kuvassa app-kansio on Magenton tiedostorakenteen juuri, code-kansio toiminnallisen ohjelmakoodin juuri ja local-kansio ajatuksellisesti lokaaliin kehitykseen tarkoitettu kansio. Code-kansion juuressa on myös core- ja community-kansiot. Core-kansion tarkoitus on toimia Magenton natiivien moduulien säilöntäpaikkana ja community-kansio yleiseen jakeluun tarkoitettujen moduulien säilytyspaikkana.

Yllä oleva kansiorakenne vastaa vain toiminnallisen koodin hierarkiaa. App-kansion juuressa on myös design- sekä etc-kansiot (kuva 3).

```
app/design/adminhtml
app/design/frontend

app/design/adminhtml/default/default/layout
app/design/adminhtml/default/default/locale
app/design/adminhtml/default/default/template

app/design/frontend/base/default/layout
app/design/frontend/base/default/locale
app/design/frontend/base/default/template
```

KUVA 3. Teemojen tiedostohierarkia

Kuvassa havainnollistetaan edellä mainittua kansiohierarkiaa design-kansion näkökulmasta. Adminhtml-kansio on tarkoitettu Magenton hallintanäkymän tyylipohjaisiin tiedostoihin ja frontend-kansio Magenton asiakasnäkymiin eli sivuille, joita ei ole tarkoitettu hallintanäkymään.

Adminhtml- ja frontend-kansioiden alla olevat default/default sekä base/default ovat teemakohtaisia kansioita. Riippuen kaupan hallintanäkymän kautta aktiiviseksi asetetun teeman (esim. default) perusteella kyseisen teeman tiedostot haetaan niille määritellyistä kansioista.

Layout-, locale- sekä template-kansiot sisältävät xml-tyyppiset näkymien konfiguraatiotiedostot, csv-tyyppiset kielikäännöstiedostot eri sanoille ja lauseille, mitä näkymissä käytetään sekä phtml-tyyppiset näkymätiedostot.

Moduulin keskeisin kansio on etc-kansio (kuva 4). Tässä kansiossa määritellään modules-kansiossa moduulin xml-tyyppinen konfiguraatiotiedosto, joka kertoo Magentolle mm. mistä moduulin eri tiedostot löytyvät ja mihin ne kuuluvat. Tästä enemmän luvussa 3.2. Moduulin tiedostohierarkia ja tarkoitus.

```
app/etc
app/etc/modules
```

KUVA 4. Konfiguraatiotiedostojen tiedostohierarkia

3.2 Moduulin tiedostohierarkia ja tarkoitus

Opinnäytetyössä kehitetyn CMSNav-moduulin tiedostohierarkia myötäilee samaa kaavaa kuin kappaleessa 3.1. kuvailtu Magento-moduulin tiedostohierarkia (kuva 5).

```
app/code/local/Piimega/Cmsnav/Block
app/code/local/Piimega/Cmsnav/controllers
app/code/local/Piimega/Cmsnav/etc
app/code/local/Piimega/Cmsnav/Helper
app/code/local/Piimega/Cmsnav/Model
app/code/local/Piimega/Cmsnav/sql

app/design/adminhtml/default/default/
layout/piimega
app/design/adminhtml/default/default/
locale/en_US
app/design/adminhtml/default/default/
template/piimega/cmsnav

app/design/frontend/base/default/
layout/piimega
app/design/frontend/base/default/
template/catalog/navigation

app/etc
app/etc/modules
```

KUVA 5. CMSNav tiedostohierarkia

Block-kansioon kuuluvat moduulin näkymien dataobjektit, jotka ovat vastuussa näkymän datan generoimisesta. Tässä on huomioitava, ettei vastaava lähestymistapa vastaa perinteistä MVC-arkkitehtuuria, sillä Magentossa Kontrolleri ei vie tietoa näkymälle, vaan näkymä on jaettu näky-

män malliin (template) sekä tämän dataobjektiin (block) (10). Jos moduulissa käsitellään Magenton hallintanäkymää kuten CMSNav-moduulissa tehdään, on tämän kansion alla myös Adminhtml-kansio, jonne sen osion ohjelmallisuus kuuluu.

Controllers-kansiossa sijaitsevat moduulin Kontrollerit. MVC-arkkitehtuureissa yleisesti Kontrollerin tarkoitus on toimia URL-reitittäjänä ohjaamalla ohjelmisto ajamaan tietty Kontrolleri-tiedosto ja varsinkin tämän action-metodi riippuen URL:n kokonaisuudesta. Action-metodin tarkoitus taas on tuoda data Kontrollerille, jonka avulla tämä voi sen näkymään tulostaa. Magentossa tämä kuitenkin hieman poikkeaa periteisestä MVC-arkkitehtuurista, sillä moduulin globaalissa konfiguraatiossa on määritelty URL-reitit Kontrollereille. Jos konfiguraatiosta löytyy Kontrolleri tietylle URL:lle, tämän konfiguraation Kontrolleri luodaan ja sen action-metodi ajetaan (11).

Etc-kansio pitää sisällään moduulin konfiguraatiotiedostot. Nämä tiedostot kertovat Magentolle, mistä mikäkin moduulin tiedosto löytyy. Näitä tiedostoja ovat muun muassa block-dataobjektit, datamalli-objektit sekä layout-tiedostojen sijainnit.

Helper-kansiossa sijaitsee yleiskäyttöisiä avustus-metodeja. Näiden tarkoitus on suorittaa tehtäviä, joita voidaan ajaa useammassa eri paikassa moduulia.

Model-kansioon kuuluvat luokat, joiden avulla Magento kommunikoi sen tietokannan ja moduulin omien tietokantataulujen kanssa.

Sql-kansiossa sijaitsee tiedostot moduulin tietokantaulujen luomiseen. On huomioitavaa, että näiden tiedostojen nimien tulee vastata sen hetkistä moduulin versiota, jotta ne voidaan ajaa.

Toiminnallisen ohjelmakoodin lisäksi CMSNav-moduuli pitää sisällään käyttäjän näkemiä näkymiä. Näiden tiedostot sijaitsevat design-kansion juuressa adminhtml- ja frontend-kansioiden alla. Adminhtml-kansio pitää sisällään moduulin hallinnan näkymien tiedostot ja frontend-kansio näkymätiedostot moduulin hallintanäkymän ulkopuolelle. Koska nämä näkymätiedostot kuuluvat oletusteemaan, on tiedostot sijoitettu default/default- sekä base/default-kansiorakenteiden alle. Täten kansiorakenne mukailee Magenton teema-asetuksia, jotka on määritelty verkkokaupan hallinnoitsijan toimesta Magenton hallintanäkymästä.

Adminhtml-kansiosta löytyvät layout-, template- sekä locale-kansiot. Layout-kansion tarkoituksena on adminhtml-kansion alla säilyttää CMSNav-moduulin hallinnan näkymien konfiguraatiotiedostoja. Template-kansion tarkoituksena on taas säilyttää näitä näkymätiedostoja. Locale-kansiosta taas löytyy käännöstiedostot halutuille kielille. Koska moduulin kehitystyö omalta osaltani jäi kesken, on locale-kansiossa ainoastaan käännöstiedosto englannille.

Frontend-kansion alla olevan layout-kansion alla sijaitsee xml-konfiguraatio CMSNav-moduulin yleisnäkymän konfigurointiin. Koska yksi moduulin ominaisuus on laajentaa Magenton perusnavigaatiota lisäämällä siihen käyttäjän moduulilla luomat sivulinkit, kertoo tämä konfiguraatiotiedosto mitä näkymätiedostoa ollaan laajentamassa. Tämä laajennettava näkymätiedosto löytyy template/catalog/navigation tiedostorakenteen alta, joka on teemakansiorakenne base/default alla frontend-kansion juuressa.

Lopuksi CMSNav-moduulin olemassaolosta on kerrottava Magentolle. Tämä tapahtuu luomalla xml-pohjainen konfiguraatiotiedosto app/etc/modules-kansioon, jota Magento lukee. Loppujen lopuksi kansiorakenteen läpikäyminen on vain pintaraapaisu moduulin toiminnasta.

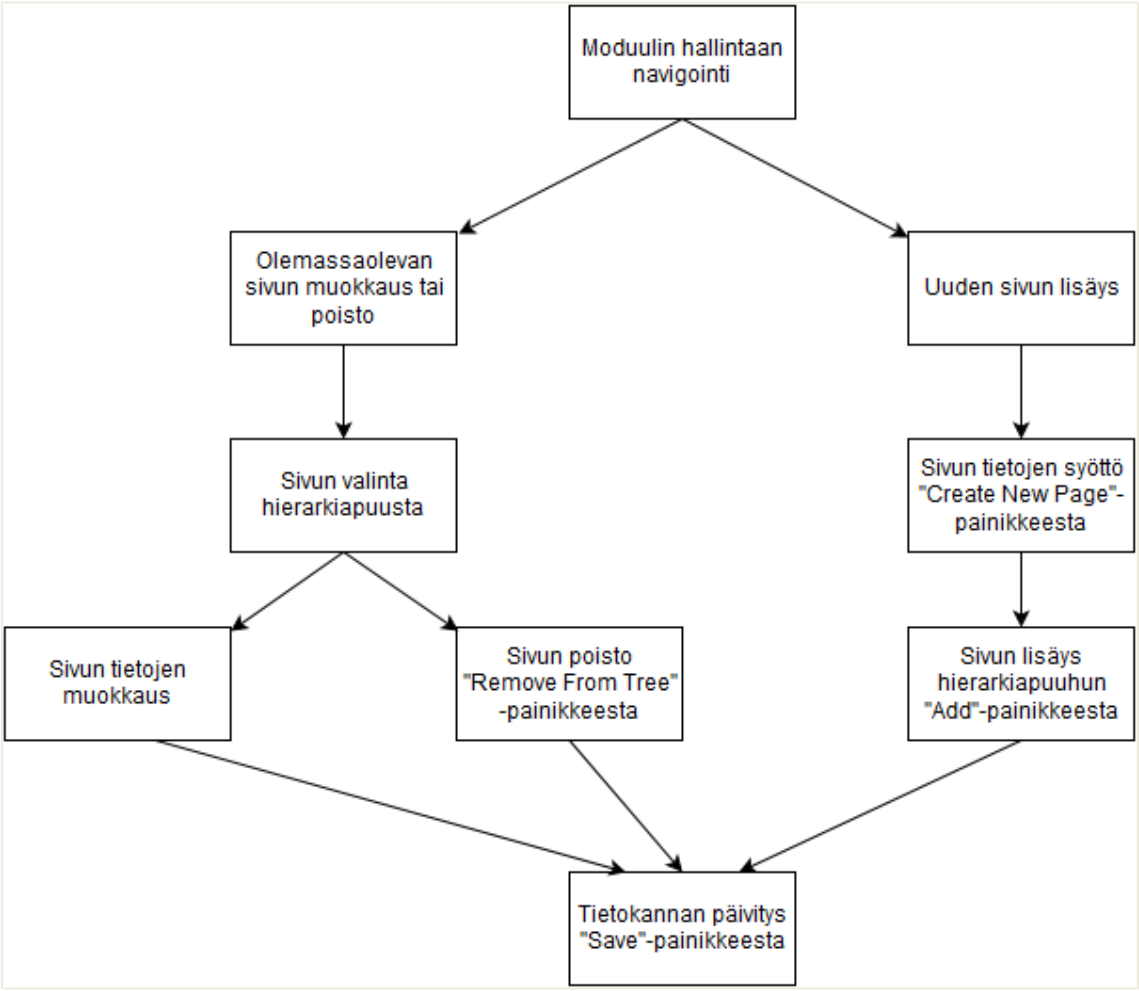
3.3 Moduulin toiminta

Tässä luvussa kerrotaan moduulin toiminta käyttäjän näkökulmasta. Moduulilla on käyttäjän näkymästä kaksi puolta, joita ovat hallinta- sekä asiakasnäkymä. Hallintanäkymässä tapahtuu moduuli- sekä sivukohtainen hallinta, missä hallinnoija luo sivut, asettaa niille sivun luonnin yhteydessä alustavan sisällön ja asettaa navigaation hierarkian haluamakseen ExtJS-pohjaisella hierarkiapuulla.

Sivujen luonnin jälkeen hallinnoija voi muokata sivujen sisältöä Magenton oman sisällönhallintajärjestelmän kautta joko navigoimalla sinne Magenton hallintasivulta taikka CMSNav-moduulin hallintasivun sivutaulukosta. Käyttäjä voi myös muokata tiettyjä sivun tietoja moduulin hallintasivulta sekä muokata sivujen hierarkiaa asiakasnäkymän navigaatiossa.

Asiakasnäkymässä moduulin toiminta näkyy Magenton oman asiakasnäkymän navigaation muokkaamisessa. Moduuli nimittäin lisää jo olemassa olevien navigaatiolinkkien perään CMSNav-moduulilla luodut linkit ja näiden hierarkian.

Moduulilla luotujen navigaatiolinkkien kohteita voivat olla Magento-verkkokaupan sisäiset sisältösivut taikka kaupan ulkopuoliset sivut (esim. www.piimega.fi). Kohteen asettaminen tapahtuu sivua luodessa taikka sitä myöhemmin muokatessa.



KUVA 6. CMSNav-moduulin käyttäjäkohtainen toimintakaavio

3.3.1 Moduulin hallintaan navigointi

Kaikki lähtee liikkeelle moduulin alustavasta konfiguraatiotiedostosta. Tämä tiedosto sijaitsee app/etc/modules kansion alla ja kertoo Magentolle moduulin olemassaolosta. Tiedostossa asetetaan moduuli aktiiviseksi, osoitetaan mistä sen lähdekoodi löytyy sekä moduulin riippuvuudet muihin moduuleihin nähden.

Moduulin toiminnallisen ohjelmakoodin tiedostohierarkiasta app/code/local/Piimega -kansiorakenteen alta löytyy etc-kansio missä on moduulikohtaiset xml-pohjaiset konfiguraatiotiedostot. Näitä ovat adminhtml.xml sekä config.xml. Edeltävässä tiedostossa on määritelty moduulin linkki Magenton hallintanäkymän navigaatioon, minkä kautta moduulia pääsee verkkokaupan hallintanäkymästä käyttämään (kuva 7). Tämän lisäksi tiedostossa konfiguroidaan uuden navigaatiolinkin ACL-oikeuksien (Access Control List) hallinta, jotta hallintanäkymän pääkäyttäjä kykenee hallitsemaan moduulin käyttöoikeuksia.

```
<menu>
    <cmsnav translate="title" module="piimega_cmsnav">
        <title>CMSNav</title>
        <sort_order>75</sort_order>
        <children>
            <cmsnav_settings translate="title" module="piimega_cmsnav">
                <title>Settings</title>
                <sort_order>1</sort_order>
                <action>adminhtml/cmsnav_settings/index</action>
            </cmsnav_settings>
        </children>
    </cmsnav>
</menu>
```

KUVA 7. Moduulin linkin luonti verkkokaupan hallintanäkymän navigaatioon

On huomioitavaa, että Magento kokoaa kaikkien moduulien adminhtml.xml tiedostot yhteen ja koostaa näistä yhteisen globaalin konfiguraation.

Config.xml tiedostossa määritellään moduulin versionumero, kerrotaan Magentolle, mistä moduulin eri ohjelmakoodin osat löytyvät, konfiguroidaan kirjoitus- ja lukuoikeudet tietokantaan, määritellään halutuille tapahtumille moduulikohtaiset käsittelijät, kerrotaan mistä Magenton ulkoasua muokkaavat konfiguraatiotiedostot löytyvät, sekä määritetään moduulikohtainen URL-reititys.

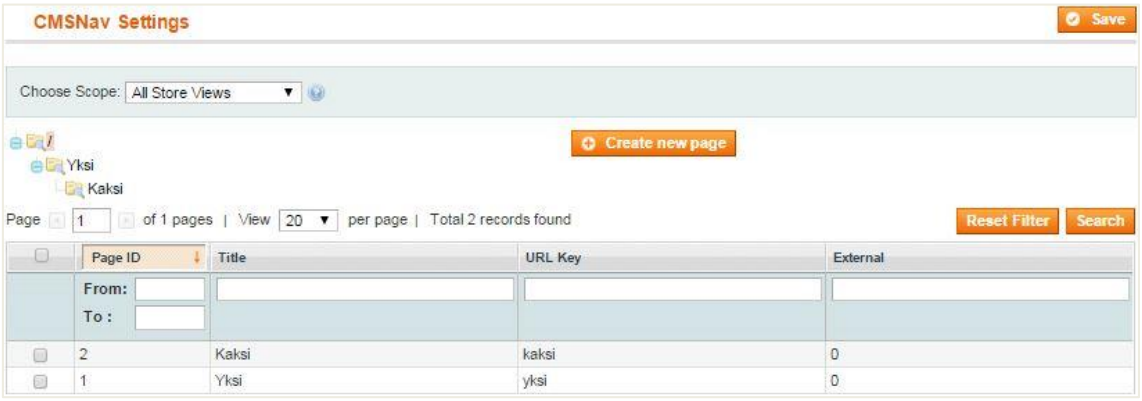
Kuvan 7 mukaisesti moduulin päälinkille on asetettu alilinkki, jolla on xml-kentässä action nimetty minkä Kontrollerin action-metodi ajetaan linkkiä painettaessa. Tässä tapauksessa Kontrollerin nimi on SettingsController ja sen ajettava action-metodi index. Voi näyttää oudolta miten Kontrolleri ja kyseinen metodi olisivat juuri näitä, kun katsoo action kentän arvoa. Magento kuitenkin lu-

kee arvoa niin, että se hakee controllers-kansion alta Adminhtml-kansion. Sieltä se löytää Cms-nav-kansion, jonka alta löytyy SettingsController-niminen Kontrolleri ja siltä taasen index-niminen metodi, joka lopulta ajetaan.

KUVA 8. Xml-konfiguraatiolla luotu moduulin hallintanäkymän linkki

3.3.2 Moduulin hallinnan käyttöliittymä

Kun lopulta kuvan 8 mukaista linkkiä painetaan, ajaa se mainitun SettingsController-kontrollerin index-nimisen action-metodin, jonka tehtävänä on luoda alustava näkymä moduulin käyttöliittymälle (kuva 9).



KUVA 9. Moduulin käyttöliittymä

Käyttöliittymästä löytyy kuvan 9 mukaisesti sivun lisääminen Create new page -painikkeen kautta. Tämä kuitenkaan ei vielä varsinaisesti luo uutta sivua ja sen etusivunäkymän navigaatiolinkkiä, vaan asettaa sen kuvassa 9 vasemmalla näkyvään ExtJS-pohjaiseen puuhierarkiaan, mistä käyttäjä voi vielä ennen tallennusta sekä tallennuksen jälkeen muokata luotuja sivuja. Sivujen tallennus, luonti ja navigaatiolinkkien luominen tapahtuu kuvan 9 oikeassa yläkulmassa näkyvästä Save-painikkeesta. Tämän onnistuessa käyttäjälle ilmoitetaan sivujen tallentuneen ja kuvan 9 alalaidassa näkyvään taulukkoon listataan luodut sivu jo luotujen lisäksi. Tämän taulukon toiminnot olivat tämän opinnäytetyön kirjoitushetkellä vielä kehitysvaiheessa. Näitä toimintoja ovat sivujen tietojen tarkempi hallinnointi sivukohtaista riviä painamalla.

Kuvan 9 ylälaidasta löytyy myös sivujen näkyvyysalueen hallinnointi. Aiemmin mainitun sivutaulukon lisäksi tämänkin toiminto on vielä opinnäytetyön kirjoitushetkellä kehitysvaiheessa. Toiminnon tarkoitus on loppujen lopuksi määrittää hallinnoitavien sivujen näkyvyysalue siinä mielessä, että Magento-verkkokaupassa voi olla useampia alikauppoja ja näiden kieliversioita. Näkyvyysalueella määritellään mille kaupoille sivut ja niiden navigaatiolinkit näkyvät.

Moduulin käyttöliittymän pohja on määritelty xml-pohjaisella konfiguraatiotiedostolla (kuva 10). Tästä tiedostosta voidaan nähdä suoraan sivun rakentuminen. Jos katsotaan block-osioiden name-kenttää nähdään, että piimega_cmsnav_settings_settings-osion alle kuuluu piimega_cmsnav_settings_manage.osio, missä kuvan 9 mukainen sivujen puuhierarkia sekä uuden sivun lisäyksen painike sijaitsee. Tämän osion alla on myös kuvan 9 mukainen taulukko johon luodut sivut listataan.

```
<layout>
    <adminhtml_cmsnav_settings_index>
        <reference name="head">
            <action method="setCanLoadExtJs"><flag>1</flag></action>
            <action method="addJs"><script>cmsnav/utility.js</script></action>
        </reference>
        <reference name="content">
            <block type="piimega_cmsnav/adminhtml_cmsnav_settings_edit" name="piimega_cmsnav_settings_settings">
                <block type="piimega_cmsnav/adminhtml_cmsnav_settings_manage"
                  name="piimega_cmsnav_settings_manage" template="piimega/cmsnav/manage.phtml"/>
                <block type="piimega_cmsnav/adminhtml_cmsnav_settings_edit_form_grid" name="cmsnav_grid" />
            </block>
        </reference>
        <block type="piimega_cmsnav/adminhtml_scope_switcher"
          name="scope_switcher" template="piimega/cmsnav//scope/switcher.phtml" />
    </adminhtml_cmsnav_settings_index>
</layout>
```

KUVA 10. CMSNav-moduulin hallintasivun pohjan konfiguraatio

3.3.3 Sivujen luonti ja poisto

Verkkokaupan hallinnoitsijan näkökulmasta moduulin hallinnointinäkymään navigoinnin jälkeen hän seuravaksi lisää uuden sivun ja tämän navigaatiolinkin. Kuvan 9 Create new page -painikkeen avulla hänelle avautuu kaavake (kuva 11), mihin syötetään itse sivun tiedot. Title-kenttään tulee sivun otsikko, URL Identifier -kenttään sivun URL-osio, Link points outside of store -valinnalla voidaan määrittää viekö asiakasnäkymän navigaatiolinkki kaupan ulkopuolelle (esim. www.google.com), Root -template valinnalla määritellään sivun alustava pohjapiirros, Meta

Keywords ja Meta Description kentät ovat hakukoneoptimointia varten ja lopuksi Content Heading -kentässä annetaan sivun sisäisen näkymään tuleva otsikko.

Link points outside of store -valinnasta riippuu luodaanko linkille omaa sivua. Jos linkki asetetaan ohjaamaan kaupan ulkopuolelle niin tässä tapauksessa ei oleellisesti sivun luontia vaadita. Tästä johtuen myös sivun tietokaavakkeen valinnat kapenevat siinä määrin, että Root template-, Meta Keywords-, Meta Description- sekä Content Heading-kentät piiloutuvat. Täten myös URL Identifier -kenttä vaatii täyden osoitteen sivulle, mihin linkki ohjaa. Sivun ollessa kaupan sisäinen tämä kenttä vaatii vain yhden sanan pituisen tunnisteen, koska se tullaan lisäämään kaupan URL-osoitteeseen (esim. http://magentokauppa.com/yksi).

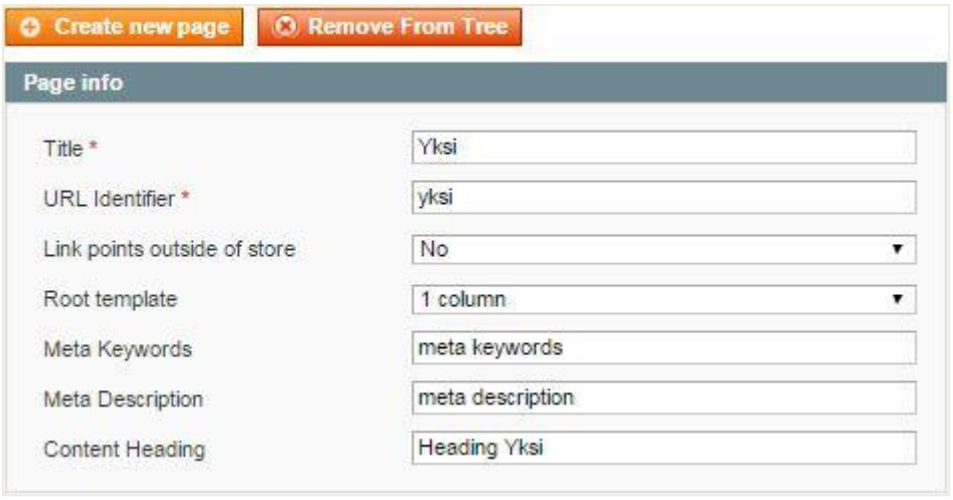
Root template -kentästä mainittakoon vielä, että tämän arvoja voivat olla ’1 column’, ’2 columns with left bar’, ’2 columns with right bar’ sekä ’3 columns’. Valitulla arvolla määritellään sivun asiakasnäkymän pohja. Esimerkiksi ’1 column’ -arvolla sivulle ei tule vasemmalle taikka oikealla erillistä osiota ylimääräiselle sisällölle, vaan sivu kattaa koko asiakasnäkymän leveyden (kuva 11).

KUVA 11. Kaavake lisättävän sivun tiedoille

Kun käyttäjä on tyytyväinen kuvan 9 kaavakkeeseen syöttämiinsä tietoihin, hän luo sivun alustavasti painamalla Add-painiketta, jonka kautta sivu listätään ExtJS-pohjaiseen hierarkiapuuhun kaavakkeen vasemmalle puolelle (kuva 9). Tästä käyttäjä voi vielä halutessaan muokata luotavan sivun tietoja valitsemalla haluamansa muokattavan sivun painamalla tätä. Tämän kautta sama kuvan 11 mukainen kaavake aukeaa aikaisemmin syötetyillä tiedoilla (kuva 12). Lisäksi kaavakkeen yläpuolelle ilmestyy Cancel ja Add -painikkeiden tilalle Create New Page- ja Remove From

Tree -painikkeet. Tässä kohtaa Create New Page -painikkeen avulla käyttäjä voi luoda jälleen uuden sivun, mutta tällä kertaa hierarkiapuusta valitun sivun alisivuksi, kuten kuvan 7 mukaisesti sivu Kaksi on luotu.

Create New Page -painikkeen vierellä olevan Remove From Tree -painikkeen avulla sivu taasen voidaan poistaa hierarkiapuusta. Täten, kun käyttöliittymän oikean yläkulman Save-painiketta (kuva 9) painetaan poistuu sivu Magenton tietokannasta ja samoten asiakasnäkymän linkki verkkokaupasta tälle sivulle poistuu näkyvistä, sillä linkit luodaan sen tiedon perusteella mitä tietokannasta saadaan CMSNav-moduulilla luotujen sivujen osalta.



KUVA 12. Sivun tietojen muokkauskaavake

3.3.4 Tarkempi katsaus Ext JS -hierarkiapuuhun

Alustavasti sivujen tietojen muokkauskaavakkeen kautta Ext JS -hierarkiapuuhun lisätyt sivut eivät vielä tallennu Magenton ja CMSNav-moduulin tietokantaan. Tämä hierarkiapuu toimii sivujen lisäyksessä vasta alustavana näkymänä sille, miltä sivujen hierarkia tulee näyttämään. Tämän ansiosta käyttäjä voi helposti siirtää sivujen paikkaa niiden yhtäläisessä hierarkiassa. Esimerkiksi käyttäjä loisi aluksi sivun Kaksi sivun Yksi alisivuksi, mutta luonnin jälkeen huomaa, että sivun oli tarkoitus olla uusi päätason linkki asiakasnäkymän navigaatioon nähden. Käyttäjän pitäisi tässä tapauksessa vain ottaa hiirellä kiinni sivusta Kaksi ja siirtää se pois sivun Yksi alta omaksi päätason linkiksi. Nyt käyttäjän ollessaan tyytyväinen sivujen hierarkiaan ja niiden alustaviin tietoihin hän

painaa kuvan 9 oikean ylälaidan mukaista Save-painiketta ja sivut tallentuvat Magenton tietokantaan CMSNav-moduulin tietokantatauluun.

Hierarkiapuu on luotu ExtJS-nimisellä Javascript-viitekehyksellä, joka ei ole Magenton kehittämä, vaan kuuluu Sencha Inc. -nimiselle yritykselle. Moduulin ohjelmakoodin näkökulmasta esimerkkinä hierarkianäkymän luonti hallintasivun alustuksen yhteydessä on kuvan 13 mukainen. Kuvassa esitelty createPages-funktio ajetaan nimenomaan silloin, kun CMSNav moduulin hallintasivu latautuu käyttäjän sille navigoidessa.

```
createPages: function() {
    for( var i = 0, len = this.pages.length; i < len; i++) {
        this.increment = len;

        var pageNode = new Ext.tree.TreeNode({
            id:                 this.pages[i].node_id,
            parent_node_id:     this.pages[i].parent_node_id,
            has_children:       this.pages[i].has_children,
            text:               this.pages[i].page_title,
            identifier:         this.pages[i].url_identifier,
            external:           this.pages[i].external_link,
            root_template:      this.pages[i].root_template,
            meta_keywords:      this.pages[i].meta_keywords,
            meta_description:   this.pages[i].meta_description,
            content_heading:    this.pages[i].content_heading,
            page_id:            this.pages[i].page_id,
            cls:                'cmsnav_page',
            expanded:           true,
            allowDrop:          true,
            allowDrag:          true
        });

        if(parent = pageNode.attributes.parent_node_id) {
            this.tree.getNodeById(parent).appendChild(pageNode);
        } else {
            this.treeRoot.appendChild(pageNode);
        }
    }
},
```

KUVA 13. ExtJS-hierarkiapuun luonti

3.3.5 Luotujen sivujen navigaatiolinkit ja sisältösivut

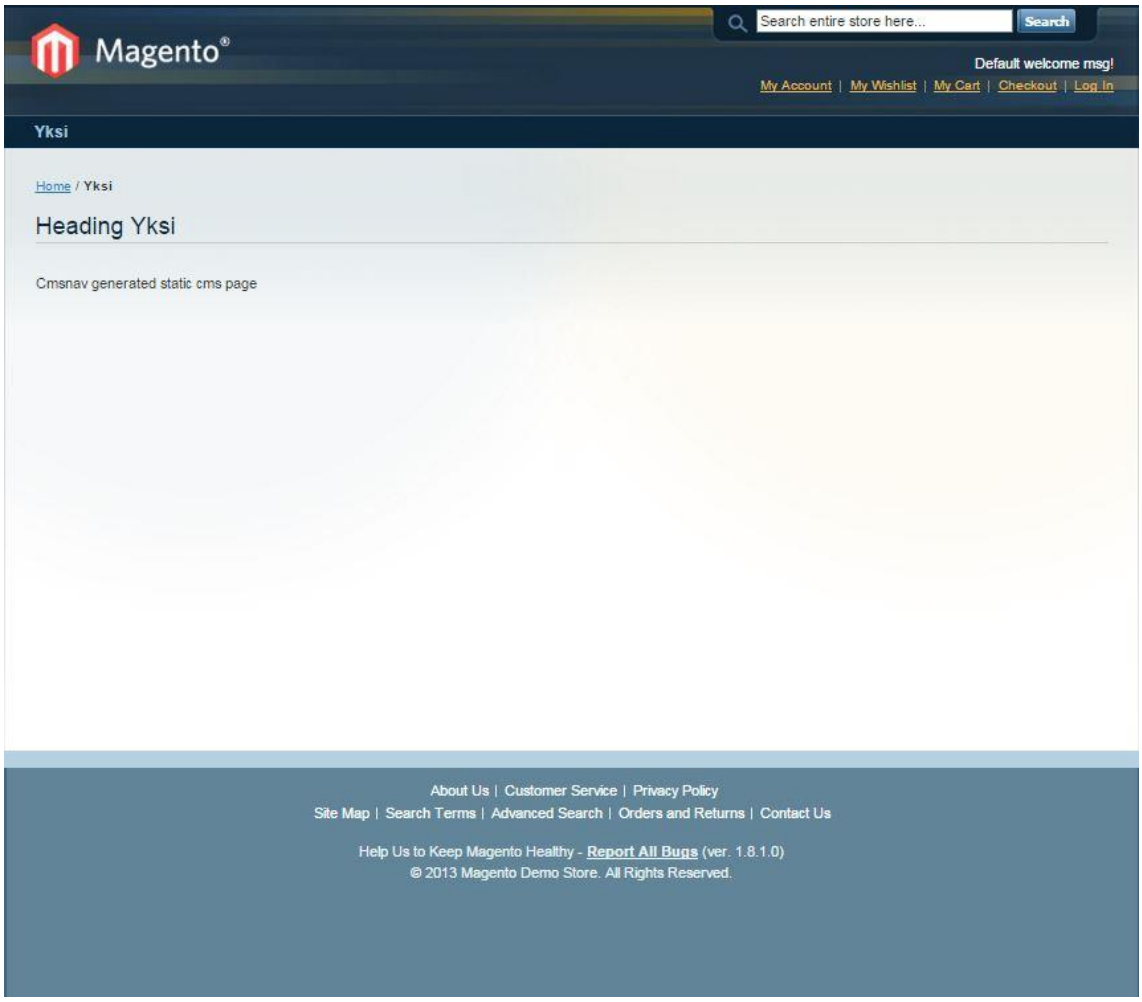
Luotuaan sivun ja tallennettuaan sen Magenton tietokantaan tulee luodun sivun linkki näkyviin kaupan asiakasnäkymään (kuva 14). Kuten kuvasta nähdään vastaa linkkien hierarkia CMSNav-moduulin hallintasivulla olleen Ext JS -pohjaisen puuhierarkian mallia, jossa sivu Kaksi toimii sivun Yksi alisivuna.

KUVA 14. CMSNav-moduulilla luodut navigaatiolinkit

Käyttäjän navigoidessa tässä tapauksessa sivulle Yksi avautuu hänelle kuvan 15 mukainen sivun asiakasnäkymä. Kun muistellaan kuvien 11 ja 12 mukaisia sivun tietojen kaavakkeita, huomataan tällä sivulla Title, Link points outside of store, Root template sekä Content Heading arvojen merkitys.

Title-kentän arvo nähdään jo navigaatiolinkistä, Content Heading-kentän arvo sivun otsikosta, joka on tässä tapauksessa Heading Yksi. Link points outside of store -kentän arvo ’No’ johti siihen, että sivu yleensä luotiin eikä navigaatiolinkkiä asetettu ohjaamaan kaupan ulkopuolelle. Lopuksi Root template -kentän arvo määritteli sivun asiakasnäkymän pohjan.

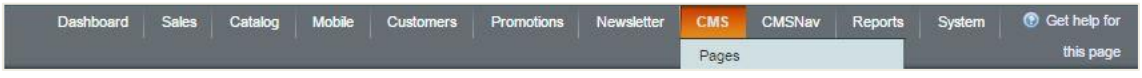
Root template -kentästä mainittakoon, että tämän arvoja voivat olla ’1 column’, ’2 columns with left bar’, ’2 columns with right bar’ sekä ’3 columns’. Esimerkiksi ’1 column’ -arvolla sivulle ei tule vasemmalle taikka oikealla erillistä osiota ylimääräiselle sisällölle, vaan sivu kattaa koko asiakasnäkymän leveyden, kuten sivun Yksi kohdalla on tehty (kuva 12 ja kuva 15).



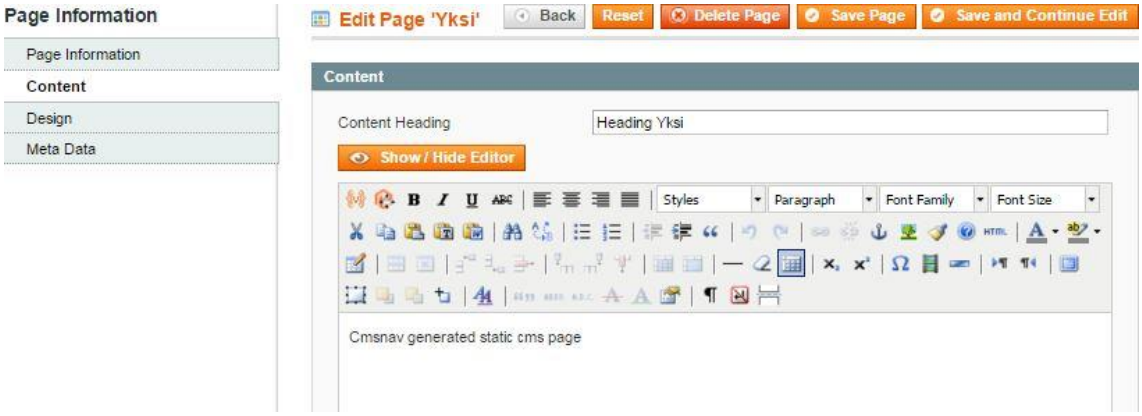
KUVA 15. Luodun sivun asiakasnäkymä

3.3.6 Sivujen sisällön muokkaus

Luotujen sivujen varsinaisen sisällön hallinnointiin pääsee Magenton omista sisällönhallinta-asetuksista. Sivun sisällön muokkaamiseen päästään verkkokaupan hallintanäkymän navigaation CMS-linkin alilinkistä Pages (kuva 16). Linkistä avautuu kaikki kaupan ja sen alikauppojen asiakasnäkymien sisältösivut eivätkä pelkästään CMSNav-moduulilla luodut sivut. Painettaessa sivun Yksi riviä avautuu kyseisen sivun hallintanäkymä. Tämän sivun vasemmassa laidassa on Content-valikko, josta avautuu sisällön muokkaustyökalu (kuva 17).



KUVA 16. Magenton asiakasnäkymien sisällönhallintalinkki



KUVA 17. Asiakasnäkymän sisältösivun muokkaustyökalu

Moduulin hallintasivulla voi käyttäjä myös siirtyä CMSNav-moduulilla luotujen sivujen muokkaamiseen hallintanäkymän alalaidassa (kuva 9) olevasta taulukosta, minne luodut sivut tulevat näkyville. Kun käyttäjä painaa sivua tästä taulukosta, siirtyy hän sisältösivujen muokkaustyökaluun (kuva 17).

Magenton sisältösivujen muokkaustyökalu on WYSIWYG-pohjainen (What You See Is What You Get), joten käyttäjä voi suoraan nähdä muokkauksiensa vaikutuksen eikä hänen tarvitse osata HTML-kuvauskieltä. Tämän editorin voi myös ottaa pois päältä Show / Hide Editor -painikkeesta (kuva 17), jolloin käyttäjä voi syöttää työkalun sisältökenttään HTMLää.

3.3.7 Moduulin toiminnan yhteenveto

Loppujen lopuksi moduulin käyttö ja sen tarkoitus on hyvin yksinkertainen. Käyttäjällä on alustava tarve luoda helposti sisältösivuja ja samalla navigaatiolinkit luomilleen sivuille asiakasnäkymän päänavigaation jatkeeksi.

Käyttäjä navigoi verkkokaupan hallintanäkymästä CMSNav-moduulin hallintasivulle mielessään sivu- ja navigaatiokokonaisuus haluamilleen sivuille ja lisää sivut alustavasti Create New Page ja Add -painikkeilla ExtJS-pohjaiseen hierarkiapuuhun.

Käyttäjän ollessa tyytyväinen hiearkiapuussa näkyvään sivu- ja navigaatiolinkkien kokonaisuuteen hän lisää sivut Magenton tietokantaan sekä CMSNav-moduulin tietokantatauluun Save-painikkeella. Tämän jälkeen sivut ja niiden navigaatiolinkit ovat nähtävissä Magenton asiakasnäkymän päänavigaation jatkeena.

Sivujen varsinaista sisältöä käyttäjä pääsee muokkaamaan Magenton omasta sisällönhallinnasta. Sivujen hierarkian ja pohjatietojen muokkaamiseen käyttäjä pääsee CMSNav-moduulin hallinnan kautta, missä hän voi muokata sivujen hierarkiaa muokkaamalla hierarkiapuun sivujen järjestystä siirtelemällä niitä hiirellä ja tämän jälkeen jälleen painamalla Save-painiketta, jolloin CMSNav-moduulin tietokantataulu päivittyy uusilla tiedoilla.

Poistaessaan sivuja ja niiden navigaatiolinkkejä käyttäjän tarvitsee vain valita moduulin hallintasivulta haluamansa sivu hierarkiapuusta ja painamalla Remove From Tree -painiketta. Tämän jälkeen hierarkiapuu päivittyy ja varsinaisesti poistaakseen sivun tietokannasta hän painaa hallintasivun Save-painiketta, jolloin CMSNav-moduulin tietokantataulu päivittyy.

Moduulin käytön lopputulos näkyy näiden toimintojen jälkeen Magenton asiakasnäkymän päänavigaatiossa ja näiden linkkien toiminnassa. Ohjatkoot ne sitten kaupan ulkopuolelle taikka kaupan sisäisesti linkitetylle sisältösivulle.

4 YHTEENVETO

Opinnäytetyön tavoitteena oli luoda moduuli Magento-pohjaisille verkkokaupoille. Moduulin tarkoitus on helpottaa verkkokauppan hallinnoitsijoiden työtä kaupan sisällönhallinnan ja navigaation osalta.

Työn yksi parhaimmista puolista oli päästä syventymään Magenton toiminnallisuuteen päivittäisten töiden ohessa. Moduulin kehityksessä pääsi hyvin syventymään Magenton hallinta- sekä kuluttajapuoleen.

Koska Magento on hyvin monimutkainen järjestelmä ja sen dokumentaatio on hyvin paljolti kolmansien osapuolien varassa, oli sille kehittäminen tämän työn haastavin osuus. Magentolle luodut eri sertifikaatit ja näiden koulutukset ovat pääasiassa alustan ensisijainen virallinen oppimislähde.

Lopuksi mainittakoon moduulin jatkokehityksestä. Tämän opinnäytetyön kirjoitushetkellä CMSNav-moduuli on vielä kehitysvaiheessa, joten moduulin toiminnallisuuksien loppuunvienti ja jatkokehitys siirtyvät työn tilaajalle, Piimega OY:lle.

LÄHTEET

1. (Magento™ creates huge success with enterprise e-commerce platform & community built on Zend Framework), Zend Framework. Saatavissa: http://www.zend.com/topics/Magento-CS.pdf. Hakupäivä 17.11.2015.
2. Open Source Initiative. Saatavissa: https://opensource.org. Hakupäivä 17.11.2015.
3. Magento License/Trademarks FAQs. 2016. Saatavissa: https://magento.com/legal/licensing. Hakupäivä 3.4.2016.
4. Zend Framework & MVC Introduction. Saatavissa: http://framework.zend.com/manual/1.12/en/learning.quickstart.intro.html. Hakupäivä 17.11.2015.
5. MVC arkkitehtuuri, Google Chrome. Saatavissa: https://developer.chrome.com/apps/app_frameworks. Hakupäivä 17.11.2015.
6. About. Zend Framework. Saatavissa: http://framework.zend.com/about/. Hakupäivä 17.11.2015.
7. ExtJS. Saatavissa: http://docs.sencha.com/extjs/6.0/. Hakupäivä 17.11.2015.
8. Git. Saatavissa: https://git-scm.com. Hakupäivä 17.11.2015.
9. Storm, Alan 2009. The Magento Config. Saatavissa: http://alanstorm.com/magento_config. Hakupäivä 19.11.2015.
10. Storm, Alan. Magento for Developers: Part 4- Magento layouts, blocks and templates. Saatavissa: http://devdocs.magento.com/guides/m1x/magefordev/mage-for-dev-4.html. Hakupäivä 10.1.2016.
11. Storm, Alan. Magento for Developers: Part 3 – Magento Controller Dispatch. Saatavissa: http://devdocs.magento.com/guides/m1x/magefordev/mage-for-dev-3.html. Hakupäivä 10.1.2016.