jamk.fi

WordPress-lisäosan kehitys

Viivi Järvinen

Opinnäytetyö
Toukokuu 2020
Tekniikan ala
Insinööri (AMK), tieto- ja viestintätekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Järvinen, Viivi	Opinnäytetyö, AMK	19.5.2020
	Sivumäärä 31	Julkaisun kieli Suomi
	-	Verkkojulkaisulupa myönnetty: x

Työn nimi
WordPress-lisäosan kehitys

Tutkinto-ohjelma
Tieto- ja viestintätekniikka

Työn ohjaaja(t)
Kari Niemi

Toimeksiantaja(t)
Great Slogan Oy

Tiivistelmä

Opinnäytetyön toimeksiantona oli toteuttaa tietojen listauslisäosa WordPress-julkaisujärjestelmälle. Lisäosan tarkoituksena olisi tukea verkkosivukehityksessä sellaisissa tapauksissa, joissa asiakayritys esimerkiksi myy tai vuokraa asuntoja.

Tehtävänä oli toteuttaa lisäosa, jonka avulla tulostetaan listauksena kohteiden halutut tiedot. Lisävaatimuksena oli myös, että loppukäyttäjän olisi mahdollisimman helppo muuttaa eri kohteiden kaikkia tietoja tarvittaessa. Erityisesti varaustilanteen tiedot tulisi voida muuttaa helposti.

Opinnäytetyössä tutustuttiin WordPressin tarjoamiin mahdollisuuksiin lisäosien kehittämisen suhteen ja näiden pohjalta toteutettiin lisäosa vastaamaan toimeksiantoa.

Lisäosa toteutettiin mukautettuja sisältötyyppejä käyttäen niin, että kohteille luotiin oma artikkelityyppi, jonka alle niitä on helppo lisätä. Artikkelityyppien sivupohjiin luotiin mukautettuja kenttiä keräämään tietoa. Kenttien kautta haettujen tietojen avulla luotiin listaus, jota kutsutaan lyhytkoodilla.

Mukautettujen kenttien ansiosta kohteiden tietojen muokkaaminen on yksinkertaista ja lyhytkoodin avulla haluttu listaus voidaan tulostaa helposti.

Lisäosan jatkokehitykselle on monenlaisia mahdollisuuksia, sillä toteutettu lisäosa ei itsessään sulje pois juuri mitään kehityksen vaihtoehtoja.

Avainsanat (asiasanat)

WordPress, Lisäosa, Kehitys, Mukautettu sisältötyyppi

Muut tiedot (salassa pidettävät liitteet)

jamk.fi

Description

Author(s)	Type of publication	Date
Järvinen, Viivi	Bachelor's thesis	19.5.2020
		Language of publication: Finnish
	Number of pages: 31	Permission for web publication: x

Title of publication
WordPress Plugin Development

Degree programme
Information and Communications Technology

Supervisor(s)
Niemi, Kari

Assigned by
Great Slogan Oy

Abstract

The assignment of the bachelor's thesis was to execute a plugin to list data for WordPress Content Management System. The purpose of the plugin is to support website development in cases where the client company, for example sells or rents apartments.

The assignment was to implement a plugin that allows the information of the objects to be printed as a list. An additional requirement was that it should be easy for the user to change all the information from different items.

In the thesis the plugin was implemented to match the assignment after getting to know about the opportunities offered by WordPress

The Plugin was implemented by creating custom post type to objects using custom content types. The custom fields were created in the template of the custom content type to gather data. The listing, called by shortcode, was created from the data gathered from the fields.

As a result, custom fields make it simple to edit objects information and the shortcode enables easy printing for listing-items.

There are many possibilities for the further development of the plugin, as the implemented plugin itself does not exclude almost any development options.

Keywords/tags (subjects)
WordPress, Plugin, Development, Custom Content Types

Miscellaneous (Confidential information)

Sisällys

Kuviot

1 Työn lähtökohdat

1.1 Taustaa ja toimeksiantaja

Opinnäytetyön toimeksiantajana toimi digitaalinen markkinointitoimisto Great Slogan Oy. Yritys keskittyy digimarkkinointiin ja toteuttaa samalla myös verkkosivuja. Sloganin toimistot sijaitsevat Jyväskylässä ja Helsingissä, mutta käytännössä se toimii yli verkon kaikkialla Suomessa. Asiakasyrityksiä Sloganilla on laidasta laitaan, niin suuria kuin pieniäkin, monenlaisilta eri aloilta.

Verkkosivut Sloganilla toteutetaan lähes poikkeuksetta WordPress-julkaisujärjestelmällä. Toteutuksissa apuna käytetään WordPressin hyväksi havaittuja lisäosia ja usein räätälöidyn kokonaisuuden aikaansaamiseksi sivujen teema luodaan itse. Sivuissa kiinnitetään huomiota käyttökokemukseen ja ne rakennetaan digimarkkinointia ajatellen yksilöllisesti ja tapauskohtaisesti.

1.2 Tehtävä ja tavoitteet

Sloganilla on useita asiakasyrityksiä, joiden liiketoimintaan kuuluu asuntojen, kiinteistöjen tai toimitilojen vuokraus ja/tai myynti. Tällaisissa tapauksissa asiakkailla on lähes aina samanlainen tarve saada verkkosivuille listattua vuokrattavat ja myytävät kohteet niin, että niiden vuokraus- tai myyntitilanne on helppoa ja yksinkertaista vaihtaa niiden muuttuessa.

Opinnäytetyön tavoitteena oli toteuttaa ns. Asuntolistaus-lisäosa WordPressille, jota voisi yleisesti käyttää yllä kuvatuissa asiakastapauksissa ja jota myös verkkosivujen loppukäyttäjän olisi helppoa käyttää, mikäli esimerkiksi kohteiden tiedoissa tapahtuu muutoksia. Ominaisuuksiltaan sopivaa lisäosaa ei toistaiseksi vielä tällaisiin tapauksiin löytynyt, joten sellaisen kehittäminen toimi tämän opinnäytetyön lähtökohtana. Opinnäytetyön teknisen toteutuksen oheessa tavoitteena oli laajentaa tietämystä WordPressin lisäosista ja niiden kehityksestä.

1.3 Soveltava kehittämistyö

Opinnäytetyö toteutettiin soveltavana kehitystyönä. Lähtökohtaisena ongelmana kehitystyölle oli oikeanlaisen WordPress-lisäosan puuttuminen. Tutkimusongelma lähdettiin ratkaisemaan tutkimalla lisäosakehitykseen liittyviä erilaisia lähteitä, suunnittelemalla niiden pohjalta lisäosan ja lopulta toteuttamalla suunnitellun lisäosan kehitystyön tuloksena.

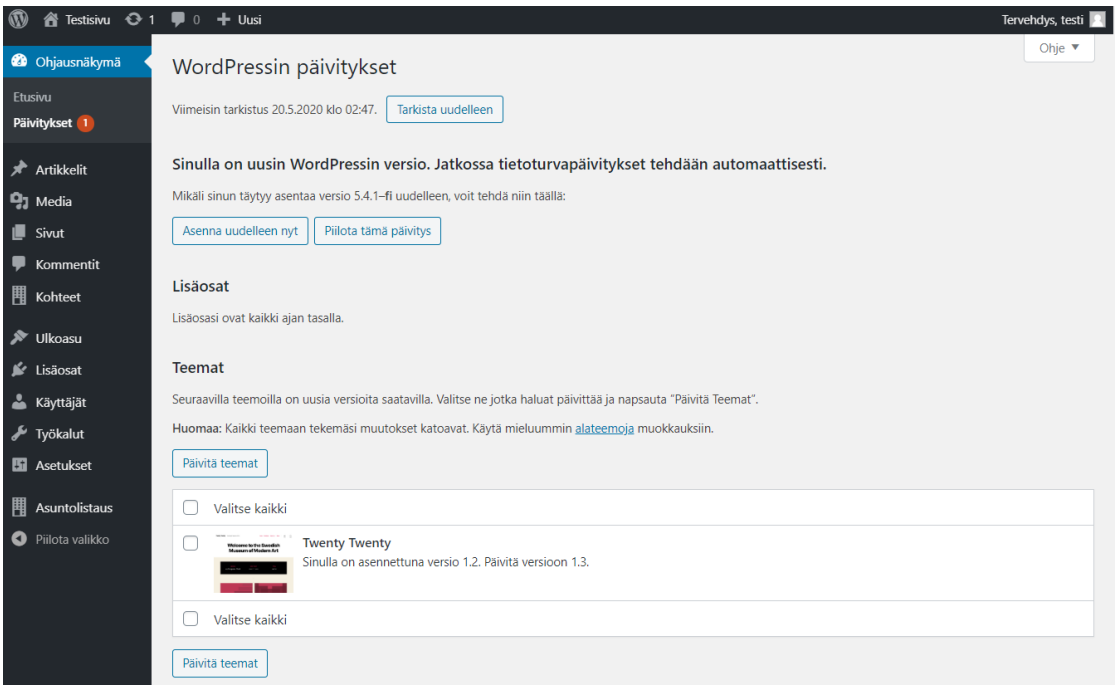
2 WordPress

2.1 Yleistä

WordPress on sisällönhallinta- ja julkaisujärjestelmä, jonka päällä toimii yli 35 % kaikista web-sivuista. Se perustuu avoimeen lähdekoodiin, on rakennettu PHP-ohjelmointikielellä ja käyttää MySQL-tietokantaa. WordPress sai alkunsa vuonna 2003 Mike Littlen ja Matt Mullenwegin toimesta ja avoimen lähdekoodin mahdollistamana kuka tahansa voi osallistua kehitystyöhön luomalla ja jakamalla uutta. (About n.d.)

2.2 WordPressin ominaisuuksia

WordPress on käyttäjälleen yksinkertainen ja joustava. Se on helppo asentaa ja sen päivittäminen onnistuu yhdellä napin painalluksella (ks. Kuvio 1). WordPress on saatavilla yli seitsemälläkymmenellä eri kielellä ja halutun kielen määrittäminen asennuksen yhteydessä on myös helppoa. (Features n.d.)



Kuvio 1. Päivitykset-sivu WordPressin ohjausnäkymässä

WordPressin avulla voi luoda millaisen sivuston tahansa ja se mahdollistaa verkkosivun julkaisun nopeasti ja helposti. Uusien sisältöjen lisääminen ja niiden hallitseminen onnistuu myös vaivattomasti. Sisältöjen julkaisun voi ajastaa toivotulle ajankohdalle, ne voidaan määritellä julkisiksi tai yksityiseksi ja ne voidaan halutessa suojata myös salasanalla. (Mt.)

Sivustolla voi olla useita käyttäjiä, jolla kaikilla ei välttämättä tarvitse olla mahdollisuutta muuttaa kaikkea sivulla. (Mt.) WordPress tarjoaa käyttäjäryhmiä, joille voidaan määritellä mahdollisia rajoitteita sivustoon liittyen. Näin voidaan varmistaa, ettei kukaan pääse vahingossa vaikuttamaan esimerkiksi ulkoasuun tai muihin tärkeisiin asetuksiin.

Kaikki WordPressin tuottama koodi on täysin W3C:n (World Wide Web Consortium) asettamien standardien mukaista. Tämän ansiosta WordPress-verkkosivustot toimivat eri selaimilla, mutta samalla säilyttävät yhteensopivuuden mahdollisuuden myös tulevaisuuden selaimille. Sitä toimiiko verkkosivu myös tulevaisuudessa, ei tarvitse jännittää. WordPress on myös valmiiksi hakukoneoptimoitu. (Mt.)

Avoimen lähdekoodin ja suosionsa ansiosta, WordPressin ympärille on kehittynyt aktiivinen yhteisö, joka kehittää ja ylläpitää WordPressiä. Yhteisöltä käyttäjä voi saada tukea sekä apua sivuston luomisessa ja muokkaamisessa mm. tukifoorumien, erilaisten blogiartikkeleiden ja tapaamisten kautta. Yhteisöön voi kuka tahansa liittyä mukaan avustamaan WordPressin ylläpidossa ja kehityksessä. (Mt.)

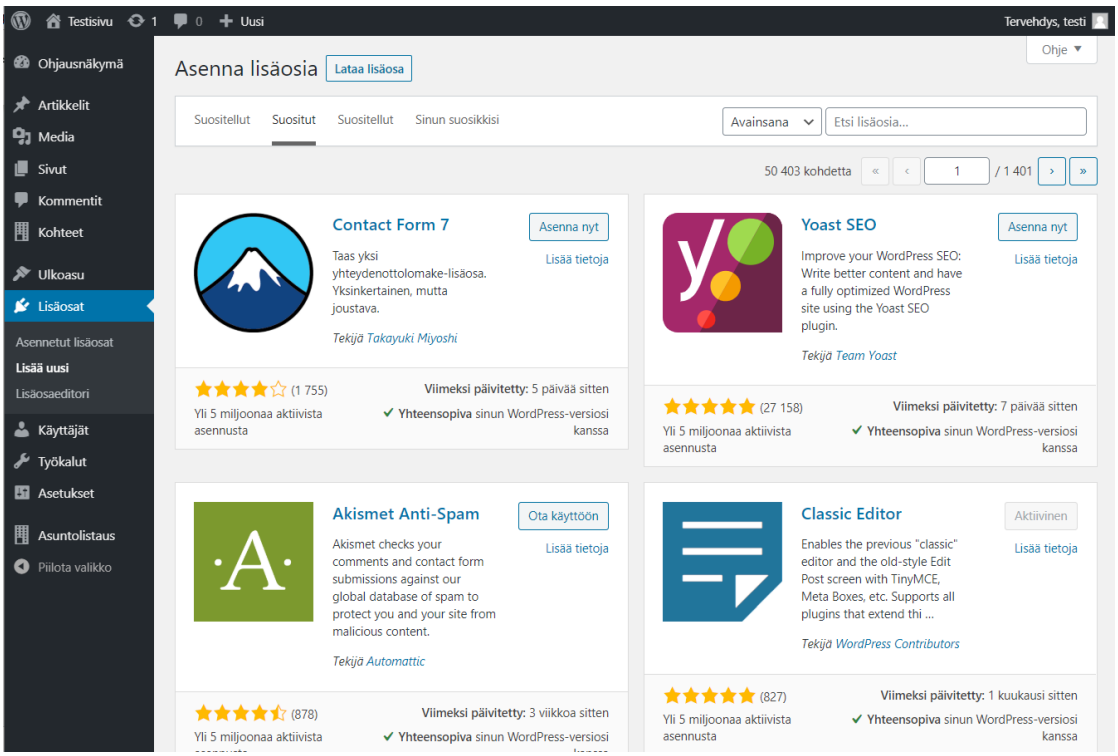
2.3 WordPressin ominaisuudet kehittäjälle

Vaikka WordPress on yksinkertainen käyttäjälle, kehittäjän näkökulmasta se on monitahoinen ja joustava. Koodia voi muokata ja laajentaa täysin vapaasti, eikä lisenssimaksuja ole edes kaupallisissa projekteissa. Kehittäjille tarjolla olevien valmiiden ominaisuuksien avulla Wordpressiä on mahdollista kehittää juuri haluttuun suuntaan. (Features n.d.)

2.3.1 Lisäosat

Lisäosat ovat koodipaketteja, jotka toimivat laajennuksina WordPressin ydintoiminnoille. Lisäosat vaikuttavat juuri näihin ydintoimintoihin muokkaamalla sitä tarvittavaan suuntaan ja aikaansaamalla näin haluttuja toimintoja ja muutoksia. (What Is Plugin n.d.)

Wordpressin omasta lisäosahakemistosta löytyy yli 56 000 valmista lisäosaa, joita kuka tahansa voi käyttää oman sivustonsa kehitykseen. Valmiin lisäosan hakeminen ja asentaminen on helppoa WordPressin ohjauspaneelin kautta (ks. Kuvio 2). Jos tarvittavia toimintoja mahdollistavaa lisäosaa ei valmiiksi löydy, voi lisäosia luoda myös itse avoimen lähdekoodin ansiosta. Wordpress tarjoaa ja ylläpitää käsikirjaa (Plugin Developer Handbook) lisäosien kehittäjälle, joka on ajankohtainen ja sisältää kaiken tarvittavan tiedon lisäosien kehityksestä. (Plugins n.d.)



Kuvio 2. Lisäosien asennussivu WordPressin ohjauspaneelissa

2.3.2 Teemat

Se miltä WordPress-verkkosivun ulkoasu näyttää, määritellään teeman avulla. WordPress tarjoaa hakemistossaan tuhansia ilmaisia teemoja ja mikäli sivustosta halutaan luoda yksilöllisempi, avoimen lähdekoodin ansiosta oman kustomoidun teeman tekeminen on myös mahdollista.

Teema määrittää sen, miten WordPressiin lisätty sisältö ja tieto esitetään sivulla. Luomalla oman teeman kehittäjä voi määritellä täysin, miten sisällöt asetellaan, miltä sivuston eri sivut näyttävät, mitä värejä tai kirjasimia käytetään ja kaikkea muuta mitä sivuston ulkoasuun kuuluu. Teeman kehittäjän tulee kuitenkin muistaa, että sivuston toiminnallisuuteen liittyviä määrityksiä ei teeman kautta kannata tehdä, tätä varten tulisi aina käyttää lisäosia. Mikäli sivuston ulkoasua halutaan joskus vaihtaa, tämä on helpointa ja järkevintä tehdä itse teemaa vaihtamalla. Jos sivuston toiminnallisuuksia on määritelty vanhassa teemassa, ne kaikki häviävät teeman mukana. (What Is Theme n.d.)

2.3.3 Sovelluskehys

Web-sovellusten tekoon WordPressillä on oma sovelluskehys. Kehys tarjoaa valmiina monia ominaisuuksia, esimerkiksi sovelluksen käyttäjätilien hallintaan liittyen, jolloin niitä ei tarvitse erikseen luoda. Lisäksi WordPressin monet ominaisuudet kuten lisäosat tarjoavat mahdollisuuksia lisätä toiminnallisuuksia niinsanotusti vain napin painalluksella. Tämä helpottaa ja nopeuttaa kehittäjien työtä. (Lema 2019.)

2.3.4 Kustomoidut sisältömallit

WordPress sisältää valmiina erilaisia malleja sisältöjen luomisen tueksi. Tällaisia ovat erilaiset artikkelityypit (post types) ja luokittelut (taxonomies). Valmiita artikkelityyppejä ovat esimerkiksi artikkelit, sivut ja liitännäiset ja luokitteluun kuuluvat mm. kategoriat ja tagit. Valmiiden mallien lisäksi WordPress tarjoaa kehittäjilleen mahdollisuuden luoda räätälöityjä sisältömalleja sivuston tarpeiden mukaan. Kustomoitujen mallien avulla kehittäjän on helpompi luoda yksilöllisiä verkkosivuja, joissa erilaiset sisällöt voidaan lajitella haluttuihin tyyppeihin ja näin ollen määritellä niille esimerkiksi haluttuja toimintoja ja ulkoasuja. (Farmer 2017.)

3 WordPress-lisäosien kehityksestä

3.1 Lisäosat

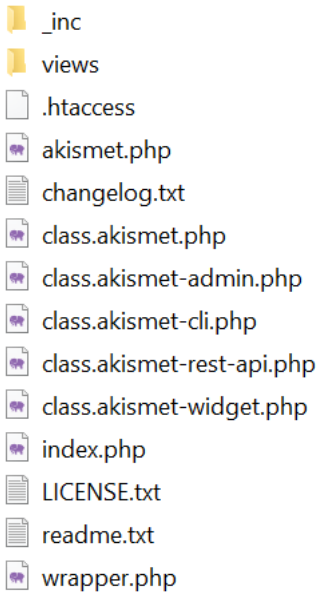
Lisäosien tarkoituksena on laajentaa verkkosivuston toimintaa niin, ettei Wordpressin ydintä muokata lainkaan. Koska opinnäytetyön tarkoituksena oli toteuttaa toimeksiantoa tukeva WordPress-lisäosa, tutustutaan tässä luvussa tarkemmin juuri lisäosien kehittämiseen. Lisäosat luodaan PHP-ohjelmointikieltä käyttäen, mutta ne voivat sisältää myös esimerkiksi JavaScript- tai CSS-tiedostoja.

3.2 Rakenne

Lisäosat sijaitsevat WordPressin wp-content/plugins -hakemistossa. WordPress sisältää valmiiksi kaksi lisäosaa, joiden nimet ovat Akismet ja Hello Dolly. Mikäli sivustolle

ei ole vielä asennettu muita lisäosia /plugins-hakemiston tulisi sisältää kaksi tiedostoa hello.php ja index.php sekä /akismet-hakemiston. Lisäosa voi yksinkertaisimmillaan olla vain yhden tiedoston sisältävä kuten Hello Dolly, toinen WordPressin oletuslisäosista, joka koostuu ainoastaan hello.php-tiedostosta. (Sabin-Wilson 2019.)

Lisäosa on rakenteeltaan tiedostopaketti, jonka rakenne kannatta pitää mahdollisimman selkeänä ja suunnitella huolella. Tämä auttaa välttämään ongelmia WordPressin ytimen kanssa ja helpottaa lähdekoodin tarkastelua. Lisäosan kehityksessä tuleekin aloittaa hakemiston luomisesta, eli luodaan kansio, joka sijoitetaan /plugins-hakemistoon. Hakemisto nimetään lisäosan mukaan, minkä jälkeen hakemiston sisälle luodaan samanniminen PHP-tiedosto. Esimerkiksi valmiina olevan Akismet-lisäosan hakemistosta (ks. kuvio 3) löytyy akismet.php-niminen tiedosto. Lisäosan PHP-tiedostoon täytyy lisätä ylätunniste (file header), jossa määritellään vähintään lisäosan nimi. Ilman ylätunnistetta WordPress ei tunnista lisäosaa lisäosaksi eikä näin ollen löydä sitä ollenkaan. (Sabin-Wilson 2019.) Esimerkki ylätunnisteesta kuviossa 4.



Kuvio 3. Akismet-lisäosan hakemiston rakenne

```php
<?php
/**
 * @package Akismet
 */
/*
Plugin Name: Akismet Anti-Spam
Plugin URI: https://akismet.com/
Description: Used by millions, Akismet is quite possibly the best way in the world to <strong>protect your
Version: 4.1.5
Author: Automattic
Author URI: https://automattic.com/wordpress-plugins/
License: GPLv2 or later
Text Domain: akismet
*/
```

Kuvio 4. Akismet-lisäosan ylätunniste

3.3 Koukut

WordPress tarjoaa kehittäjälleen funktioita, jotka mahdollistavat lisäosien vuorovaikutuksen WordPressin ytimen kanssa ja toiminnallisuuksien muutoksen kajoamatta itse ytimeen. Näitä funktioita kutsutaan koukuiksi (hooks). Koukkuja on kahdenlaisia: toiminnat (actions) ja suodattimet (filters). Koukkuja käyttääkseen kehittäjän tarvitsee luoda kustomoitu takaisinkutsufunktio (callback) koukulle tiettyä toimintaa tai suodatinta varten. (Plugin Basics n.d.)

Toimintakoukut mahdollistavat tiedon lisäämisen ja muutokset WordPressin toimintatapoihin. Ne vaikuttavat siihen, miten WordPressin koodi toimii sitä kuitenkaan muuttamatta. Toimintakoukkujen takaisinkutsufunktiot voivat suorittaa tehtäviä, kuten esimerkiksi tiedon lisääminen tietokantaan. Toimintakoukut eivät palauta mitään takaisin kutsuvalle koukulle. Kuviossa 5 esimerkki toimintakoukusta ja sen toiminnasta. Funktio ajetaan sen jälkeen kun ”init”-koukku toteutetaan. (Hooks n.d.)

```php
function test_callback() {
    // tähän toiminta
}
add_action( 'init', 'test_callback' );
```

Kuvio 5. Esimerkki toimintakoukusta

Suodatinkoukut antavat mahdollisuuden muuttaa jo olemassa olevien funktioiden dataa. Niiden takaisinkutsufunktiot muokkaavat haluttua muuttujaa ja palauttavat sen muokattuna. Ne siis suodattavat muuttujan ja palauttavat sen sitten koukulle tulevaa käyttöä varten. Kuvion 6 esimerkissä the_title on suodatin, jonka toteuttamisen jälkeen funktio ajetaan. Suodatinkoukkujen tulee toimia eristetyllä tavalla, eikä niiden käytöstä kuulu syntyä sivuvaikutuksia muihin toimintoihin. (Hooks n.d.)

```
5   function test_filter_title($title)
6   {
7       return 'The '. $title . ' was filtered';
8   }
9   add_filter('the_title', 'test_filter_title');
10
```

Kuvio 6. Esimerkki suodatinkoukusta

3.4 Asennus ja poisto

Lisäosaa luodessa kehittäjä tarvitsee kolme erilaista peruskoukkua (basic hooks). Nämä koukut ovat aktivointikoukku (activation hook), deaktivointikoukku (deactivation hook) ja asennuksen poistokoukku (uninstall hook). Näitä koukkuja kutsutaan, kun WordPressin hallintapaneelin kautta muutetaan lisäosan statusta. (Plugin Basics n.d..)

Kun lisäosa otetaan käyttöön, ajetaan aktivointikoukku ja kun lisäosa poistetaan käytöstä, deaktivointikoukku suoritetaan. Aktivointikoukun avulla voi esimerkiksi asettaa sivustolle uusia oletusarvoja, yliajaa joitakin WordPressin valmiita asetuksia tai lisätä tietokantatauluja. Deaktivointikoukulla voi puolestaan muun muassa tyhjentää välimuistin ja poistaa väliaikaisia tiedostoja, joita on tallennettu lisäosaan. (Activation / Deactivation Hooks n.d.)

Kun lisäosan asennus poistetaan kokonaan, halutaan samalla poistaa lisäosan luoma sisältö myös. Tällaisia sisältöjä ovat esimerkiksi tietokantaan luodut taulut. Lisäosan

poistamiseen kehittäjällä on mahdollista valita kahdesta eri poistometodista (Uninstall Methods). Ensimmäinen vaihtoehto on poistokoukku, joka toimii samoin kuin aktivointi- ja deaktivointikoukku. Toinen vaihtoehto on käyttää uninstall.php-tirdostoa. Tiedosto tulee sijoittaa lisäosa-hakemiston juureen ja se ajetaan automaattisesti, kun lisäosa poistetaan WordPress-asennuksesta. (Uninstall Methods n.d.)

3.5 Rajapinnat

WordPress tarjoaa kehittäjilleen myös useita ohjelmointirajapintoja (Application Programming Interface, API), jotka auttavat yksinkertaistamaan koodia lisäosaa luotaessa (Plugin Basics n.d.). Ohjelmointirajapinnat tarjoavat toimintoja valmiina kehittäjän käyttöön niin, ettei kaikkea tarvitse kirjoittaa alusta asti itse.

3.5.1 Plugin API

Plugin API mahdollistaa lisäosakehityksen. Se tekee mahdolliseksi liittää WordPressin ytimen rinnalle toiminnallisuuksia. Liittäminen tapahtuu luvussa 3.3 käsiteltyjen koukkujen avulla. Lisäosan ja ytimen koukut vuorovaikuttavat toistensa kanssa ”tarttumalla” toisiinsa. (Plugin API n.d.)

3.5.2 Settings API

Settings API antaa kehittäjälle mahdollisuuden määrittää asetussivuja ja mahdollistaa asetuslomakkeita sisältävien sivujen puoliautomaattisen hallinnan. Rajapinnan avulla voidaan rekisteröidä asetussivuja sekä niiden sisällä olevia osioita ja kenttiä samanaikaisesti. Myös olemassa olevien asetussivujen sisälle voidaan rekisteröidä uusi osioita ja kenttiä. Vaikka rajapinta helpottaa kehittäjän työtä paljon, vaatii rekisteröinnin ja kenttien validointi kuitenkin ponnisteluja. (Settings API n.d.)

3.5.3 Shortcode API

Shortcode API on yksinkertainen rajapinta WordPress-lyhytkoodien luomiseen. Lyhytkoodeja voidaan käyttää sekä artikkeleissa, että sivuilla. Mahdollistamalla lyhytkoodien lisäämisen, rajapinta antaa kehittäjälle valmiudet luoda erilaisia sisältöjä ja

lisätä ne helposti sivustolleen. API sisältää apufunktioita oletusominaisuuksien asettamiseen sekä hakemiseen ja tukee sekä itsestään sulkeutuvia että sulkevia lyhytkoodeja. (Shortcode API n.d.)

3.6 Tietoturva

Tietoturvasta huolehtiminen lisäosan kehityksessä on erityisen tärkeää. Mikäli lisäosan tietoturvassa esiintyy aukkoja, ne voivat vaarantaa koko sivuston turvallisuuden.

3.6.1 Käyttäjäoikeudet

Tärkeintä tietoturvan kannalta on käyttäjäoikeuksien tarkastaminen. Jokaisella käyttäjällä on käyttäjärooli, joka määrittää sen, millaiset oikeudet käyttäjällä on, eli mitä hän voi tehdä WordPressin ohjauspaneelissa. Lisäosaa rakentaessa tuleekin kiinnittää huomiota näihin rooleihin ja miettiä, halutaanko kaikille käyttäjärooleille antaa pääsy kaikkiin lisäosan toimintoihin, vai halutaanko niitä rajoittaa jotenkin. (Checking User Capabilities n.d.)

3.6.2 Tiedon validointi

Kaiken lisäosan ulkopuolelta tulevan datan tarkastaminen on tietoturvan kannalta tärkeää. Validointi on prosessi, jossa ennalta määrätyn mallin mukaisesti analysoidaan, onko tieto kelvollista vai ei. Validointi on tärkeää, sillä vääränlaisen tiedon tallentaminen tietokantaan voi vaikuttaa epätoivotusti sivuston toimintaan. Validointi tulisi tapahtua, ennen kuin dataa käytetään toimintojen suorittamiseen. Validoinnissa voi käyttää valmiita PHP-funktioita tai WordPressin tarjoamia sopivia funktioita, joita on useita. Funktioita voi myös kirjoittaa itse PHP.llä tai JavaScriptillä. (Data Validation n.d.)

3.6.3 WordPress Noncet

Noncet ovat numeroista generoituja sarjoja, joilla tarkistetaan tulevien pyyntöjen alkuperä ja tarkoitus. Yhtä Noncea voidaan käyttää vain yhden kerran. Mikäli lisäosa

antaa käyttäjille mahdollisuuden lähetää tietoja, on varmistuttava siitä, että käyttäjä on juuri se, joka väittää olevansa, ja että hänellä on tarvittavat oikeudet kyseisen toiminnan suorittamiseen. Koska molemmat tiedot tarkastetaan samanaikaisesti, nonceilla voidaan myös varmistaa, että käyttäjän on tarkoitus suorittaa kyseinen toiminto eikä se tapahdu vahingossa. Myös nonceja varten WordPressillä on tarjota kehittäjilleen valmiita funktioita. (Nonces n.d.)

4 CASE: Asuntolistaus-lisäosan kehitys

4.1 Työn vaatimukset

Opinnäytetyön toimeksiannossa tärkeimmät lähtökohdat lisäosan kehitykseen olivat kohteiden listaaminen sekä yksittäisten kohteiden varaustilanteen muuttaminen, niin että se olisi myös loppukäyttäjälle helppoa. Listauksessa tulisi näkyä halutut tiedot kohteesta ja etenkin se, onko kohde sillä hetkellä saatavilla vuokrausta tai ostoa varten. Lisäksi asiakasyrityksen sivustosta vastaavan henkilön tulisi olla helppoa lisätä tai poistaa kohteita ja muuttaa niiden statusta.

4.2 Suunnittelu

Lisäosan suunnittelussa huomioitiin toimeksiantajan omat käytänteet asiakkaiden verkkosivujen toteutuksessa sekä yrityksen sellaiset asiakastapaukset, joissa kyseisestä lisäosasta olisi hyötyä. Koska toimeksiantajan puolelta verkkosivukehityksessä panostetaan sivujen yksilöllisyyteen, niiden toteutuksessa käytetään lähes poikkeuksetta kustomoituja sisältötyyppejä. Jotta lisäosa tukisi mahdollisimman hyvin muuta kehitystä, myös sen toteutuksessa olisi hyvä käyttää samankaltaisia menetelmiä.

Lisäosa suunniteltiin toteutettavaksi niin, että se aktivoitaessa luo oman artikkelityypin (Custom Post Type) kohteille. Artikkelityypin alle voidaan lisätä kohteita, minkä jälkeen lisäosa hakee kaikkien kohteiden halutut tiedot ja listaa ne määrätylle sivulle. Listauksen tulostamiseksi päätettiin käyttää lyhytkoodia (Shortcode), jonka avulla listaus voidaan lisätä helposti haluttuun paikkaan sivustolla.

4.3 Toteutus

Lisäosa oli suunnitteluvaiheen aikana kulkenut työnimellä ”Asuntolistaus”, ja koska lisäosan nimi tulisi aina olla uniikki mahdollisen julkaisemisen vuoksi, tarkastettiin WordPressin lisäosien valikoimasta, löytyykö ennestään sen nimistä lisäosaa. Sellaista ei valmiina löytynyt, joten lisäosa nimettiin Asuntolistaukseksi.

4.3.1 Lisäosan luonti

Lisäosan kehitystyö alkoi teoriaan tutustumalla. WordPressin oma kattava dokumentaatio tarjosi apua ja sen avulla saatiin luotua lisäosan runko, kuten luvussa 3.1 kuvataan, ja perustoiminnot kuten aktivointi- ja deaktivointikoukut. Poistometodeista päädyttiin käyttämään uninstall.php-tiedostoa.

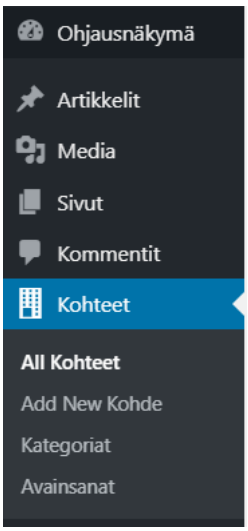
4.3.2 Kustomoidun artikkelityypin luonti

Kun lisäosan runko oli kasassa, oli aika lähteä toteuttamaan kohdelistausta. Ensimmäiseksi luotiin kohteille oma yksilöllinen artikkelityyppi, jonne niiden tiedot voidaan tallentaa. Tarkoituksena on, että lisäosa luo artikkelityypin heti, kun se aktivoidaan. Tämä varmistetaan init-toimintakoukulla add_action()-funktiossa. Artikkelityyppi toteutettiin funktiolla, jolla kutsutaan register_post_type()-funktiota, joka luo artikkelityypin (ks. kuvio 7). Käytetty register_post_type()-funktio ei luo uutta taulua tietokantaan, vaan artikkelityypin alle lisätyt kohteet tallennetaan jo valmiiseen wp_posts-tauluun WordPressin tietokantaan.

```php
// Luodaan Custom Post Type
function alistaus_create_post_type() {

    $labels = array(
        'name' => 'Kohteet',
        'singular_name' => 'Kohde',
        'add_new' => 'Add New Kohde',
        'add_new_item' => 'Add New Kohde',
        'edit_item' => 'Edit Kohde',
        'new_item' => 'New Kohde',
        'all_items' => 'All Kohteet',
        'view_item' => 'View Kohde',
        'search_items' => 'Search Kohteet',
        'not_found' => 'No Kohteet Found',
        'not_found_in_trash' => 'No Kohteet found in Trash',
        'parent_item_colon' => '',
        'menu_name' => 'Kohteet',
    );
    // rekisteröidään artikkelityyppi
    register_post_type( 'kohde', array(
        'labels' => $labels,
        'has_archive' => true,
        'public' => true,
        'menu_icon' => 'dashicons-building',
        'supports' => array( 'title', 'editor', 'thumbnail','page-attributes' ),
        'taxonomies' => array( 'post_tag', 'category' ),
        'exclude_from_search' => false,
        'capability_type' => 'post',
        'rewrite' => array( 'slug' => 'kohde' ),
        )
    );
}
add_action( 'init', 'alistaus_create_post_type' );
```

Kuvio 7. Kuva artikkelityypin luovasta koodista

Kuten kuviosta 8 nähdään, artikkelityypille luodaan rekisteröinnin yhteydessä oma valikkokohde, joka näkyy WordPressin ohjausnäkymän sivuvalikossa. Valikon kautta käyttäjä pääsee helposti lisäämään uusia kohteita.



Kuvio 8. Luotu artikkelityyppi ohjausnäkymän valikossa

4.3.3 Mukautetut kentät

Seuraavaksi tuli aika päättää, miten tarvittava tieto tallennetaan kohteiden-artikkeli-sivuille ja miten sitä sieltä haetaan. Koska listaukseen halutaan useita yksittäisiä tietoja kaikista kohteista, päädyttiin käyttämään mukautettuja kenttiä (Custom Fields) tietojen käsittelyyn. WordPress mahdollistaa tietyssä määrin mukautettujen kenttien lisäämisen artikkeleihin, mutta tarjottu alusta ei ole niin ketterä erilaisten kenttien kanssa kuin olisi toivottavaa Asuntolistaus-lisäosan kannalta.

Toinen, parempi vaihtoehto, on käyttää hyväksi jo valmiina olevaa lisäosaa kenttien luomiseen. Advanced Custom Fields-lisäosa antaa mahdollisuuden käyttää itseään runkona lisäosa- tai teemakehitykselle, jolloin sen tarjoamat useat valmiit kentät ovat kehittäjän käytössä. Advanced Custom Fields (ACF) -lisäosasta on olemassa ilmainen versio, sekä maksullinen Pro-versio. Tässä tapauksessa käytettiin ilmaista versiota, jota saa käyttää aivan vapaasti. ACF-lisäosa sisällytettiin kehitteillä olevaan lisäosaan lisäämällä tiedostot includes-kansioon ja ajamalla kuvion 9 näyttämät funktiot.

```php
// Määritellään polku ja URL ACF-lisäosalle.
define( 'MY_ACF_PATH', plugin_dir_path( __FILE__ ) . 'includes/acf/' );
define( 'MY_ACF_URL', plugin_dir_url( __FILE__ ) . 'includes/acf/' );

// Sisällytetään ACF-lisäosa.
include_once( MY_ACF_PATH . 'acf.php' );


// Muokkaa URL-asetusta virheellisten URL-osoitteiden korjaamiseksi.
add_filter('acf/settings/url', 'my_acf_settings_url');
function my_acf_settings_url( $url ) {
    return MY_ACF_URL;
}

// Näytä tai piilota ACF admin-menu
add_filter('acf/settings/show_admin', 'my_acf_settings_show_admin');
function my_acf_settings_show_admin( $show_admin ) {
    return false;
}

```

Kuvio 9. Advanced Custom Fields-lisäosan sisällyttäminen

Kun ACF-lisäosa oli sisällytetty mukaan, täytyi seuraavaksi lisätä itse kentät tietojen keräämiseksi ja ohjata ne näkymään luodun artikkelityypin mallipohjassa. Tämä tapahtui luomalla ensin kenttäryhmä nimeltä Kohde (ks. kuvio 10), johon luotiin halutut kentät. Lopuksi kentät ohjattiin Kohteet-artikkelityypin artikkelipohjiin (ks. kuvio 11).

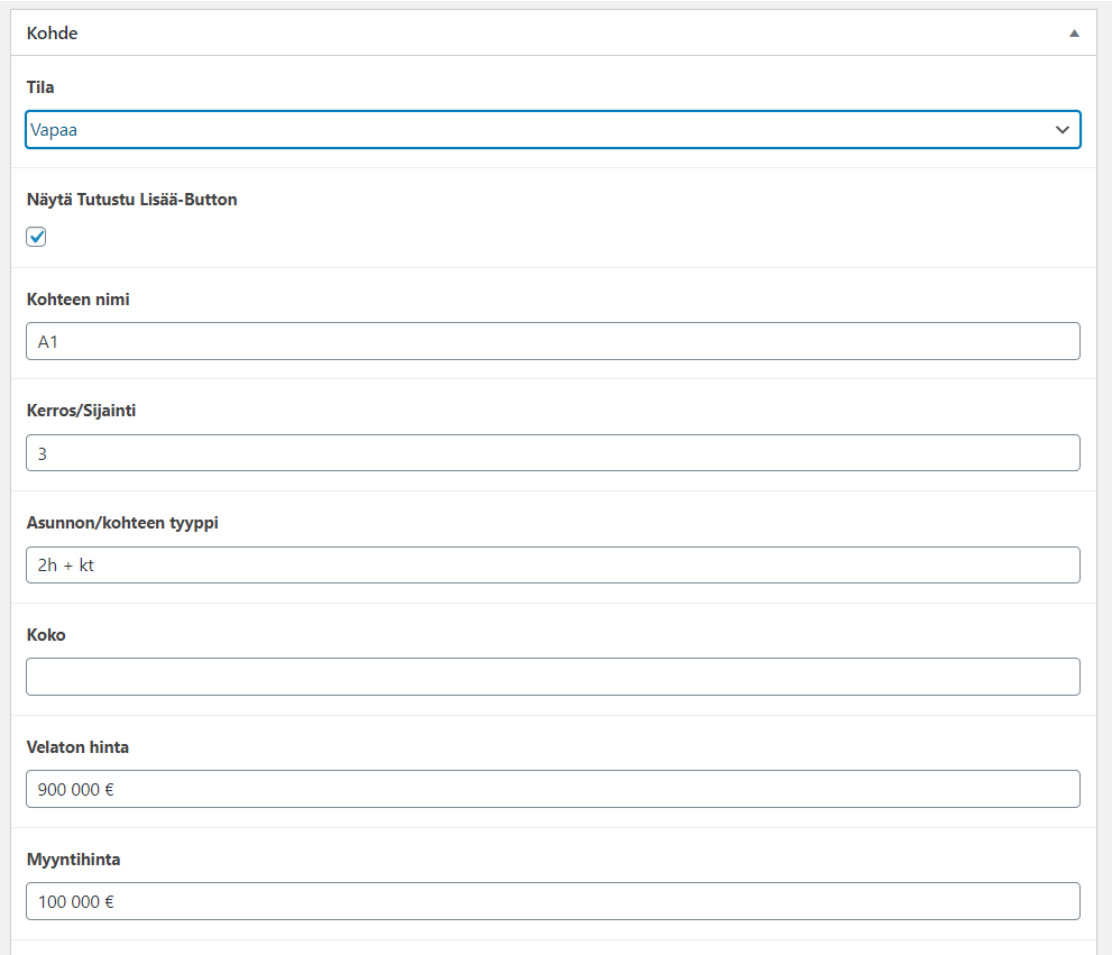
```php
// Lisätään halutut ACF-kentät luodun artikkelityypin sivupohjaan.
if( function_exists('acf_add_local_field_group') ):

    acf_add_local_field_group(array(
        'key' => 'group_1',
        'title' => 'Kohde',
        'fields' => array (

```

Kuvio 10. Kenttäryhmän luonti

```
347             array (
348                 'key' => 'field_form_alistaus',
349                 'label' => 'Lomake',
350                 'name' => 'lomake',
351                 'type' => 'wysiwyg',
352             ),
353 
354         ),
355         'location' => array (
356             array (
357                 array (
358                     'param' => 'post_type',
359                     'operator' => '==',
360                     'value' => 'kohde',
361                 ),
362             ),
363         ),
364     ));
365 
366     endif;
```

Kuvio 11. Acf-kentän luominen ja kenttäryhmän ohjaaminen oikeaan sijaintiin

Kenttiä luodessa yritettiin ottaa huomioon erilaiset asiakastapaukset, jossa lisäosaa voisi käyttää, sekä mahdollisuuden verkkosivuston kehittäjälle, tehdä jokaiselle kohteelle oma sivu. Sen vuoksi luotiin myös sellaisia kenttiä, joiden tietoja kohdesivujen olisi hyvä sisältää, mutta joita ei itse listauksessa käytetä. Kuviossa 12 näkymä yksittäisen kohteen muokkaussivusta.



Kuvio 12. Kohteen muokkaussivun kenttiä

4.3.4 Kohteiden listaus

Kenttien luonnin jälkeen siirryttiin kohteiden tietojen listaukseen. Listauksen tulostamista varten luotiin TulostaListaus()-funktio, joka hakee halutut tiedot jokaisesta Kohteet-artikkelityyppiin lisätystä kohteesta (ks. kuvio 13).

```
// Tulostetaan listaus ja luodaan lyhytkoodi
function TulostaListaus() {
        $args = array(
            'post_type'      => 'kohde',
            'sort_by' => 'ASC',
        );
        $loop = new WP_Query($args);
        while ( $loop->have_posts() ) {
            $loop->the_post();
            ?>
```

Kuvio 13. Haetaan artikkelityypin kohteet

TulostaListaus()-funktion sisällä haetaan myös syötetyt tiedot kentistä. Kenttien sisältämä data haetaan kuviossa 14 esitetyllä tavalla niin, että mikäli kenttä on jätetty tyhjäksi, sitä ei tulosteta ollenkaan. Näin lisäosa toimii joustavammin erilaisissa tapauksissa, kun jokaista kenttää ei ole pakko käyttää.

```
<?php if( get_field('kohde') ): ?>
    <div class="listaa kohde"><p><?php echo the_field('kohde'); ?></p></div>
<?php endif; ?>
```

Kuvio 14. Haetaan ja tulostetaan kenttään lisätty tieto

Tämän jälkeen luotiin lyhytkoodi, joka mahdollistaa listauksen näyttämisen siellä, missä kehittäjä haluaa sen esiintyvän. Lyhytkoodi kutsuu luotua TulostaListaus()-funktiota (ks. kuvio 15).

```
}
add_shortcode('asuntolistaus', 'TulostaListaus');

```

Kuvio 15. Lyhytkoodin luonti

Lyhytkoodin avulla tulostettua listausta on muotoiltu mahdollisimman vähän valmiiksi, jotta sen voi sisällyttää helposti verkkosivustoille ja sen tyylittely voidaan toteuttaa sivuston tyylien mukaan. Kuviossa 16 listaus on tulostettu lyhytkoodia

käyttäen tyhjälle sivulle. Tilan vaihtoehdot on kuviossa korostettu eri väreillä havainnollistamistarkoituksessa.

A 2	3	2h + kt	900 000 €	100 000 €	100 000 €	250 €	Vuokrattu	Tutustu lisää
A 8	1	2h + kt	700 000 €	500 000 €	100 000 €	2560€	Myyty	
A 11	1	2h + kt	700 000 €	500 000 €	100 000 €	650€	Varattu	
A 11	3	2h + kt	900 000 €	100 000 €	100 000 €	250€	Vapaa	Tutustu lisää

Kuvio 16. Kohdelistaus

4.3.5 Asetussivu

Vaikka lisäosa ei varsinaisesti vaatinut asetussivua, päätettiin se kuitenkin toteuttaa. Mahdollista jatkokehitystä ajatellen on hyvä, jos sivu on valmiiksi olemassa ja tämänhetkisessä tilanteessa se käy hyvin ohjesivustona. Asetussivu luotiin ja tallennettiin Settings API-rajapinnan avulla. Kuviossa 17 näkyvillä asetussivun luontiin käytetyt funktiot.

```
105  // Luodaan lisäosalle yksilöity asetus-valikko
106  add_action('admin_menu', 'asuntolistaus_create_menu');
107
108  function asuntolistaus_create_menu() {
109
110      // Luodaan uusi päätason valikko
111      add_menu_page('Asuntolistaus', 'Asuntolistaus', 'administrator', __FILE__, 'asuntolistaus_settings_page' ,
112
113      // Kutsutaan asetusten rekisteröinti-funktiota
114      add_action( 'admin_init', 'register_asuntolistaus_settings' );
115  }
116
117
118  function register_asuntolistaus_settings() {
119      // rekisteröidään asetukset
120      register_setting( 'asuntolistaus-settings-group', 'new_cpt_name' );
121      register_setting( 'asuntolistaus-settings-group', 'singular_cpt_option' );
122
123  }
```

Kuvio 17. Asetussivun lisääminen

Seuraavaksi luotiin itse asetussivun sisältö (ks. kuvio 18). Sivulle sisällytettiin tiedoksi listauksen lyhytkoodi ja luotujen acf-kenttien arvot ja kenttätyypit. Ohjeistuksen tarkoituksena on mahdollisesti helpottaa verkkosivuston kehittäjän työtä, sillä tarvittava tieto lyhytkoodista ja apu kohdesivun tekemiseen löytyvät samasta paikasta.

```
125  // Asetukset-sivun sisältö
126  function asuntolistaus_settings_page() {
127  ?>
128  <div class="wrap">
129      <div class="alistaus_asetukset">
130          <h1>Asuntolistaus-lisäosa</h1>
131          <div class="alistaus_lyhytkoodit">
132              <h2>Shortcode:</h2>
133              <p><strong>[asuntolistaus]</strong></p>
134              <p>Lisää lyhytkoodi haluamallesi sivulle, kohdelistauksen tulostamiseksi</p>
135          </div>
136
```

Kuvio 18. Asetussivun sisällön luominen

5 Tulokset ja jatkokehitys

5.1 Tulokset

Opinnäytetyön tuloksena saatiin toteutettua WordPress-lisäosa, joka listaa kohteiden tietoja. Listaus tulostaa kaikkien lisättyjen kohteiden ne tiedot, jotka kohteille määritellään. Lyhytkoodin ansiosta listaus voidaan joustavasti sijoittaa sivustolla minne vain.

Lisäosan toteutuksen tavoitteissa oli tärkeää myös, että kohteen varaustilanteen vaihtaminen olisi myös loppukäyttäjälle helppoa ja tässä onnistuttiin hyvin valintalistaus-kenttää käyttämällä. Myös kohteen muidenkin tietojen vaihtaminen ja päivittäminen, sekä uusien kohteiden lisääminen on yksinkertaista.

Valmis lisäosa auttaa verkkosivun kehittäjää nopeuttamalla ja helpottamalla itse kehitystyötä. Lisäosan luoma artikkelityyppi ja sen lisäkentät, mahdollistavat kohteiden tietojen käytön myös muualla sivustolla. Jokaiselle kohteelle voidaan luoda oma sivu, jonne voidaan hakea tiedot kohteille luodusta kenttäryhmästä. Näin ollen tietoja ei tarvitse syöttää useaan kertaan verkkosivustolle. Tämän huomioitiin lisäosan kehitystyössä luomalla myös sellaisia kenttiä, joita ei listauksessa käytetä, mutta joiden mahdollistamia tietoja kohteiden yksittäisille sivuille varmasti haluttaisiin. Tällaisia kenttiä ovat esimerkiksi kuva ja kuvailuteksti.

Lisäosan kehityksessä kiinnitettiin huomiota myös siihen, että sitä voisi käyttää mahdollisimman monipuolisesti erilaisissa tapauksissa. Kohteista halutaan listata erilaisia tietoja esimerkiksi sen mukaan, ovatko ne myynnissä vai tarjolla vuokralle. Tämä huomioitiin toteutuksessa niin, että listaus tulostaa vain sellaiset kentät, joihin on lisätty tietoa. Näin ollen sivuston kehittäjä voi määrittää, mitä tietoja listauksessa näytetään jättämällä tyhjäksi ne kentät, joita listaukseen ei haluta.

Opinnäytetyön tavoitteena oli myös kartuttaa tietoa ja osaamista lisäosakehityksessä. Lisäosan toteutuksen ohessa tuli etsittyä, tutkittua ja luettua paljon aiheeseen liittyvää tietoa monista erilaisista lähteistä ja tietotaito lisääntyi samalla melkein huomaamatta. Opinnäytetyön puitteissa tehdyssä tutkimuksessa käsitys WordPressin monipuolisuudesta kasvoi. Avoimen lähdekoodin ansiosta WordPressiin liittyvä kehitystyö on hyvin joustavaa ja valmiiden ominaisuuksien huomattava määrä auttaa kehittäjää tehostamaan työtään.

5.2 Jatkokehitys

Tuloksena saadun lisäosaa olisi mahdollista kehittää entistä monipuolisempaan suuntaan. Nyt lisäosa luo vain yhden kustomoidun artikkelityypin, mutta haluttaessa sitä voisi laajentamaan niin, että artikkelityyppejä pystyisi esimerkiksi asetussivun kautta luoda lisää tarvittaessa. Samalla jokaiselle artikkelityypille voitaisiin luoda oma yksilöllinen lyhytkoodi, jolloin niiden kohdelistaukset voisi eritellä halutulla tavalla. Lisäksi lisäosaa voisi laajentaa esimerkiksi hakemaan tiedot kohdesivulle jostain sivuston ulkopuolelta, niin ettei niitä tarvitsisi manuaalisesti lisätä sivustolle.

Myös yksi jatkokehityksen paikka voisi olla Advanced Custom Field-lisäosan päivittäminen maksulliseen Pro-versioon. Maksullinen versio tuo mukanaan ketteryyttä monipuolisemmilla kentillä sekä esimerkiksi Options-sivun ominaisuutena. Options-sivun luonti acf:n avulla mahdollistaisi lisäkenttien käytön myös kyseisellä asetussivulla. Jatkokehityksessä kannattaa kuitenkin ottaa huomioon myös se, että maksullisen ACF Pro-lisäosan mukana tulee enemmän rajoitteita kuin ilmaisen version kanssa. Pro-versiota ei saa käyttää ilmaiseksi jaettavissa lisäosissa, kuten ilmaista, vaan tuotettua lisäosaa voi jakaa vain maksullisena. Joten, jos jatkokehityksen oheessa

suunnitelmissa on lisäosan julkaisu ja sen muille WordPress-käyttäjille jakaminen, kannattaa maksullisen version käyttöä harkita kunnolla.

Lisäosan jatkokehitykselle löytyy useita erilaisia mahdollisuuksia, sillä se ei olemassa olevilla toiminnoillaan juurikaan sulje vaihtoehtoja ulkopuolelle. Koska lisäosa on kuitenkin kehitetty toimeksiantajan kohtaamia tietynlaisia asiakastapauksia varten, olisikin viisainta kartoittaa jatkokehitystarpeita tällaisten asiakastapausten kautta ja kysyä ideoita ja mielipiteitä mahdollisesti myös asiakkailta.

6 Pohdinta

Opinnäytetyön tuloksena syntynyt lisäosa toteuttaa sille asetetut vaatimukset ja sen kehityksen ansiosta WordPressin lisäosiin liittyvä ymmärrys kasvoi myös huomattavasti. Vaikka itse lisäosa on loppujen lopuksi aika yksinkertainen, sen toteuttamisessa oli kuitenkin perehdyttävä syvemmin koko lisäosakehitykseen.

Toimeksiantoa lähdettiin toteuttamaan ilman aikaisempaa kokemusta lisäosien kehityksestä, tutustumalla erilaisiin lisäosista ja niiden kehityksestä kertoviin lähteisiin. Lisäksi pohdittiin toimeksiannon tärkeimpiä tavoitteita ja niiden toteutusta lisäosakehityksen näkökulmasta. Tältä pohjalta suunniteltiin lisäosan rakenne ja toiminta.

Suunnitteluvaiheessa keskityttiin toimeksiannon vaatimuksiin ehkä jopa liian kapeakatseisesti, eikä näin ollen toteutussuunnitelma ollut niin monitahoinen kuin se olisi voinut olla. Tämä johti siihen, että kehitystyön loppuvaiheilla, kun ymmärrys lisäosista kasvoi, alkoi kehittymään ideoita, joiden pohjalta koko lisäosan rakentaminen olisi ollut ehkä parempi aloittaa. Onneksi näitä ideoita on kuitenkin mahdollista hyödyntää haluttaessa jatkokehityksen parissa. Toteutettu lisäosa ei sulje näitä mahdollisuuksia pois.

Lisäosan kehityksessä otettiin hyvin huomioon erilaiset mahdollisuudet tapauksissa, joissa sitä voidaan käyttää. Mahdollisimman vähäinen tyylittely ja listauksen joustavuus helpottavat listauksen sisällyttämistä eri sivustoille. Lisäksi ylimääräiset kentät, joita listauksessa ei käytetä, olivat hyvä ja kehittäjää helpottava lisäys.

Itse lisäosan toteutus alkoi hitaasti rajallisen kehitystietämyksen vuoksi, mutta yrityksen ja erehdyksen kautta lisäosa alkoi valmistua. Kehitystyön tärkeimpänä päämääränä oli tuottaa listaus, johon haetaan määrättyjen kohteiden halutut tiedot ja siinä onnistuttiin. Samoin onnistuttiin myös vaatimuksessa tietojen muuttamisen helppoudesta. Eli vaikka suunnitteluvaihe tietyllä tapaa epäonnistuikin, lopputuloksen osalta päästiin kuitenkin asetettuihin tavoitteisiin.

Lähteet

About. N.d. WordPressin kehittäjille suunnattu verkkosivusto. Viitattu 27.4.2020 https://wordpress.org/about/.

Activation / Deactivation Hooks. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/plugin-basics/activation-deactivation-hooks/.

Checking User Capabilities. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/security/checking-user-capabilities/.

Data Validation. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/security/data-validation/.

Farmer, J. 2017. Creating Custom Content in WordPress: Custom Post Types. Blogiartikkeli austraalialaisen WordPress hosting-yrityksen Wpmu Devin verkkosivustolla. Viitattu 29.4.2020. https://premium.wpmudev.org/blog/creating-content-custom-post-types/

Features. N.d. WordPressin kehittäjille suunnattu verkkosivusto. Viitattu 27.4.2020 https://wordpress.org/about/features/.

Hooks. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/hooks/.

Lema, C. 2019. Pitching WordPress As An Application Framework. Artikkeli WordPressin sovelluskehyksestä kirjoittajan omalla verkkosivustolla. Viitattu 28.4.2020. https://chrislema.com/pitching-wordpress-as-an-application-framework/.

Nonces. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/security/nonces/.

Plugin API. N.d. Wordpress Codex, verkkosivusto. Viitattu 7.5.2020 https://codex.wordpress.org/Plugin_API

Plugin Basics. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/plugin-basics/.

Plugins. N.d. WordPressin kehittäjille suunnattu verkkosivusto. Viitattu 27.4.2020. https://wordpress.org/plugins/.

Sabin-Wilson, L. 2019. WordPress All-In-One For Dummies. For Dummies.

Settings API. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 7.5.2020. https://developer.wordpress.org/plugins/settings/settings-api/.

Shortocode API. N.d. WordPress Developers Guide: Common APIs Handbook, verkkosivusto. Viitattu 7.5.2020. https://developer.wordpress.org/apis/handbook/shortcode/.

Uninstall Methods. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 8.5.2020. https://developer.wordpress.org/plugins/plugin-basics/uninstall-methods/.

What is Plugin. N.d. WordPress Developers Guide: Plugin Handbook, verkkosivusto. Viitattu 29.4.2020. https://developer.wordpress.org/plugins/intro/what-is-a-plugin/.

What is Theme. N.d. WordPress Developers Guide: Theme Handbook, verkkosivusto. https://developer.wordpress.org/themes/getting-started/what-is-a-theme/. Viitattu 29.4.2020.