



**УНИВЕРСИТЕТ ПО БИБЛИОТЕКОЗНАНИЕ И
ИНФОРМАЦИОННИ ТЕХНОЛОГИИ**

**КАТЕДРА "КОМПЮТЪРНИ НАУКИ"
СПЕЦИАЛНОСТ «КОМПЮТЪРНИ НАУКИ»**

ДИПЛОМНА РАБОТА

на тема:

**ПРИЛОЖЕНИЕ НА ПЛЪГИНИТЕ /РАЗШИРЕНИЯТА/ В CMS
ПЛАТФОРМАТА WORDPRESS**

Дипломант:

Силвия Ганчева

задочно обучение

Ф.№ 37-КНЗ

Научен ръководител:.....

(проф. дн. Иван Гарванов)

София

2016

Резюме

Ганчева, С. Приложение на пългините /разширенията/ в CMS платформата WordPress. Научен ръководител (проф. дн. И. Гарванов). С. 2016. Катедра «Компютърни науки». Бакалавърска специалност "Компютърни науки". УНИБИТ.

82 стр. Брой източници – 18.

Целта на дипломната работа е да представи предимствата при използване на системи с отворен код като по-задълбочено е разгледана WordPress системата, да се разгледат по-популярните разширения, които използва системата и да се проектира пългин Слайдер за изображения . Описани са принципите, по които работи, начинът за инсталирането и използването на WordPress. Освен достъпен и лесен за употреба интерфейс, WordPress притежава и възможност за добавяна на пългини /разширения/ с различна функционалност към основното ядро на системата. Такива могат да бъдат пългини за оптимизация, кеширане, контактни форми, връзки със социалните мрежи, електронни магазини, галерии и др. В резултат на натрупани знания и умения е разработен пългин от тип Слайдер за изображения, който е бил необходим да бъде добавен към определена тема, която не е поддържала такъв. Пългинът е разработен, така че да функционира и с други теми на WordPress. Използван е семпъл дизайн, подходящ за съвременен сайт, който предоставя лесен за използване интерфейс с различни опции. Дипломната работа може да послужи на хора, занимаващи се с WordPress, начинаещи разработчици на пългини и теми за системата, както и на WordPress програмисти с повече опит, които биха искали да обогатят знанията си.

Ключови думи: WordPress, CMS платформа, система, пългин, слайдер, изображения, приложение, отворен код, Content Management System.

Съдържание:

Увод	4
I. Глава първа - Анализ на системите за управление на съдържанието.	
Въведение в WordPress.	7
1.1 Същност, характеристики и значение на системите за управление на съдържанието	7
1.2 Архитектура на веб базираните системи за управление на съдържанието	11
1.1.1 Функционалност	11
1.1.2 Система за управление на бази данни и изпълними скриптове	14
1.1.3 Темплейт система	16
1.3 Сравнителен анализ и видове системи за управление на съдържанието	19
1.4 Въведение в WordPress	23
1.4.1 Защо да изберем WordPress	23
1.4.2 WordPress инсталация	23
1.4.3 Инсталиране на WordPress тема	26
II. Глава втора - Приложение на разширенията (plugins) в WordPress	30
1.2 Какво е плъгин	30
1.3 Инсталация и активация на плъгини	30
1.4 Популярни плъгини в WordPress	33
1.4.1 JetPack	33
1.4.2 WordPress SEO by Yoast	43
1.4.3 Contact form 7	43
1.4.4 W3 Total Cache	46
1.4.5 Hyper Cache Extended	48
1.4.6 Woo Commerce	50
1.4.7 YetAnother Related Plugin	53
1.4.8 qTranslate и WPML (The WordPress Multilingual Plugin)	54
1.4.8.1 qTranslate	54

1.4.8.2	WPML	55
III.	Глава трета – Проектиране на plugin Слайдер	56
3.1	Основни неща, които трябва да знаем, преди да проектираме нашия плъгин	56
3.2	Да създадем плъгин от тип Слайдер за изображения	61
3.2.1	Първа стъпка – Дефиниране на съдържанието	61
3.2.2	Втора стъпка – Категоризация на слайдовете	64
3.2.3	Трета стъпка – Добавяне на опции	65
3.2.4	Четвърта стъпка – Визуализация на страница с опции	67
3.2.5	Пета стъпка – Добавяне на съдържание	68
3.2.6	Шеста стъпка – Създаване на слайдер	69
3.2.7	Седма стъпка – JavaScript и CSS	71
3.2.8	Осма стъпка – Добавяне на пост/страница със Слайдер	73
3.2.9	Девета стъпка – Лицензиране(Стандарти)	74
3.2.10	Десета стъпка – Визуализация	74
	Заклучение	77
	Използвана литература	79

Увод

В началото на третото хилядолетие компютърните технологии намират приложение в почти всички сфери от реалния живот и различните видове социални, икономически, стопански, производствени, търговски, рекламни и други дейности. Това е наложено от все по-големите и по-прецизни изисквания към обработката, съхраняването и достъпа до информацията, която е главен обект в много от тези дейности. Уеб сайтовете в световната мрежа съдържат огромни количества от информация, които са достъпни без каквито и да било ограничения от всяка точка на света и по всяко време. В много случаи обаче тази информация не се актуализира редовно и навреме и често е непълна, противоречива, остаряла или погрешна. Най-честата причина за това е, че уеб сайтът е сложна комбинация от технологии, графичен дизайн и бизнес логика и всяка промяна на неговото съдържание изисква намесата на квалифицирани за целта специалисти. По тази причина поддържането и използването на квалифицирана и гарантирана информация подлежи на по-специално управление и координация поради високото си комерсиално значение. Основни потребители на такъв род информация се явяват бизнес потребителите. Те са тези, които се нуждаят от сигурност и надеждност на данните.

Системите за управление на съдържанието (CMS) спомагат за разрешаването на подобни проблеми, като автоматизират и улесняват процеса на добавяне и промяна на съдържанието в сайтовете. Те са компютърен софтуер, който подпомага своите потребители в процеса на управление на съдържанието на динамичен уеб сайт, като улеснява организацията, контрола и публикуването на големи количества документи и друго съдържание като изображения и мултимедийни ресурси. Към WordPress може да се инсталират готови плъгини (разширения), които предоставят допълнителни възможности и подобряват функционалността на системата. Съществува богат набор от разработени плъгини, както свободно разпространявани, така и с изработка срещу заплащане. Такива могат да бъдат намерени на официалния сайт на създателите на системата.

Целта на настоящата дипломна работа е да изследва, анализира и сравни характеристиките на различните видове плъгини в системите за управление на съдържанието, както и изискванията, на които трябва да отговарят. На база на този анализ е разработен плъгин – Слайдер за изображения, който може да се интегрира в различни теми, използвани от CMS платформата на WordPress. Чрез този плъгин се оптимизира и улеснява работата на програмиста при смяна на тема и съдържание, тъй като плъгинът се явява допълнителна част към основата на платформата. Той улеснява и не утежнява цялостно сайта или блога, базиран е на PHP, JavaScript и CSS.

Глава 1 е теоретична, изяснени са същността, основните характеристики и архитектурата на една уеб базирана система за управление на съдържанието. Направен е сравнителен анализ на видовете системи за управление на съдържанието като са описани положителните и отрицателните характеристики. В следствие на изводите е направена въвеждаща част в WordPress, състояща се от предимства на системата и основна инсталация.

Глава 2 разглежда приложението на разширенията (plugins) в WordPress. Изяснено е понятието “plugin”, основните принципи при инсталиране в WordPress, също така са разгледани популярните плъгини в системата. Те добавят различни функционалности и подобрения към сайта или блога и са от най-различен тип. Необходим е плъгин, с който да следим посещаемостта към сайта; друг, който добавя Search Engine Optimization (SEO), за да направим сайта откриваем от търсачките в Internet (Google и др.); трети, който добавя контактна форма към WordPress, с което се улесняват потребителите, при желание да се свържат с нас. Препоръчително е използването на плъгин за кеширане на информацията, както и на няколко допълнителни: за електронен магазин в сайта и за превод на няколко езика.

В Глава 3 са разгледани принципите и практиките при разработването на плъгини. WordPress системата е с отворен код и всеки програмист може да разработи плъгин или тема, но е важно те да са оптимално съвместими с добрите практики за програмиране. Разработен е плъгин Слайдер за

изображения, който предоставя възможност за създаване и редактиране на снимките, както и добавяне на нови такива, независими от темата, която се използва за визуализация. Плъгинът добавя допълнителни характеристики като категория, размери, различна скорост и ефекти при смяна на изображенията. Той е относително лесен като интерфейс и настройки за потребителя, което улеснява работата с него – изисква се единствено работеща WordPress система и познания при инсталирането и активирането на плъгини в панела. Плъгините се разработват за функционалността, а темите – за интерфейса на системата. Целта на разработения плъгин е да добави допълнителна функционалност, която е независима при смяна на темата и същевременно допълнително украсява сайта и би могла да се използва като галерия от изображения. Слайдерът може да изобразява допълнителна актуална информация за фирма/компания/ под формата на снимки от събития или дейности, които тя предлага. Аудиторията, за която е предназначен плъгинът са потребителите, които имат блогове или сайтове на WordPress, но нямат Слайдер, включен към темата.

I. Глава първа: Анализ на системите за управление на съдържанието. Въведение в WordPress.

1.1 Същност, характеристики и значение на системите за управление на съдържанието

Система за управление на съдържанието (англ.: Content Management System – CMS) е вид приложен софтуер, който позволява публикуване и редактиране на съдържание в Интернет, както и поддръжка на главен интерфейс. Целта е да се улесни изграждането на динамичен уебсайт, с възможност за лесна и бърза промяна на съдържанието му. Тези системи значително улесняват работата в екип и предлагат много възможности за делегиране на различни административни права и роли в процеса на създаване и редактиране на съдържанието.

Уеб базираната система за управление на съдържанието обхваща цялостния жизнен цикъл на уеб страниците на сайта – от осигуряването на прости инструменти за създаване на съдържанието, до публикуването и архивирането му. Тя също осигурява възможността за управление на структурата на сайта, външния вид на публикуваните страници и навигацията, осигурена за потребителите. [16]

Управлението на съдържанието (Content Management) е набор от процеси и технологии, които поддържат еволюционния жизнен цикъл на електронната информация. Тази информация често се нарича съдържание или по-точно казано електронно съдържание. То може да е под формата на текст – като документи, мултимедийни файлове – като аудио или видео файлове, или всеки друг тип файлове, чийто жизнен цикъл изисква управление. Жизненият цикъл на електронното съдържание се състои от 6 основни фази:

- Създаване (create);
- Преработване (update);
- Публикуване (publish);

- Трансформиране (translate);
- Архивиране (archive);
- Премахване (retire). [18]

Управлението на съдържанието е съвместен процес. То често включва следните основни роли и отговорности:

- Автор на съдържанието: Отговорен за създаването и редактирането му;
- Редактор: Отговорен за проверката на съдържанието и за начина му на поднасяне на читателите;
- Издател: Отговорен за публикуването на съдържанието за употреба;
- Администратор: Отговорен за запазването на съдържанието в хранилище, така, че по-късно то да може да бъде намерено и използвано. Администраторът също управлява потребителите и правата, които има всеки от тях върху съдържанието;
- Потребител или гост: Човекът, който чете или възприема съдържанието, след като то е публикувано.

Уеб базираните системи за управление на съдържанието често се използват за съхраняване, контролиране, редактиране и публикуване на специфична за дейността документация като новини, бизнес документация, технически ръководства, обучаващи материали, продуктови каталози, ценови листи, маркетингови брошури и др.[5] Те се разработват като част от така наречените динамични сайтове и обикновено работят на сървър на уеб сайта. Те осигуряват контролиран достъп на различни групи потребители като администратори, главни редактори, редактори и автори на съдържание. Повечето системи използват база данни за съхранение на съдържанието и слой за презентация, който показва съдържанието на посетителите на сайта като набор от шаблони. Достъпът до системата обикновено се извършва чрез уеб браузър, като е възможно и използване на FTP за качване на съдържание.[7]

Системата за управление на съдържанието се различава от софтуера за изграждане на уеб сайтове като Microsoft FrontPage или Macromedia Dreamweaver по това, че позволява на неспециалисти с малко или никакъв опит да правят промени в съществуващ уеб сайт. Системата за управление на съдържанието е лесен за използване инструмент, който дава на упълномощените за това потребители възможността да управляват уебсайта. Тази система е инструмент за поддържане на уеб сайт, а не за създаването му.

Значение на системите за управление на съдържанието

Системата за управление на съдържанието е критична за успеха на почти всеки сайт, но все още много организации не са запознати с тази технология. Голяма част от съдържанието е остаряло или неточно и трудно за намиране, обновяването на сайта е сложно и външният му вид е остарял. Това би могло да постави организацията в много неизгодна позиция пред нейните клиенти, ако те имат някакви въпроси или оплаквания. Системата за управление на съдържанието е създадена специално за да разреши тези и много други проблеми, свързани с управлението и поддържането на голям корпоративен уеб сайт.

Има широк кръг от предимства, до които може да доведе внедряването и използването на CMS:

- Намаляване на разходите по поддръжка на сайта;
- Добре организиран процес по създаване на съдържанието;
- По-бързо създаване на нови страници и обновяване на страници;
- По-голяма съгласуваност;
- Подобряване на навигацията на сайта;
- По-голяма гъвкавост на сайта;
- По-голяма сигурност;
- Намаляване на дублирането на информация;
- По-големи възможности за разрастване на сайта; [10]

По този начин организацията може да спести време и средства за обучение на персонала за публикуване на съдържание. Намалява се също работата на IT отдела по промени в уеб сайта. Системата намалява и времето за публикуване, позволявайки по-бърз достъп до вече публикуваното съдържание. Това е важен резултат за модерната организация. Колкото по-бърз е достъпът до публикуваното съдържание, толкова по-актуално и значимо е то.

Освен гореизброените, особено важно предимство, което системата за управление на съдържанието може да осигури е подпомагането на бизнес целите и стратегиите на организацията, като например повишаване на продажбите, увеличаване на потенциалните клиенти, повишаване на доверието на потребителите и подпомагане на връзките с клиенти.

Съществува връзка между системите за управление на съдържанието и другите информационни системи в организацията, включително:

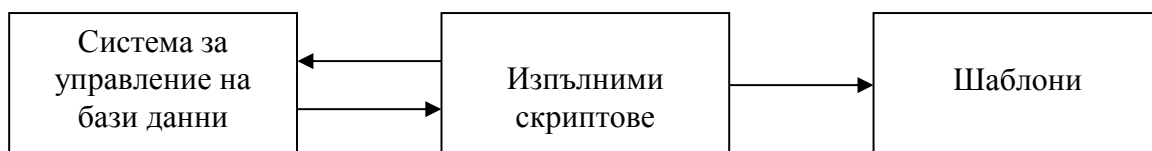
- Управление на документи;
- Управление на записи;
- Управление на електронна информация.

В момента тези системи обикновено се продават по отделно и постигането на съвместимост между тях не е лесно. Напредък в това отношение постига създаването на така наречените системи за управление на фирмено съдържание (Enterprise Content Management Systems- ECMS), които комбинират централна система за управление на съдържанието с други инструменти за управление на пълния набор от информация съществуваща в рамките на организацията. Тези системи са все още относително недоразвити и не е постигнато съгласие по въпроса какви възможности трябва да включват.[7]

Тъй като системите за управление на съдържанието са относително нов продукт на пазара и все още много организации не са запознати добре с тях, те имат потенциала силно да разширят своите пазарни позиции и да опростят значително създаването и поддръжката на сайтовете.

1.2 Архитектура на уеб базираните системи за управление на съдържанието

Съвременните системи за управление на съдържанието са изградени с трислойна архитектура, илюстрирана на Фигура №1 [2]. Всеки от тези слоеве ще бъде обяснен по-подробно в следващите точки.

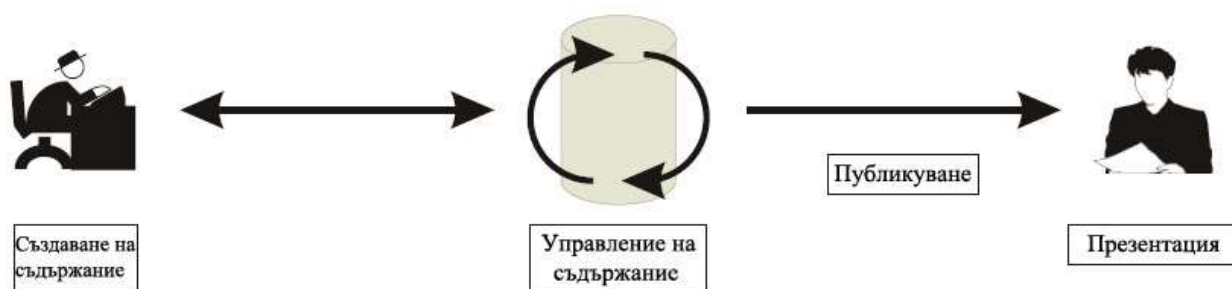


Фиг. №1 Архитектура на системата за управление на съдържанието

1.2.1 Функционалност

Функционалността на системата за управление на съдържанието може да се раздели на следните основни компоненти, представени на Фигура №2 [2]:

- създаване на съдържание
- управление на съдържание
- публикуване
- презентация



Фиг. №2 Компоненти на функционалността на системата за управление на съдържанието

Създаване на съдържание

На предна позиция в една система за управление на съдържанието стои лесна за използване среда за създаване на съдържание, проектирана да работи по подобие на Word. Тя осигурява нетехнически начин за създаване на нови страници или обновяване на съдържание без необходимост да се владее HTML. CMS позволява също да се управлява структурата на сайта, в кои секции и раздели да се разполагат страниците и как да са свързани една с друга. Много системи дори предлагат опростено drag-and-drop преструктуриране на сайта, без да се нарушават хипервръзките. Днес почти всички CMS осигуряват уеб базирана среда за създаване на съдържание, което допълнително улеснява тези задачи и позволява обновяването на съдържанието да бъде извършвано дистанционно. Тази среда е ключът за успеха на всяка система за управление на съдържанието. Чрез осигуряване на опростен механизъм за поддържане на сайта, създаването на съдържание може да бъде възложено на различни служители на организацията. Например маркетинговият мениджър може да поддържа секцията за новини и връзки с обществеността, докато продуктивният мениджър поддържа актуален продуктовия каталог.

Управление на съдържание

След като една страница бъде създадена тя се съхранява в централното хранилище на системата за управление на съдържанието. То съхранява цялото съдържание на сайта, както и друга поддържаща функциите и работата на CMS, информация. Това централно хранилище позволява на системата да осигурява редица полезни функции:

- Поддържане на информация за всички предишни версии на една страница, кой какво е променял и кога;

- Гарантиране на това, че всеки потребител може да променя само тази секция от сайта, за която е отговорен и за която са му дадени права от администратора;
- Интегриране със съществуващи информационни източници и IT системи. [18]

По-важното, системата за управление на съдържанието осигурява редица възможности за управление на работния процес по създаване на съдържанието. Това се илюстрира най-добре чрез следния пример:

Авторите на съдържание добавят своите документи в системата. Редакторите коригират, приемат или отхвърлят документи. Дизайнерите създават шаблоните и подреждат съдържанието в сайта. След това главните редактори са отговорни за публикуването на съдържанието в сайта. Системата за управление на съдържанието контролира и подпомага всяка стъпка от този работен процес, включително техническите задачи по публикуването на документите на един или повече уеб сървъра. На всяка стъпка CMS управлява статуса на страницата и уведомява хората, които са отговорни за съответния етап от работния процес. По този начин се позволява на повече автори да участват в управлението на съдържанието на сайта и се осигурява стриктен контрол върху качеството, достоверността и съгласуваността на информацията.[17]

Публикуване

Когато съдържанието е вече в завършен вид в хранилището, то може да бъде публикувано в интернет сайта. Системите за управление на съдържанието притежават и използват мощни средства за публикуване, които позволяват предварително дефинирани външен вид и подредба на сайта да бъдат автоматично приложени върху новото съдържание по време на публикуването му. Те могат също да позволяват едно и също съдържание да бъде публикувано

в множество сайтове. Разбира се, всеки отделен сайт има свой собствен изглед, така че CMS позволява на графичните дизайнери и уеб програмистите да дефинират и променят външния вид на сайта, който ще бъде приложен от системата. Тези възможности за публикуване гарантират, че страниците в целия сайт ще изглеждат еднотипно и позволяват висок стандарт на презентация. Те позволяват също авторите да се концентрират върху създаването на съдържанието, оставяйки външния облик на сайта изцяло на системата за управление на съдържанието. Тя изцяло автоматизира процеса на публикуване на съдържанието. [11]

Много от системите поддържат концепцията за ‘single source publishing’. Това означава, че те могат да публикуват автоматично дадена единица съдържание освен като HTML страница и в различни други формати, като PDF, Word, WAP и XML.

Презентация

Системата за управление на съдържанието може също да осигурява възможности за повишаване на качеството и ефективността на сайта. Например CMS може да изгради навигацията на сайта чрез прочитане на структурата на съдържанието направо от хранилището. Тя също улеснява поддръжката на множество браузъри и достъпността за потребители с увреждания. CMS прави сайта актуален, динамичен и интерактивен.

1.2.2 Система за управление на бази данни и изпълними скриптове

Съдържанието и цялата друга информация, свързана със сайта обикновено се съхранява в сървърно-базирана система за управление на релационни бази от данни. Използват се изпълними скриптове, написани на скриптов езици или инструменти като Coldfusion, PHP, Perl, JSP, ASP, Python

за взаимодействие с данните и превръщането им във визуално съдържание. Данните, съхранявани в базата данни се извличат и преобразуват в html страници или други документи. След това страниците, контролирани и публикувани от системата за управление на съдържанието могат да бъдат видени от посетителите на сайта.

База данни е организирана колекция от данни, използвана с цел моделиране на някакъв вид организация или организационен процес. В дните преди релационния модел на бази данни обикновено са се използвали два други модела за поддържане и действие с данни – йерархичен модел на бази данни и мрежов модел на бази данни. [14]

Извличането на данни от релационна база данни става с езика SQL (StructuredQueryLanguage – Език за структурирани заявки). SQL е стандартният език, използван за да се създават, модифицират и поддържат бази данни и да се отправят заявки към тях. Повечето от днешните по-важни софтуерни програми за релационни бази данни включват различни форми на реализация на SQL.[2]

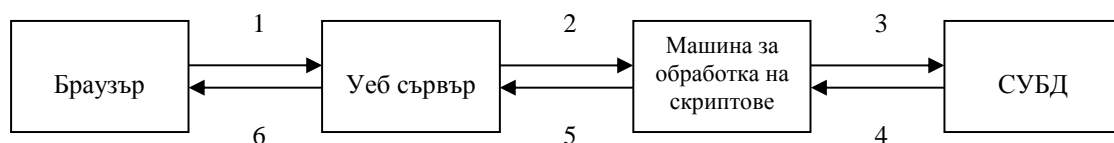
Една система за управление на релационни бази данни (СУРБД) представлява софтуерна програма, която се използва за създаване, поддръжка, модифициране и манипулиране с една релационна база данни. Много програми, реализиращи СУРБД предоставят инструменти за създаване на приложения за крайния потребител, които взаимодействат с данните, съхранени в базата данни.

СУРБД от типа клиент/сървър е разположен на компютър, изпълняващ ролята на сървър за бази данни, а потребителите взаимодействат с данните чрез приложения, разположени на техните собствени компютри като клиент на бази данни. СУРБД тип клиент/сървър от доста време се използват широко за управление на големи количества споделени данни. Някои примери за тях са:

- Microsoft SQL Server;
- Oracle Application Server;
- MySQL;
- PostgreSQL;

- MSSQL;[2]

Уеб базираните приложения с база данни следват обща структура, показана на Фигура №3.



Фиг. №3 Структура на уеб базираните приложения с база данни

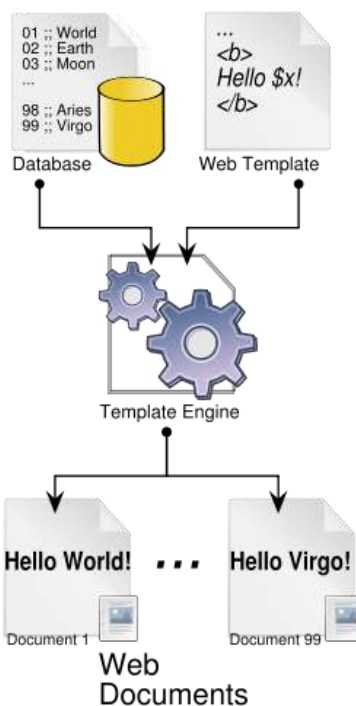
Основната архитектура се състои от уеб браузър, уеб сървър, машина за обработка на скриптове и система за управление на бази данни. Тя работи по следния начин[2]:

1. Потребителският уеб браузър изпраща HTTP заявка за определена уеб страница, например index.php.
2. Уеб сървърът получава заявката за index.php, извлича файла и го подава на машината за обработка на скриптове.
3. Машината започва да обработва скрипта. В скрипта има команда за свързване с базата данни и изпълняване на заявка. Машината отваря конекция към СУБД и изпраща съответната заявка.
4. СУБД получава заявката, обработва я и изпраща резултатите обратно на машината за обработка на скриптове.
5. Машината за обработка на скриптове довършва скрипта, което обикновено включва форматиране на резултатите в HTML. След това връща получения резултат на уеб сървъра.
6. Уеб сървърът изпраща резултата обратно на брауъра, където потребителят може да го види.

1.2.3 Темплейт система

Темплейт системата най-общо описва софтуера и методологиите, използвани за създаване на голямо количество уеб страници и уеб документи.

Тези системи обработват уеб темплейти чрез софтуер, наречен машина за работа с темплейти (Template engine). На Фигура №4 е представен основния процес за създаване на уеб документи, при който съдържанието (от база данни) и „спецификациите за презентация“ (уеб темплейт) се комбинират (чрез машината за работа с темплейти) за създаване на голямо количество документи.



Фиг. №4 Създаване на уеб документи

Темплейт системата се състои от следните елементи:

- Машина за работа с темплейти: основният обработващ елемент на системата;
- Източници на съдържание: различни видове входни потоци от данни, като такива от релационна база от данни, XML файлове и други видове локални или мрежови данни;
- Уеб темплейти: специфицирани в зависимост от езика за темплейти (Template language);[3]

За уеб дизайнера, когато всяка уеб страница е създадена чрез уеб темплейт той мисли за нея като за модулно структурирана страница с компоненти, които могат да се модифицират независимо един от друг –

заглавна част, заключителна част, глобална навигационна лента, локална навигационна лента и съдържание.

За програмистите езика за създаване на темплейти предлага по-ограничена логика, която се използва само за адаптиране на презентацията, а не за сложни бизнес алгоритми.

За останалите членове на екипа по разработването на сайта системата за уеб шаблони позволява на администратора на сайта да се съсредоточи върху техническата поддръжка и авторите на съдържание да се фокусират върху съдържанието.

Обобщено, употребата на система за уеб шаблони има следните предимства:

- Улеснена промяна на дизайна на сайта поради разделението на логиката на извличане на съдържанието от презентацията на сайта;
- Улеснено унифициране на интерфейса: менютата и другите презентационни елементи лесно се уеднаквяват, за улеснение на потребителите, които посещават сайта;
- Възможност за независима работа по дизайна и кода на сайта от различни хора по едно и също време;
- Увеличаване на производителността на разработване.[3]

Уеб темплейтът е елемент от уеб темплейт системата, който представлява предефинирано схематично оформление на страницата и определен брой програмни инструкции, написани на езика за темплейти. Употребата на уеб темплейти се свързва с един от основните принципи в дизайна – разделяне на съдържанието от презентацията.

Уеб темплейтът има определени базови характеристики. Тези характеристики могат да се опишат чрез следните основни принципи:

- Принцип на ефективното разделяне;
- Принцип на гъвкавата презентация.

Ефективно разделяне: обща цел на уеб разработчиците е да създават приложения, които са гъвкави и лесни за поддържане. Важна предпоставка за постигане на тази цел е разделянето на бизнес логиката от презентационната логика. Затова разработчиците използват уеб темплейт системи, които осигуряват в различна степен това разделение.[6]

Гъвкава презентация: в основата на ефективното разделяне е необходимостта от максимална гъвкавост в кода и ресурсите, отнасящи се до презентационната логика.[6]

1.3 Сравнителен анализ и видове системи за управление на съдържанието.

Системите за управление на съдържанието могат да се разделят на няколко вида:

- CMS със затворен изходен код – потребителят няма достъп до изходния код. Повечето от комерсиалните системи се разпространяват по този начин.
- CMS с отворен изходен код – потребителят има достъп до изходния код на програмата при определени условия. Има различни видове лицензи и условия, при които потребителят получава достъп до изходния код.
- CMS със свободен изходен код – потребителят има достъп до изходния код. Получава следните права – изпълнява програмата, изучава как работи програмата и да я променя за своите нужди, разпространява копия на програмата, подобрява програмата и публикува промените.[8]

Тъй като CMS е вид софтуер, той става обект на авторско право и обикновено отношенията между автора и крайния потребител се уреждат чрез специален договор, наричан софтуерен лиценз. Лицензите предоставят на потребителя правото да ползва CMS при определени условия. Предоставянето на това право обикновено става в рамките на няколко широко прилагани

модела:

- Платена CMS - Потребителят заплаща еднократно или периодично определена сума на автора на CMS. Това е най-често срещаният модел при CMS, разработвана с комерсиална цел.
- Безплатна CMS - Потребителят не заплаща за ползването на CMS.
- Свободна CMS - Безплатна CMS с отворен код, която потребителят може да променя и препубликува при определени условия.[8]

Готови решения:



Wordpress е най-популярната CMS система, първоначално планирана за лични блогове, днес предлага толкова много възможности, че с нея може да се направи онлайн магазин, каталог, новинарски сайт, форум, социална мрежа и почти всичко останало. Тя е безплатна с отворен код, базирана на PHP и MySQL. WordPress предлага изключително лесен административен панел, който е полезен за обикновените потребители, същевременно отворения код предлага голяма свобода на специалистите. Цялата система предлага огромен набор от безплатни теми и плъгини. Платформата е пусната в разпространение на 27 май 2003 г. от Мат Мюленвег и Майк Литъл.[4]



Joomla! е бесплатна система за управление на съдържанието с отворен код, написана на PHP, за публикуване на уеб съдържание. Използва база данни MySQL и техники на Обектно-ориентирано програмиране ООП. Тя е резултат от разделянето на екипа, който разработва Mambo през 2005 г. Joomla е сложна система, която предлага много възможности за създаване на уеб съдържание. Разделена е на следните разширения:

- Компоненти – Това е основно средство, което разширява функционалните възможности на Joomla и представлява набор от скриптове, които отговарят за определени функции
- Плъгини – Това са по-напреднали разширения и по същество, обработват дадено събитие.
- Шаблони – Описват основния дизайн на сайтовете в Joomla. Те са разширенията, които позволяват на потребителите да променят външния облик на сайта.
- Модули – Средство за разширяване на възможностите в Joomla. В повечето случаи, модулите изпълняват функции, които показват информация и се явяват допълнение към компонентите.
- Езици – Много прости разширения, които може да се използват като основна част или като софтуерно разширение.[12]



Drupal е система за изграждане на сайт и управление на съдържанието, с помощта на която може бързо и без специални познания по уеб дизайн и програмиране да се поддържа уеб сайт. Подходящ за изграждане на личен или фирмен сайт с блог и форум. Също така има модули за електронен магазин. Вградените модули дават възможност за подредба на съдържанието по

категории, оптимизация за търсачки, публикуване на новини от сайта в други сайтове с RSS и други функции. CMS системата е написана на езика PHP и съхранява информацията в база.[9]



OpenCart е софтуер за онлайн магазини с отворен код, с който може да се отвори професионален онлайн магазин бързо и лесно, като осигурява всичко необходимо за успешна работа. Не изисква специализирани познания за работа с електронни магазини. С OpenCart се работи с лекота, тъй като има опростен и лесен за работа интерфейс и предлага неограничени възможности за неговото развиване, персонализиране и обогатяване.[15]



Magento е CMS система с отворен код, специализиран за електронна търговия. Създаден е на 31 март 2008г. Използва технологията Zend Framework. Като цяло това е доста сложен продукт, с който не се работи лесно, но за сметка на това предлага неограничени възможности и функции, които конкурентите му не предлагат.[13]

1.4 Въведение в WordPress.

1.4.1 Защо да изберем WordPress?

Лесен за използване

WordPress е лесен за употреба. Неговият интерфейс и редизайн във версия 2.7 е по-лесен и по-интуитивен, отколкото, когато и да било преди. Управлението на публикации в блога, страници, мнения, потребители, плъгини се задейства само с няколко кликания.

Лесен за учене

Всичко е обединено в едно WordPress ръководство. Въпреки че системата е сложна и по-трудна за разработка, работата с нея отнема само няколко часа, нужни, за да се усвоят множеството инструменти, които WordPress предлага.

Много повече от Блог!

WordPress помага на хората да създават много повече от блогове. Това е платформа, която дава възможност за създаване на магазини, каталози, продукти и дори цели общности с потребители.

1.4.2 WordPress инсталация

По време на създаването на блог, трябва да имаме в предвид следните няколко неща:

- **Име на домейн.** Къде ще инсталираме WordPress ако нямаме интернет страница!
- **Хост.** Ще се нуждаем и от хост сървър за нашата WordPress инсталация. Има хиляди решения при избора на хост, като се започне от големи хостинг корпорации и се стигне до независими хостинг решения от физически лица.
- **FTP клиент.** FTP или File Transfer Protocol, е по-лесна възможност да качим инсталационните файлове на WordPress на нашия сървър, директно

от компютъра. Също така имаме нужда от FTP клиент, за да инсталираме WordPress теми, плъгини, а в някои случаи, за да инсталираме WordPress сами.

Инсталиране с **Фантастико**

Най-бързият и лесен начин да получим WordPress инсталация в движение е чрез програма, наречена **Фантастико**. Фантастико е включен по подразбиране в повечето стандартни хост системи. За да видим дали имаме достъп до тази програма, влизаме в нашия cPanel профил (обикновено <http://нашия-домейн.com/cpanel>) и търсим Фантастико (или Фантастико De Luxe в някои случаи), илюстрирано на Фигура №5. [4]



Фиг. №5 Фантастико в cPanel профил

Инсталиране на WordPress е възможно със **Фантастико** само в три бързи стъпки:

Стъпка 1

Отваряме **Фантастико**.

Стъпка 2

Кликаме върху "WordPress", а след това "New installation" (Фигура №6).



Фиг. №6 Инстлиране на WordPress чрез Фантастико

Стъпка 3

Попълваме необходимите полета и натискаме "Finish installation" (Фигура №7).



Фиг. №7 Инстлиране на WordPress чрез Фантастико

Инсталиране за 5 минути

Това е бърз вариант на инсталация, за онези от нас, които не обичат да чакат прекалено дългите инсталации:

1. Изтегляме и разархивиране в WordPress пакета.
2. Създаваме база данни за WordPress на нашия уеб сървър, както и MySQL потребител, който има всички привилегии за достъп до него.

3. Преименуваме WP-config-sample.php файла на WP-config.php.

4. Отваряме WP-config.php през текстов редактор и попълваме нашите данни, както е обяснено във файла WP-config.php.

5. Поставяме WordPress файла в желаното място на нашия уеб сървър.

6. Ако искаме да направим WordPress инсталация в собствена поддиректория от нашия уеб сайт (напр. <http://нашия-домейн/име-на-папка/>), преименуваме директорията WordPress до името, което искаме да има на поддиректория и я местим до желаното място на уеб сървъра.

7. Стартираме инсталацията на WordPress чрез файла wp-admin/install.php.

8. Ако сме инсталирали WordPress в главната директория, можем да го отворим от: <http://нашия-домейн.com/wp-admin/install.php>.

9. Ако сме инсталирали WordPress в нашата собствена поддиректория наречена blog, например, трябва да посетим: <http://нашия-домейн.com/blog/wp-admin/install.php>.

Това е всичко! WordPress сега би трябвало да бъде инсталиран безпроблемно.

1.4.3 Инсталиране на WordPress тема.

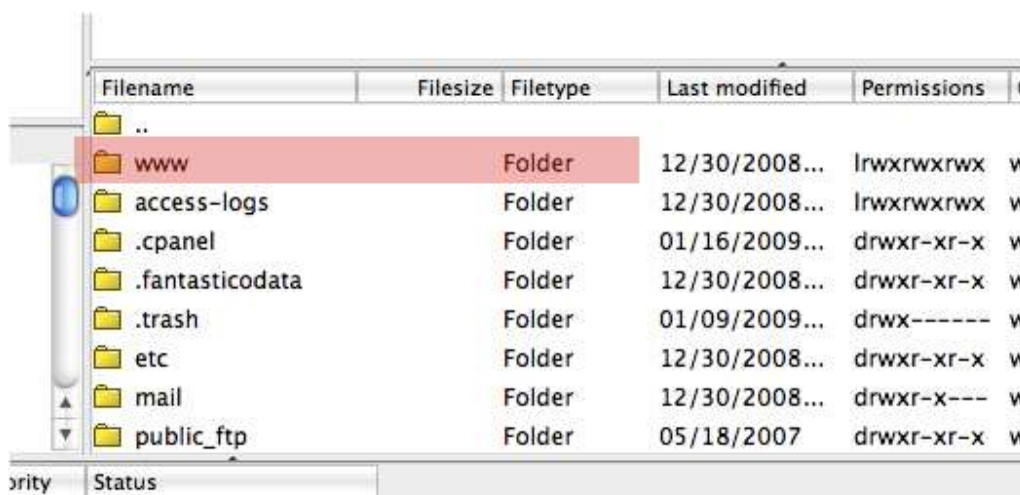
Дизайнът на WordPress сайт, по подразбиране е известен като „тема“. Съществуват буквално хиляди безплатни WordPress теми и инструкции как да ги инсталираме, като също така може да закупим и специални премиум теми, които предлагат множество допълнителни опции за персонализиране.

Търсейки в Google за „свободни WordPress теми“, ще попаднем на хиляди резултати, от които бихме могли да изберем подходящата за нас тема.

Инсталирането на темата става бързо и просто.

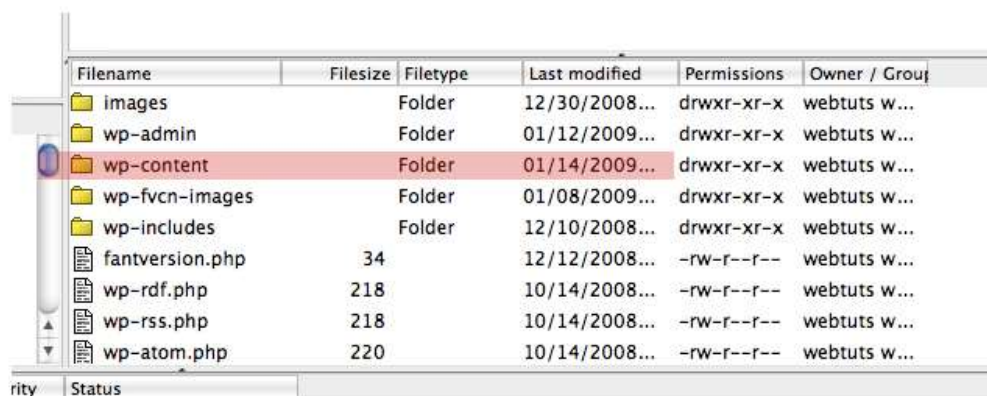
1. Изтегляме тема по наш избор.

2. Отваряме нашия FTP клиент и влизаме в основната / WWW / папка (Фигура №8).



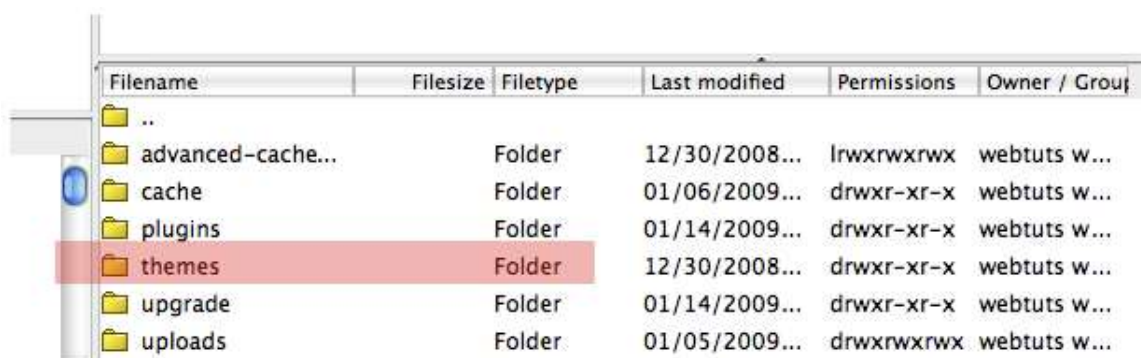
Фиг. №8 Инсталиране на WordPress тема

3. Придвижваме се към нашето WP-съдържание и по-точно в директорията (/ WWW / WP-content), показано на Фигура №9.



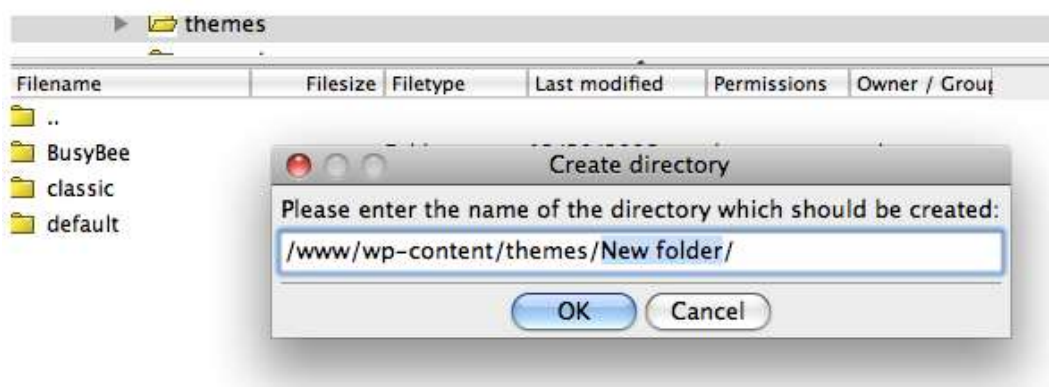
Фиг. №9 Инсталиране на WordPress тема

4. Придвижваме се към папката (/ wp-content/themes) – Фигура №10.



Фиг. №10 Инсталиране на WordPress тема в папка themes

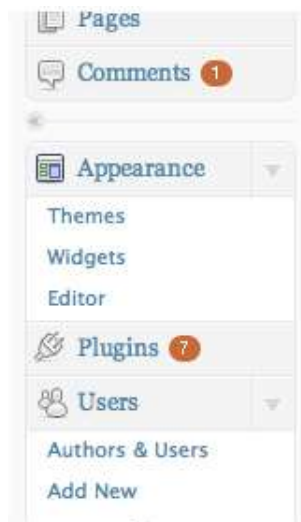
5. Създаваме нова директория с име по наш избор (С десния бутон > „Създаване на директория“) – Фигура №11.



Фиг. №11 Създаване на нова директория за темата

6. Избираме темата, която сме изтеглили в тази нова директория.

След това влизаме в профила си в WordPress административния панел (<http://нашия-домейн.com/wp-admin>). В лявата странична лента, кликваме върху „Външен вид“, а след това „Палитри“ („Themes“), както е показано на Фигура №12.



Фиг. №12 WordPress административен панел

Ако всичко дотук е преминало гладко, би трябвало да видим темата, която сме инсталирали. Просто избираме темата и кликваме върху "Инсталиране." Това е всичко!

Извод

В Глава първа са обяснени понятията: Система за управление на съдържанието; Жизнен цикъл на електронното съдържание; Основни роли при управление на съдържанието. Системите на управление на съдържанието използват хранилище за публикуване на съдържанието, автоматизирани шаблони за визуализацията, лесно добавяне на нови възможности, които разширяват функционалността на сайта. Има много предимства като: ниски разходи по поддръжка, съгласуваност, по-добра навигация, повече сигурност и др., които могат да доведат до внедряване и използване на CMS. Тази система има трислойна архитектура, състояща се от Система за управление на бази данни, Изпълними скриптове и Шаблони. Използва скриптов езици като PHP, Perl, ASP, Python и др. за взаимодействие с данните и преобразуването им в HTML страници или други документи. В Главата е направен сравнителен анализ на съществуващите CMS системи – WordPress, Joomla, Python, Drupal, OpenCart и Magento, на базата на който е избран WordPress, заради лесната употреба, интуитивния интерфейс и бързата инсталация. Показано е

постъпково инсталиране на WordPress на определен сървър и на нашия компютър, базистно боравене с директориите и инсталиране на тема (интерфейс на системата). Вече имаме работеща система за управление на съдържанието, необходимо условие за разработване и инсталиране на плъгини.

II. Глава втора: Приложение на разширенията/plugins/в WordPress.

2.1 Какво е плъгин

WordPress плъгин е програма или група от една или повече функции, написани на PHP, която добавя специфична група от услуги или функционалности към WordPress сайт или блог. Той може да се интегрира с различни методи от WordPress API (Application Program Interface). На нашия уебсайт могат да бъдат инсталирани не само плъгини, които могат да добавят допълнителна функционалност към WordPress, също и такива, които могат да са като нова платформа за нашия сайт. WordPress е направен с помощта на стандарт за висококачествен код и произвежда семантично маркиране, което прави сайта много привлекателен за търсачките.

Много SEO специалисти, могат да направят сайта ни още по-интересен с помощта на подходящите плъгини. WordPress е лесен за управление. Той предлага вграден Updater, която позволява да актуализираме плъгини и теми от админ таблото. Освен това ни уведомява, когато има нова версия на WordPress на разположение.

Плъгините са чудесен начин за постоянен достъп до уебсайтове, през мобилните приложения. В най-общ смисъл плъгините са неизменна част от системата на WordPress и осигуряват новостите, от които се нуждае всеки съвременен и целящ бърз прогрес, бизнес!

2.2 Инсталация и активация на пългини

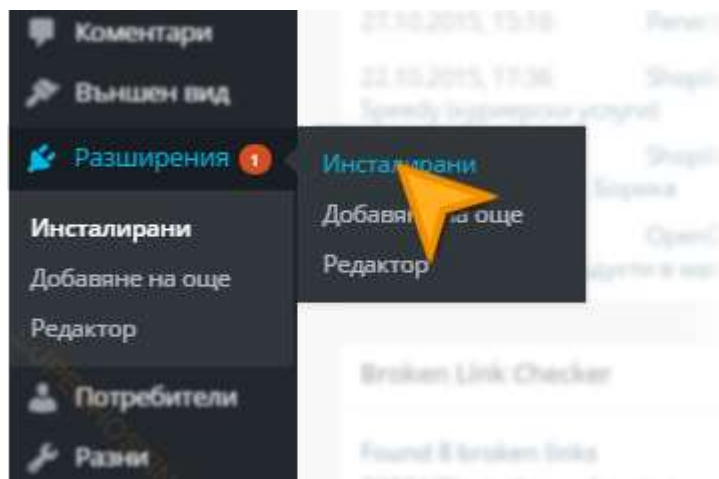
Към WordPress може да се инсталират готови пългини (разширения), които предоставят допълнителни възможности и подобряват функционалността на системата. Съществува богат набор от разработени пългини, както свободно разпространявани, така и с изработка срещу заплащане. Такива могат да бъдат намерени на официалния сайт на създателите на системата или на други специализирани в областта сайтове за изработка и разпространяване. Каталогът с пългини можете да намерим на официалния сайт на WordPress:

<https://wordpress.org/plugins>. [4]

Инсталацията на пългини се осъществява в няколко стъпки, които изпълняваме в административния панел на системата:

<http://your-domain.com/wp-admin>

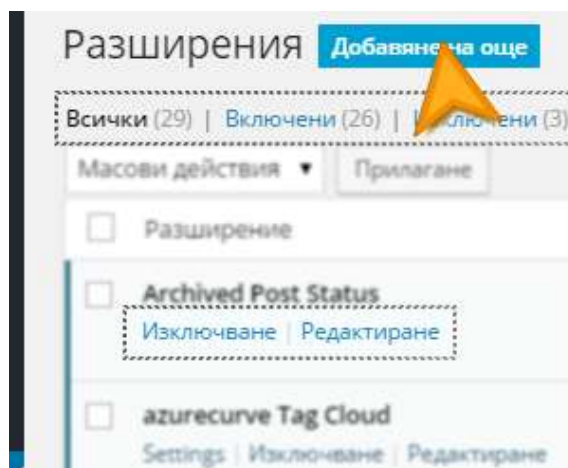
Бележка: Заместете your-domain.com с името на вашия домейн.



Фиг. №13 Инсталирани пългини в WordPress системата

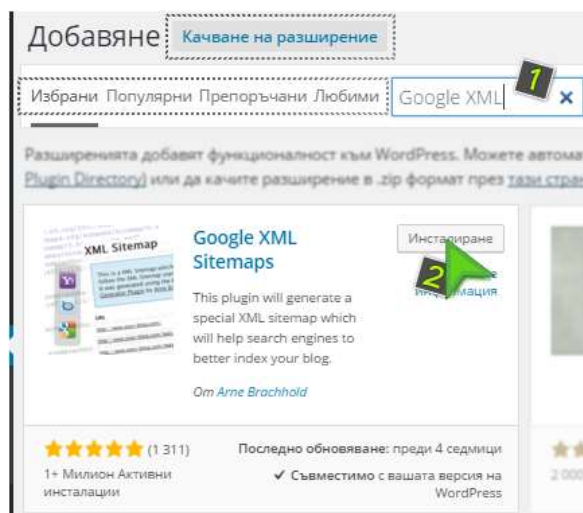
Избираме от менюто "Разширения" → "Инсталирани" (Фигура №13). Тук ще намерим текущите разширения, които са инсталирани, с кратко описание на техните функционалности.

Можем да ги подредим по "Всички", "Включени" или "Изключени" плъгини. За всяко включено разширение е наличен бутон "Изключване" и "Редактиране", а за останалите съответно е активна опцията "Включване" и "Изтриване". За да добавим нов плъгин, избираме "Разширения" -> "Добавяне на още", както е на Фигура №14.



Фиг. №14 Добавяне на нови плъгини

"Качване на разширение" - Можем да използваме тази опция, в случай че сме свалили локално на нашия компютър разширение от сайта на WordPress или друг специализиран сайт. Избираме бутон "Избор на файл" и посочваме .zip архива, разположен на нашия компютър, след което натискаме "Инсталиране" (Фигура №16) и изчакваме, докато приключи автоматичната инсталация. За да включим вече инсталираното от нас разширение, е необходимо да изберем "Включване".



Фиг. №15 Инсталиране на нови плъгини

"Избрани"; "Популярни"; "Препоръчани"; "Любими" - Можем да разгледаме готови колекции с безплатни разширения, подредени по определен критерий. За всяко от разширенията има възможност за "Още информация" или "Инсталиране". [4]

"Търсене на разширения" - В това поле въвеждаме дума, по която ще се извърши търсене в базата с плъгини на WordPress и натискаме клавиша на клавиатурата "Enter". Обикновено разширенията са наименувани на английски език, поради което при търсене по ключова дума на кирилица най-често не са налични резултати.

"Инсталиране" - От списъка с резултати можем да прегледаме плъгините, като обърнем внимание на информацията за тях - рейтинг, активни инсталации, последно обновяване и съвместимост с версията на нашия WordPress. След като изберем даден плъгин, кликваме на "Инсталиране", за да го инсталираме.

2.3 Популярни плъгини в WordPress

2.3.1 JetPack – този плъгин, показан на Фигура №16, съчетава набор от приставки, приспособления и услуги от WordPress.com. Той добавя към нашия сайт някои общи възможности с един лесен за използване и познат интерфейс. [4]



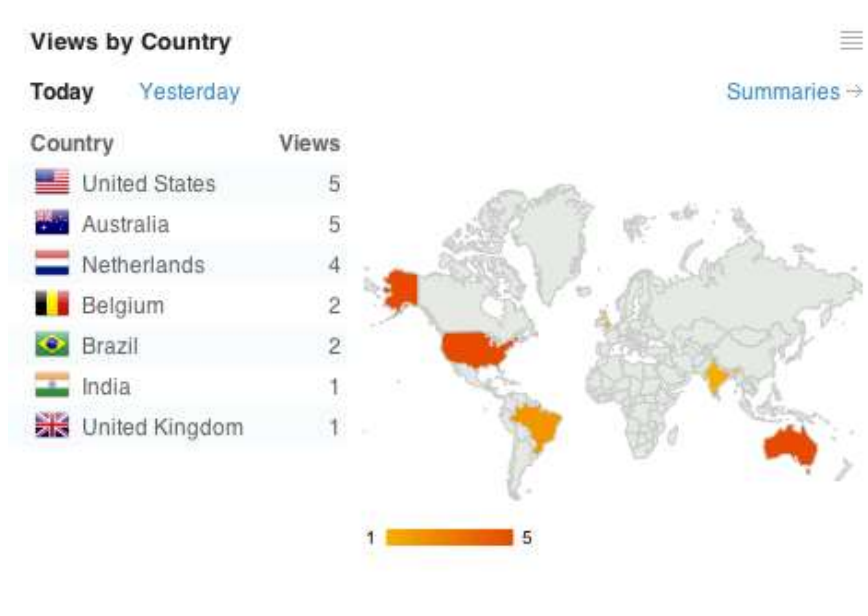
Фиг. №16 Плъгинът JetPack

Изисквания за инсталацията:

- Най-новата версия на WordPress.
- PHP версия 5.
- WordPress.com сметка
- XML-RPC активиран и публично достъпен.

Статистика – „Cloud“ Статистика

Анализа на функциите на Jetpack ще започнем със статистиката в WordPress. Статистиката, включена в Jetpack, предоставя лесен начин за получаване на популярните показатели чрез лесен и интуитивен интерфейс. И въпреки, че тя не съдържа всички данни, които можем да получим с помощта на Google Analytics например, тя дава отлична представа за статистиката на нашия сайт. Тя също така ни позволява да видим статистически данни директно от конзолата на WordPress, както е показано на Фигура №17.



Фиг. №17 “Cloud“ статистика


Можем също да видим най-популярните ключови думи и фрази, които посетителите използват, за да намерят нашия сайт. Наистина добра възможност в статистиката е проследяването на връзките, което не се предлага от показателите на Google Analytics (Фигура №18). Всяко кликуване върху връзките към нашия сайт ще бъде отчетено.



Фиг. №18 Статистика за посещаемостта на сайта

Коментари – Jetpack Коментари

Когато става дума за един блог, коментарите могат да бъдат основен начин за връзка с публиката. WordPress работи добре с коментарите, а също има изобилие от приставки, които добавят тази възможност. Функцията на Jetpack за коментари позволява на потребителите да публикуват от Twitter, Facebook или WordPress акаунти (Фигура №19). Това спестява време на потребителите, те не трябва да създават нов профил, за да коментират статията.



Фиг. №19 Форма за коментари

Можем да използваме и функцията за абонамент на Jetpack(RSS), за да бъдат информирани с всичко ново в сайта ни нашите коментатори. Всички актуализации или нови публикации ще бъдат изпратени на потребителя по електронна поща (според зададени настройки).

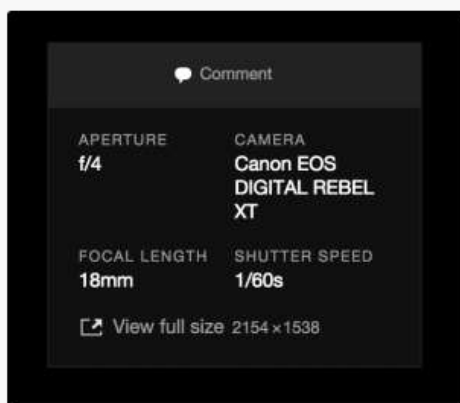
Jetpack Carousel – Вградена галерия

Галерията на Jetpack добавя много добре изглеждащ ефект в стила на Lightbox изображенията във всяка стандартна галерия на WordPress. Тя ни позволява да добавяме коментари, а също така има опция за показване на данни в EXIF формат. Регулирането на цветовете, показано на Фигура №20, с прост избор между бял и черен фон отнема няколко секунди и изглежда наистина добре.

You can set a black or white background for Carousel:



You can also choose to display the photo metadata (Exif) in the Carousel, if it's available.



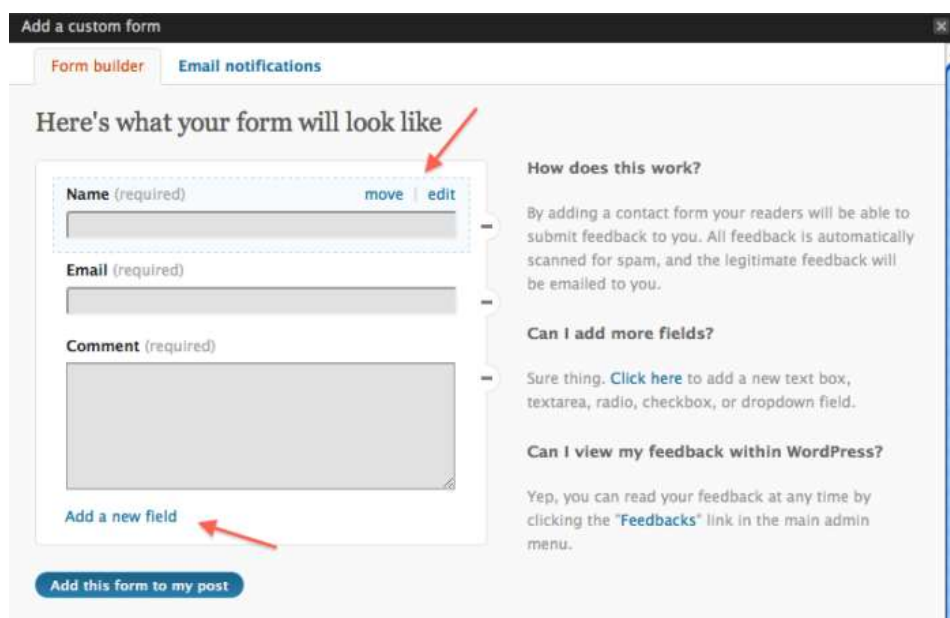
Фиг. №20 Jetpack Carousel

Форма за контакти

Принципно тя не е задължителна. Ако публикуваме имейл адрес и телефонен номер, посетителите на сайта пак ще могат да се свържат с нас. Ние искаме да ги улесним максимално. По този начин ще получаваме повече запитвания и съобщения от посетителите на сайта ни. За целта ползваме популярното разширение Contact Form или платеното Gravity Forms , което също работи отлично и е по-мощно.

Инсталираме Contact Form 7 като отидем на "Разширения>Добавяне на нови" и въведем името на плъгина в полето търсене. Формата за контакт, която плъгинът създава автоматично, е напълно достатъчна за нашите цели. Разширението създава код от типа "contact-form-7 id="95" title="Contact form 1" [ограден в квадратни скоби], който трябва да поставим на страницата, където искаме да бъде формата за контакт. В Jetpack има единна много лесна, но добра вградена форма. Във всеки запис, или страница, можем просто да кликнем върху бутона "Добави Форма" и ще имаме една много семпла форма за

контакти. Имаме няколко типа полета, но те са основно предназначени за получаване на информация от потребителите. Можем да получим бърз и лесен интерфейс за добавяне на форми за всяка публикация или страница, както е представено на Фигура №20.



Фиг. №20 Jetpack форма за контакти

Ако сме активирали Akismet, нашите форми също ще бъдат защитени от спам. Това решение е идеално, ако имаме нужда от прости форми. Всички те са вградени в един единствен плъгин.

Полезни допълнения

Вече разгледахме някои от основните функции, но има и други. Jetpack включва 27 отделни функции / плъгини, нека разгледаме накратко още някои от тях.

Разпространение в социалните мрежи

Почти всеки сайт или блог сега е свързан със социалните мрежи и ние можем да разпространяваме съдържание в тях. Например бутоните "Споделяне" са много популярни. Jetpack осигурява чудесен начин да добавим тези бутони. Можем лесно може да изберем кои мрежи искаме да включим и как искаме да

се визуализира бутонът (Фигура №21). Можем също така да добавим друг вид социална мрежа, ако не можем да намерим тази, която търсим от списъка с наличните.



Фиг. №21 Jetpack бутони за социалните мрежи

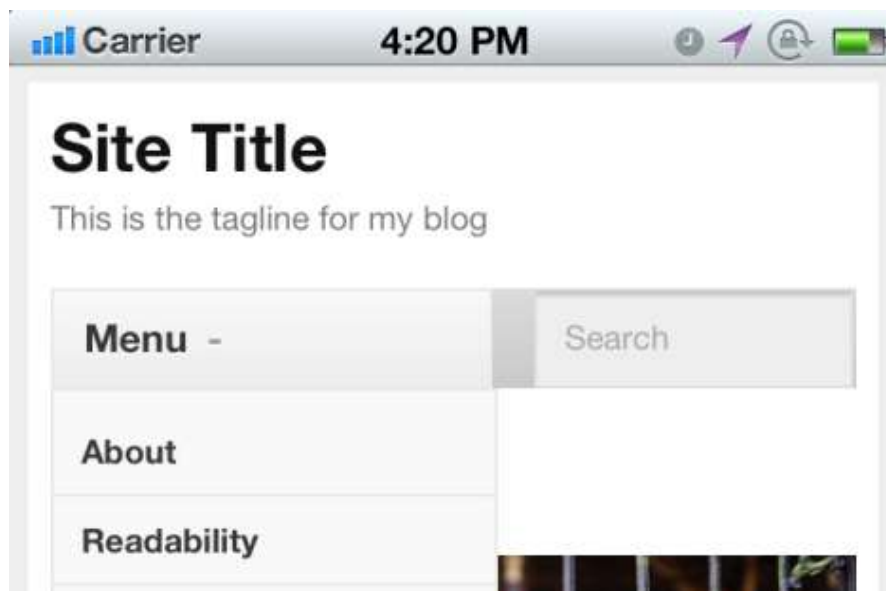
Gravatar и информация за автора

Ако използвате WordPress, вероятно знаете какво е Gravatar – това е друга услуга от Automattic, която работи като хостинг услуга за профилни снимки. Можем да създадем акаунт и да свържем снимката на профила си с нашия имейл адрес. Това означава, че като актуализираме образа на профила си на едно място, той се актуализира във всички сайтове, в които имаме профил (ако те използват Gravatar).

Jetpack съдържа и функция, наречена Gravatar Hovercards. Ако е включена, когато мишката е върху изображението ще се появи информация от профила ни в Gravatar /биография и снимка/. Тази допълнителна функционалност работи добре и позволява на потребителя бързо да получи информация за коментаторите.

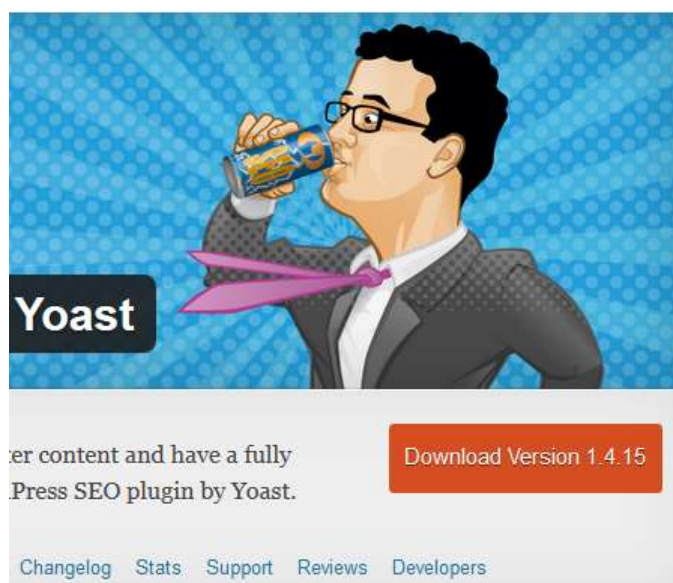
Mobile Theme

Активирайки тази опция, ще получим основна мобилна тема, показана на Фигура №22. Дизайнът е темата на WordPress Twenty Eleven. Това е много удобен, бърз и евтин начин да получим мобилна тема за нашия сайт.



Фиг. №22 Мобилна версия на Jetpack

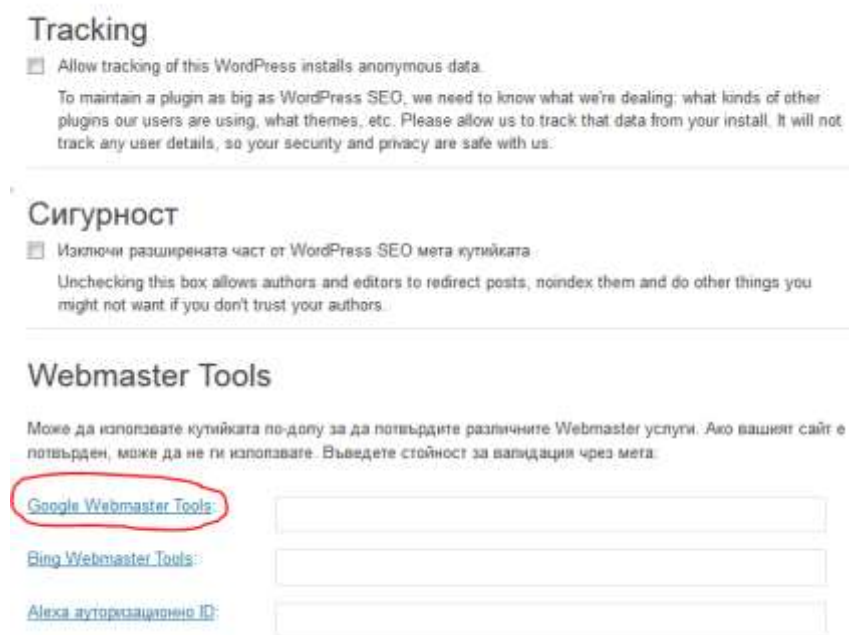
2.3.2 WordPress SEO by Yoast - Този плъгин ще ни помогне със SEO (Search Engine Optimization) оптимизацията на текстовете и това е много полезно, автоматично ще ни казва как и къде да използваме нашите ключови думи, за да ни намерят в Интернет (Фигура №23). Yoast е един SEO специалист от Холандия, който е създал този плъгин, в момента е най-добрият SEO плъгин за платформата WordPress. [4]



Фиг. №23 WordPress SEO by Yoast

След като инсталираме плъгина, той ще се появи в администраторския панел. Най-важните опции са събрани в разделите за проследяване и сигурност ("Tracking" и "Security"), показани на Фигура №24:

- чрез Тракингът пращаме информация на Yoast и му позволяваме да подобри плъгина.
- Сигурност – засега оставяме тази опция деактивирана



The screenshot shows the 'WordPress SEO' settings page. It is divided into three main sections: 'Tracking', 'Сигурност' (Security), and 'Webmaster Tools'. In the 'Tracking' section, there is a checkbox labeled 'Allow tracking of this WordPress install's anonymous data.' which is checked. Below it, a text block explains that tracking is needed to maintain the plugin's size and to provide better recommendations. In the 'Сигурност' section, there is a checkbox labeled 'Изключи разширената част от WordPress SEO мета кутийката' (Exclude the advanced part of the WordPress SEO meta box) which is unchecked. Below it, a text block explains that unchecking this box allows authors and editors to redirect posts, noindex them, and do other things they might not want if they don't trust their authors. In the 'Webmaster Tools' section, there is a text block explaining that the meta box can be used to verify various Webmaster services. Below this, there are three input fields for 'Google Webmaster Tools', 'Bing Webmaster Tools', and 'Alexa авторизационно ID'. The 'Google Webmaster Tools' field is highlighted with a red circle.

Фиг. №24 Опции на Tracking и Security

Titles and Metas

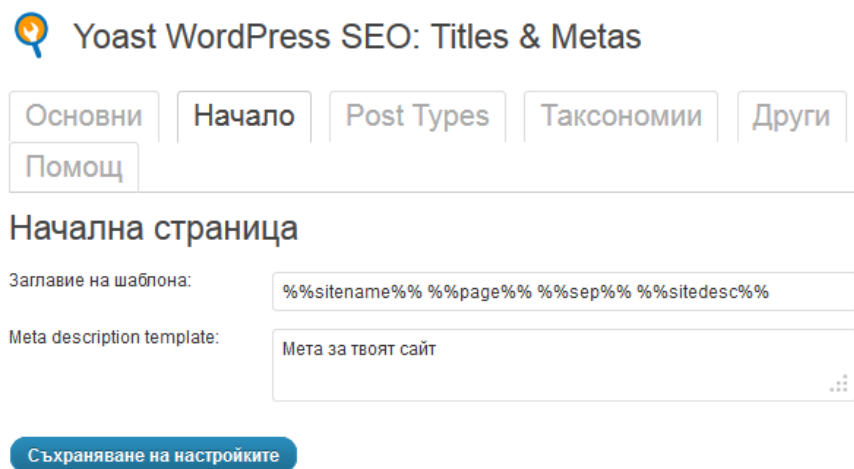
Използваме ги, за да нагласим изреченията, които излизат в долната част в резултатите на Google, например:

Уроци по рисуване и живопис за кандидат-студенти и ученици
silviagancheva.com/kursove-uroci-risuvane/ ▼
Уроци по рисуване, кандидат-студенти, ученици, портфолио, ... Добре дошли в сайта на Силвия Ганчева! ... Уроците по рисуване са подходящи за:
Посетихте тази страница много пъти. Последно посещение: 28.08.15

Фиг. №25 Резултат за сайт при търсене в Google

Горната снимка (Фигура №25) е от страницата на уеб дизайна на моя блог, появява се главното описание и името на сайта.

Следващата снимка (Фигура №26) е на „Meta tags“ от началната страница. В Meta tags description трябва сложим максимум 150 букви.



Фиг. №26 Визуализация на Titles & Metas в Yoast SEO

Google не използва в голяма част от случаите ключови думи, но останалата част от търсачките – все още използват. Ето защо е важно при определяне на мета таговете на постове и страници, те да бъдат разделени със запетая.

Това може да стане по следния начин:

- %%title%% - заглавия
- %%sitename%% - име на сайта
- %%excerpt%% - кратко описание, ключови думи
- %%sep%% — разделители /-,_./

Използвайте последователността, показана на Фигура №27, за да създаде заглавия и описания на постове и страниците:



Фиг. №27 Препоръчителна последователност за задаване на заглавие и описание на пост

Social - с тази настройка, можем да свържем нашия сайт с различни социални мрежи като Facebook, Twitter и Google Plus (Фигура №28), Pinterest и др.



Фиг. №28 Визуализация на Google+ настройки

Съществуват още настройки за добавяне на XML карта, Permalink настройки, RSS бутони, редактиране на файлове.

В "Content" частта вече излиза автоматично това, което сме конфигурирали преди в "SEOtitle", ако то е дълго или не ни харесва, можем да го променим. Най-важното е, след като направим анализи на ключовите думи, които хората търсят в интернет, да ги сложим в полето "Focus Keyword".

"Meta description" /Мета описание/ е един текст от 150 думи, който се показва в долната част на резултатите на Google. "Content Analysis" /Анализ на страницата/ е мястото, където плъгинът ни показва какви неща още можем да подобрим. "Advanced" /Разширени/ е мястото, където можем да си изберем дали искаме да излизаме в резултатите на Google, оттук можем да махнем поста или страницата от Site Map и да кажем на Google да не влиза в страницата. В "Social" /Социални/ можем да определим с какво заглавие, различно от това на публикацията, искаме да излиза тя в социалните мрежи. Тези опции са представени на Фигура №29:



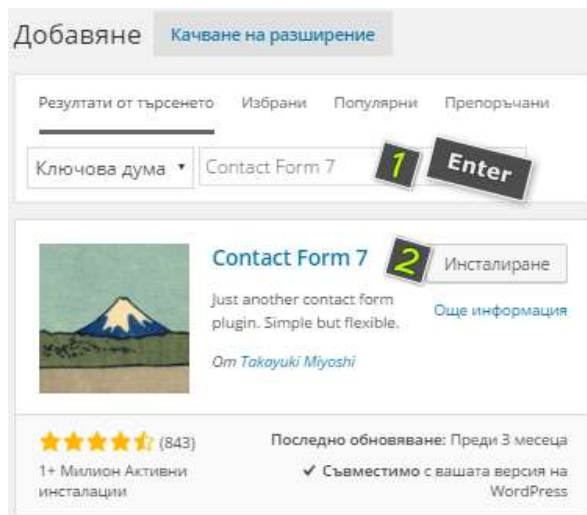
Фиг. №29 Настойки на content частта

Заклучение

Използването на тази приставка не само подобрява намирането на нашия сайт от търсачките, но има и положително въздействие върху броя на кликванията, както и привлича допълнителен трафик и нови посетители към нашия сайт или блог.

2.3.3 Contact form 7 – плъгин за добавяне на форма за контакти на нашия сайт, който предлага и опция за активиране на допълнителен код за сигурност - reCAPTCHA към контактната форма. reCAPTCHA кода се ползва за защита от спам съобщения.

Инсталацията на плъгина може да се извърши автоматично през административния панел на WordPress. От главното меню на WordPress в ляво изберете "Разширения" -> "Добавяне на още", както е показано на Фигура №30. В полето за търсене намираме "Contact Form 7". След инсталацията трябва да направим настройките на разширението.[4]

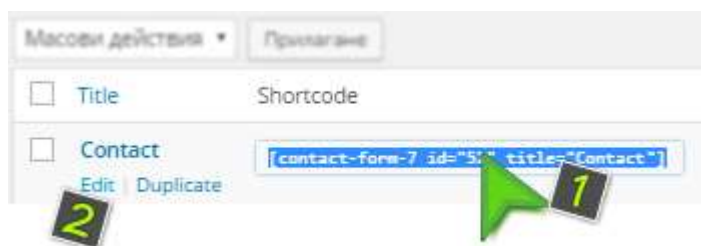


Фиг. №30 Инсталация на Contact Form 7

След като е инсталиран и активиран плъгина добавяме краткия код (1 – **Фигура №31**) (Shortcode) за контактната форма в страницата или публикацията, където искаме да се показва (например страница "Контакти").

Кодът за формата можем да намерим в настройките за формата или в списъка с форми Contact -> Contact Forms. По подразбиране след инсталация на плъгина има създадена една контактна форма Contact form 1, която можем да редактираме или създадем нова.

След като сме поставили кода за контактната форма, ще можем веднага да я видим от "Преглед" на страницата/публикацията/. За да извършим настройка и редакция на самата контактна форма, кликваме на бутона Edit (2 – **Фигура №31**).

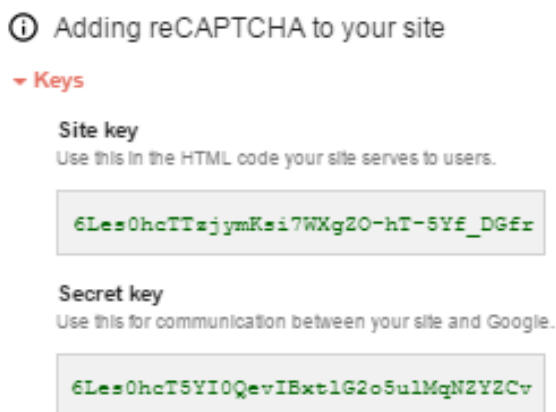


Фиг. №31 Shortcode на Contact Form 7

Добавяне на код за сигурност reCAPTCHA

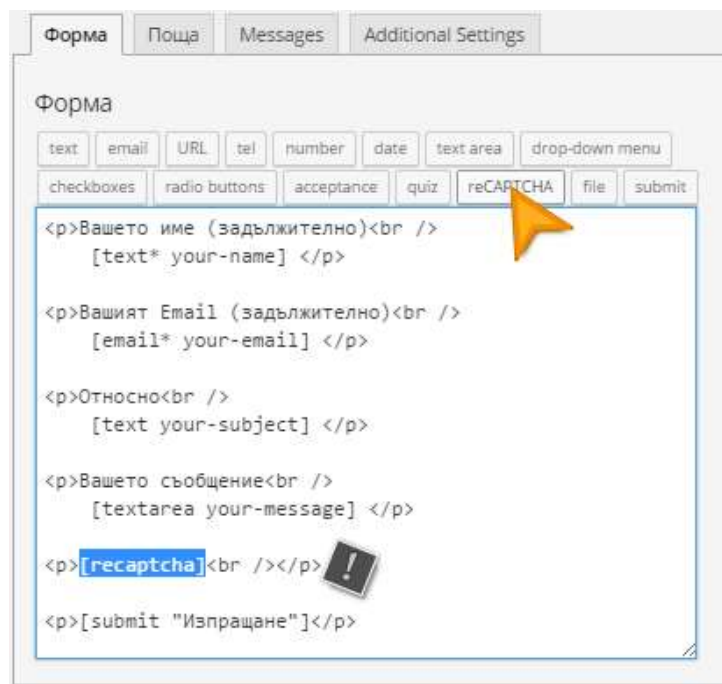
За да добавим кода за сигурност на Google - reCAPTCHA, първо ще е необходимо да разполагаме с два ключа (Site key и Secret key) – Фигура №32. Тези ключове можем да вземем от акаунт в Google, след като регистрираме нашия сайт:

<https://www.google.com/recaptcha/admin>



Фиг. №32 Добавяне на код за сигурност към сайта

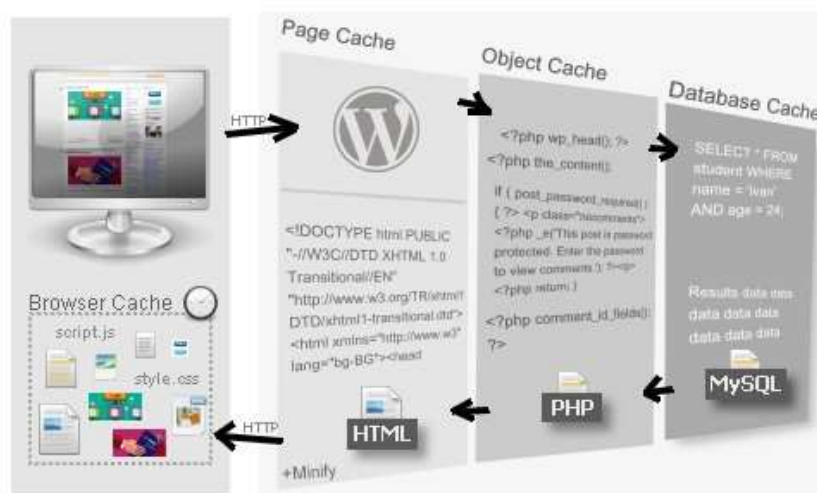
След като разполагаме с ключовете отиваме на настройките на плъгина Contact Form 7 през меню Contact -> Integration. Кликваме на бутона Configure Keys и въвеждаме двата ключа. От страницата с настройките на формата Contact -> Contact Forms избираме Edit на формата. В страницата на формата, в редактора с текста поставяме курсора на мишката там, където искаме кода за сигурност да се намира и кликваме на бутона reCAPTCHA. Можем да преместим кода [recaptcha] във формата, където смятаме че е най-подходящо да се показва, например преди бутона Изпращане, представено на Фигура №33.



Фиг. №33 Добавяне на код за сигурност в края на Формата за контакти

2.3.4 W3 Total Cache – плъгин за кеширане на информацията и своеобразна система за ускоряване на зареждането на уеб сайта.

Кеширането на информация се състои в това, че резултатът от динамично генерираната информация на сайта се запазва/записва. Така при следващо зареждане на сайта потребителят ще получи тази записана информация, вместо да изчаква генерирането на данните (Фигура №34). С кешираната информация се спестяват време и ресурси, необходими за непрекъснатото извличане и генериране на информация на сайта.[4]



Фиг. №34 Метод за кеширане на информацията

Плъгинът W3 Total Cache може да кешира информацията на няколко нива. На най-високото ниво е Page Cache. Там се записва готовият, окончателният код (HTML). В създаването му са използвани резултатите от SQL заявките и изпълнението на скриптовете на сайта. Следващото ниво е Object Cache, където се кешират обекти, представляващи временни данни, които са необходими по време на зареждането на дадена страница. И на най-ниското ниво е Database Cache, където се кешират най-често изпълнявани SQL заявки към базата данни.

Page Cache

При достъпване на страница от сайта, WordPress ще стартира и изпълни няколко действия, за да може да предостави информация в тази страница. Този процес представлява генериране на динамично съдържание. Скриптовете на страницата ще се стартират и ще се изпълнят няколко SQL заявки към базата данни. След това резултатите от базата данни ще се обработят и PHP ще генерира финалния код на страницата – HTML кодът, който се изпраща от уеб сървъра към уеб браузъра. Този процес на генериране на динамична информация, освен че отнема време, заема и определени ресурси от хостинг акаунта.

Когато Page Cache е активиран, генерираният финален HTML код на страницата ще бъде записан за следваща употреба. При следващо достъпване на тази страница системата директно ще подаде запазения HTML код към уеб сървъра, а той ще го изпрати на уеб браузъра. По този начин се спестява голяма част от динамичното генериране на информация, което се отразява в ускорено зареждане на страницата и намалена консумация на ресурси на сървъра.

Препоръчителна технология за кеширане: Memcached.

Основни настройки на Page Cache: В секция **General** нека обърнем внимание на опцията **Don't cache pages for following user roles**. Активирането на тази опция и избирането на потребителите, изброени под нея, ще ни гарантира, че никой логнат администратор, редактор или автор, няма да получава кеширана

информация. Това е полезно, когато в сайта се публикува често съдържание и ще предотврати показването на неактуална информация на потребителите.

Advanced

В тази секция, в полето на „Memcached host name:port / IP:port:“, трябва да поставим коректния порт за Memcache – например 127.0.0.1:**11236**. Кой е портът за Memcache можете да проверим през cPanel -> Memcached Manager.

Какво е Memcached?

Memcached е свободен софтуер с отворен код (Open source) (Фигура №35), който се използва за ускоряване на динамични уеб сайтове. Memcached кешира в паметта данни и обекти от изпълнението на заявки към базите данни, API извиквания или резултати от зареждането на страниците. Първоначално е създаден, за да се повиши производителността на LiveJournal. В наши дни вече се ползва от много сайтове като Wikipedia, Flickr, Twitter, Youtube, WordPress, Digg и др.



Фиг. №35 Кеширане посредством метода Memcached

2.3.5 Hyper Cache Extended – Кеширащ плъгин за WordPress, с който превръщаме сайта си в статичен

Описание

Hyper Cache Extended е популярен плъгин за WordPress, който прави нашият уебсайт от динамичен в статичен. Това помага доста за по-бързото зареждане на сайта и основно за много по-малкото натоварване на нашият хостинг.

Плъгинът е подходящ за големи WordPress сайтове с хиляди публикации и много трафик. Но не е задължително сайтът ни да е голям. Подходящ е също и за малки блогове, които се хостват на сравнително евтини хостинги с включени малък брой процесорни минути на ден.

Възможности

Настройките на плъгина Hyper Cache Extended, показани на Фигура №36, са сравнително малки и лесни за потребителя. Hyper Cache Extended има опция за задаване на време кога да се прави нов кеш.

Cached pages timeout е точно тази опция. В нея трябва да зададете през какъв интервал от време плъгинът да прави прекеширане на блога.

Max server load average е средна стойност на натоварването на сървъра. По подразбиране е 5, но ние можем да я променяме. Ако натоварването на нашия сървър е над зададената стойност, то нов кеш няма да бъде създаден, а потребителите ще виждат стар кеш. Това е по-добрия вариант от колкото да се вижда съобщение за грешка в сървъра.

Configuration	
Cached pages timeout	<input type="text" value="720"/> (minutes) Minutes a cached page is valid and served to the browser. After this time more used and will be regenerated on next request.
Max server load average	<input type="text" value="5"/> Hyper Cache Extended will serve the cached version if the server load is below this value.
Cache invalidation mode	<input type="text" value="Only modified posts"/> <input type="button" value="v"/> <input checked="" type="checkbox"/> Invalidate home, archives, categories on update "Invalidation" is the process of deleting cache and generating a new one. It happens when a post is updated, new comment, etc. so one or more cache entries are considered like a post modification where they should be regenerated.
Disable cache for commenters	<input type="checkbox"/> When users leave comments, WordPress sends a message to the browser to keep those features, enable this option. The cache will be regenerated.
Feeds caching	<input checked="" type="checkbox"/> When enabled the blog feeds will be cache. This improves the efficiency.

Фиг. №36 Настройки на Hyper Cache Extended

Disable cache for commenters е опция, с която разрешаваме или забраняваме коментарите в блога да се кешират. Ако оставим отметката празна, то плъгина Hyper Cache Extended ще кешира и коментарите. Това означава, че ако имаме блог, в който хората постоянно публикуват коментари, то те ще бъдат видими чак след следващото прекеширане. Ако включим отметката, то блогът ни ще се кешира, но без коментарите. Тоест те ще бъдат постоянно видими. Това от своя страна означава, че блогът ни няма да е съвсем статичен и ще отнема малко повече сървърни ресурси за да показва новите коментари.

Feeds caching е друга опция, с която също можем да намалим още процесорното време в което сървърът обслужва нашият блог. Когато я поставим, Hyper Cache Extended ще кешира и RSS-а на блога ни. Ако я махнем, то RSS-а няма да се кешира и блогът ни ще изразходва малко повече сървърни ресурси.

Плъгинът Hyper Cache Extended също така има още няколко функции, като например бутон с който изчистваме целия кеш. Също така плъгинът има функция наречена Cleaning Process, като тя има две опции – включено и изключено. Ако изберем опцията включено, плъгинът сам ще почиства старите кешове и ще държи само новите. Предлага се възможност и да правим кеш и на началната страница. По подразбиране опцията е изключена. Ако не обновяваме много често блога си, а същевременно товарим доста сървър, на който го хостваме, то можем да я включим.

Hyper Cache Extended има интеграция с плъгина WordPress Mobile Pack. Ако блогът ни например използва шаблон, който няма адаптивна версия, то най-вероятно бихме си сложили някой плъгин, който прави сайтът ни да се показва добре на всички смартфони и планшети. Ако сме избрали плъгина WordPress Mobile Pack, то Hyper Cache Extended ще кешира и него.

Плюсове

- Превръща WordPress сайта от динамичен в статичен.
- Блогът се зарежда по-бързо.
- Сървърът на който хостваме блога си се натоварва с 80-90% по-малко.

- Лесни настройки.
- Hyper Cache Extended е безплатен.

Минуси – няма

Заклучение

Ако имаме голям WordPress сайт с много посещения, Hyper Cache Extended може да ни спести не малко пари за хостинг. Той прави сайта ни изцяло статичен посредством кеш. И въпреки, че има доста подобни плъгини, голяма част от тях повече пречат отколкото помагат.

2.3.6 Woo Commerce - WooCommerce е безплатно разширение за електронна търговия (Фигура №37), което ни позволява лесно да продаваме всичко. С безпроблемната си интеграция с WordPress, WooCommerce е най-използваното решение за електронна търговия, което предоставя пълен контрол както на собствениците на електронни магазини, така и на разработчиците.[4]

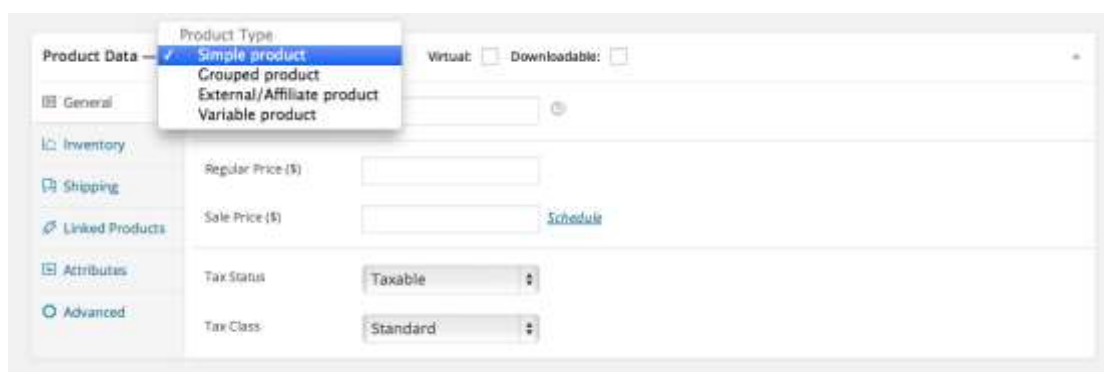


Фиг. №37 Woo Commerce

С WooCommerce можем да продаваме и физически, и дигитални стоки с всякакви размери и форми. Можем да предлагаме продуктови вариации, множество различни конфигурации и мигновени файлове за изтегляне, и дори да продаваме affiliate стоки от сайтове за онлайн търговия. С платените разширения можем да предлагаме резервации, членства и периодични абонаменти.

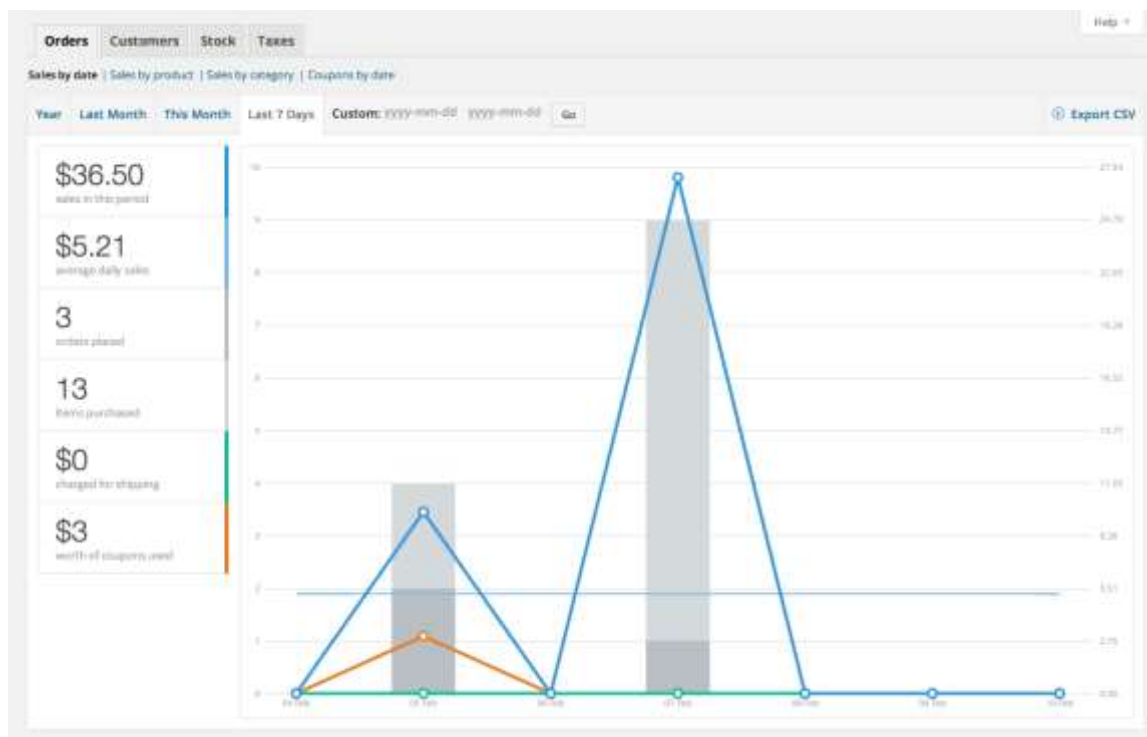
Предлага безплатна доставка, доставка с фиксирана ставка, или изчисление в реално време. Можем да ограничим доставките до конкретни държави, или да отворим магазина си за целия свят. Доставките съдържат множество настройки (Фигура №38). WooCommerce дори поддържа директни доставки от производителя.

WooCommerce предлага широка възможност от начини за плащане - поддържа всички основни видове кредитни карти, PayPal, BACS (банкови трансфери) и наложен платеж. Повече от 140 регионални методи за разплащане предлагат интеграция с WooCommerce, включително популярни методи като Stripe, Authorize.net и Amazon Payments.



Фиг. №38 Опции на разширението Woo Commerce

Автоматичната инсталация е най-лесната възможност, тъй като WordPress управлява файловите трансфери и няма нужда да напускаме браузъра си. За да извършим автоматична инсталация на WooCommerce, влизаме в администрацията на WordPress и от менюто „Разширения“ избираме "WooCommerce". На Фигура №39 е представена примерна графика на продажбите в менюто с опции на пългина:



Фиг. №39 Графика на продажбите

Лесен за разширяване и адаптиране, и с отворен код - WooCommerce е създаден с мисъл за разработчиците. С неговата мощна и гъвкава платформа можем да създаваме всякакви електронни магазини - от малки до огромни. Разработен с REST API, WooCommerce може да се интегрира с всяка външна услуга. Данните на нашия магазин могат да бъдат достъпни навсякъде, по всяко време, със 100% сигурност. WooCommerce позволява на разработчиците лесно да създават и променят магазин, който надминава техните очаквания. Без значение от размера на магазина, който искаме да създадем, WooCommerce ще оправдае нашите изисквания. С постоянно растяща колекция от повече от 300 разширения



Фиг. №40 Визуализация на примерна страница

ние можем да добавим всякаква функционалност, за да оправдаем очакванията на клиента си (Фигура №40). Можем дори да създадем наше собствено решение.

2.3.7 Yet Another Related Posts Plugin (YARPP) – с този плъгин може да задържим вниманието на посетителите за по-дълго. Това разширение поставя автоматично под всяка публикация няколко сродни публикации, които биха могли да заинтересуват читателя. Само изчислява кои сродни публикации да покаже. При по-големи сайтове това разширение може да доведе до проблеми с бързодействието.

Основни характеристики YARPP плъгин:

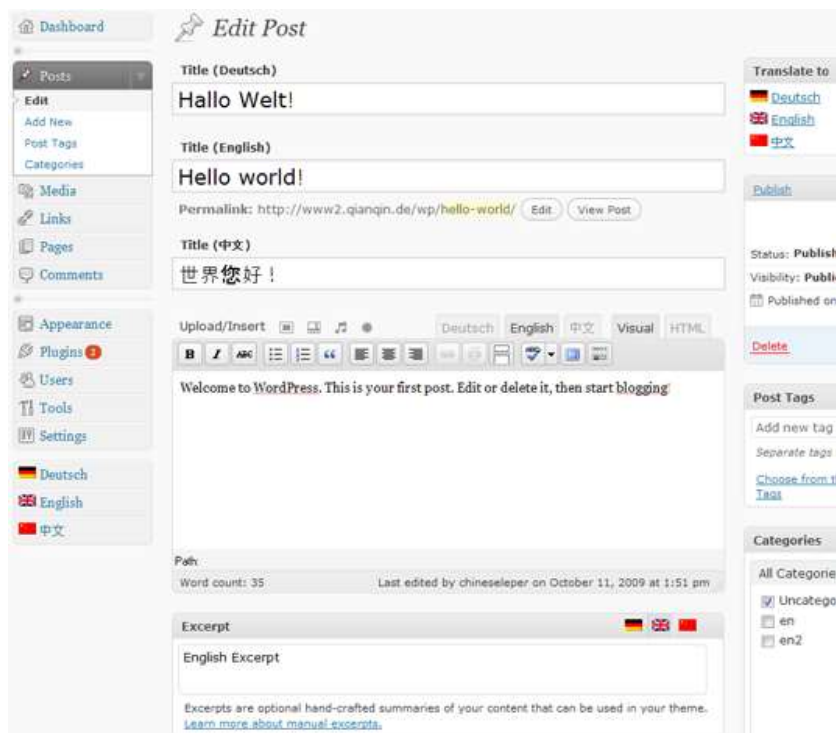
- Шаблони. Имаме много възможности за проектиране от показан списък с възможности.
- Свързва подобни постове, страници и входни данни.
- Той използва интегриран адаптивен алгоритъм, който отчита наименованието на постове, текстовете, етикети, категории, и таксономия. Приставката Yet Another Posts Related Plugin ще ни помогне да намерим съдържание, свързано по някакъв начин.
- Кеширане. Всички данни по сходни позиции се кеширват, което подпомага производителността на сайта.
- Показване на подобни постове в RSS. Всъщност, можем да информираме нашите читатели при наличието на нов пост чрез RSS.
- Изключване на определени етикети и категории.

2.3.8 qTranslate и WPML (The WordPress Multilingual Plugin) – тези плъгини използваме, когато имаме нужда сайтът ни да се преведе на един или няколко езика. Плъгинът qtranslate е безплатен, но понякога преводът не е достатъчно добър, WPML е платен и много по-добър вариант.[4]

2.3.8.1 qTranslate – След като свалим плъгина от тук <https://wordpress.org/plugins/qtranslate-x/installation/>, трябва да следваме инструкциите за инсталирането му:

- Трябва да качим цялата папка qTranslate в директория wp-content/plugins/ на нашия сървър
- Влизаме в WordPress с администраторския ни профил
- qTranslate трябва да е видим в списъка с възможни за активиране плъгини
- Активираме qTranslate
- В Настройки ще се появи нова връзка – qTranslate, от където можем да настроим плъгина на необходимите езици (Фигура №41)
- Добавяме си Widget (буквално „джаджа“) qTranslate, за да позволим на посетителите да избират предпочитания от тях език.

След което следва по-трудното – да напишем сайт на много езици.

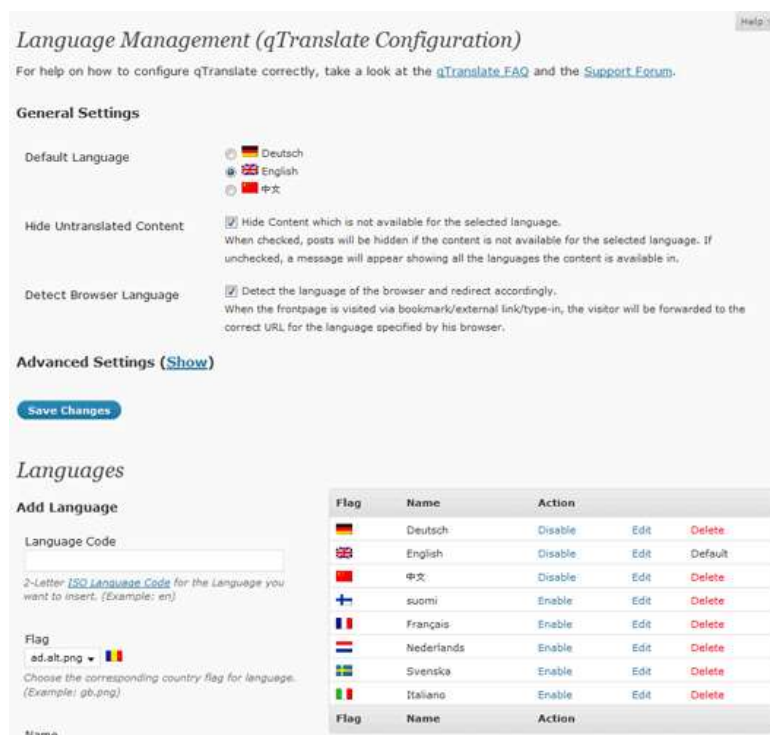


Фиг. №41 Настройки за превод на текста

Още няколко уточнения:

- qTranslate работи с версии 2.6 до 2.6.3 на WordPress

- за всеки език, който искаме сайтът ни да има, следва да добавим и съответните .mo файлове в директория wp-content/languages/
- за повече информация, осъвременяване и т.н., посетете сайта на автора на qTranslate (Фигура №42) – <http://www.qianqin.de/qtranslate/>



Фиг. №42 Настройки на qTranslate

2.3.8.2 WPML – <https://wpml.org/>

С WPML можем да превеждаме страници, записи, потребителски типове обекти, таксономия, менюта и дори текстовете. Всяка тема или плъгини, която използва WordPress API, се превръща в многоезична с WPML. Тъй като плъгинът е платен, в него се предлага и пълен support за WPML, за да можем да направим перфектния многоезичен сайт.

Извод

В Глава втора е изяснено, че програма или група от функции, написани на PHP, която добавя допълнителни функционалности към WordPress сайт или блог, се нарича WordPress плъгин. Постъпково е показано как се инсталира плъгин в

WordPress през wp-admin панел. Направено е описание на популярните и добавящи най-добри подобрения плъгини, а именно:

- Jetpack съчетава голям набор от полезни приспособления като ежедневна статистика за посещаемостта на сайта/блога; конзола за коментари; вградена галерия; форма за контакти; бутони за „Споделяне“ в социалните мрежи; практична тема за мобилен телефон
- WordPress SEO by Yoast е плъгин, който помага за оптимизирането на текстовете и съдържанието на сайта по ключови думи за по-добро позициониране в Google
- Contact form 7 е плъгин за добавяне на форма за контакти в сайта, която съдържа опция за активиране на допълнителен код за сигурност (reCAPTCHA)
- W3 Total Cache и Hyper Cache Extended са плъгини за кеширане на информацията, които спестяват време и ресурси, необходими за непрекъснатото генериране на информация на сайта
- Woo Commerce е разширение за електронна търговия
- YARP е плъгин, който автоматично генерира сродни публикации, които биха заинтересували потребителите на сайта
- qTranslate и WPLM са разширения за превод на сайта на няколко езика

Подробното описание на плъгините е полезно за базовото им използване и разширяване функционалността и потреблението на сайта.

III. Глава трета – Проектиране на plugin Слайдер

3.1 Основни неща, които трябва да знаем преди да проектираме нашия плъгин:[1]

Приложно-програмният интерфейс (на английски: Application Programming Interface, API) е интерфейсът на изходния код, който

операционната система или нейните библиотеки от ниско ниво предлагат за поддръжката на заявките от приложния софтуер или компютърните програми.

Приложно-програмният интерфейс предоставя един по-абстрактен и опростен план за разработчика на приложения, който би му спестил изучаването на няколко различни слоя от Операционната или софтуерната система зад интерфейса. По този начин се достига ефективност и бързина при адаптирането на нови софтуерни технологии. В миналото терминът се е използвал за обозначението на интерфейса между две програми.

- Shortcode API ([https://codex.wordpress.org/Shortcode API](https://codex.wordpress.org/Shortcode_API)) – това е „кутия“, оградена в квадратни скоби, която има специфично име и която може да регистрираме с плъгини, през теми или съществува в ядрото на WordPress и може да визуализира някакви данни

Например:

```
[gallery id="123" size="medium"]
```

е вграден shortcode в WordPress, който може да използваме, за да покажем снимки, прикачени към наш пост. За shortcode можем да посочваме и параметри, подобни на HTML таговете.

- Widgets API ([https://codex.wordpress.org/Widgets API](https://codex.wordpress.org/Widgets_API)) – widget са вградени функционалности в WordPress, като „Календар“, „Категории“, „RSS бутон“, „Search бутон“ и др., които можем да използваме в нашия сайт.
- Plugin API ([https://codex.wordpress.org/Plugin API](https://codex.wordpress.org/Plugin_API)) – hooks („хукове“) е място в кода в нашия плъгин, тема, в ядрото, на което можем да се закачим. Необходими са, защото ядрото е стандартизирано за определена категория блокове или сайтове, но чрез хуковете има вариант да променим някаква част от кода, която вече е генерирана.

Например:

- [add_action\(\)](#) – добавя функция към съществуващ „хук“
- [do_action\(\)](#) – изпълнява функция към съществуващ „хук“

- [add_filter\(\)](#) – филтрира данни
- [apply_filters\(\)](#) – филтрира данни

- HTTP API (https://codex.wordpress.org/HTTP_API) – позволява ни да се свързваме към външни услуги и да извикваме различни заявки
- Options API (https://codex.wordpress.org/Options_API) – записва данни, опции и стойности в базата от данни. Можем да го използваме, когато искаме да запишем код, който да валидираме и ако този код е в базата, плъгинът да не се отключва (пр. за платен плъгин); ако имаме регистрация и искаме ключовете да се записват в базата данни. В Options API се записват също така и всички опции в нашия сайт.

Например :

- [add_option\(\)](#)
- [get_option\(\)](#)
- [update_option\(\)](#)

- Setting API (https://codex.wordpress.org/Settings_API) – препоръчително е да се използва, когато submit-вате по-сложни теми с опции или admin panel-и в wordpress.org. Напълно се припокрива с Options API, с тази разлика, че трябва да пишем много повече код и сами да правим валидациите
- Transients API (https://codex.wordpress.org/Transients_API) – базира се на Options API, но има лимит от към време, използва се прозорче, което искаме да запишем временно, да имаме временен достъп до нещо (т.е. записва се опция, която е валидна за 10мин, 1 час и пр.). След изтичане на времето и след Refresh няма да имаме повече достъп до тази опция
- Rewrite API (https://codex.wordpress.org/Rewrite_API) – всички начини да направим „пътищата“(paths) по-разпознаваеми (използва се за SEO).

Дефинираме данните чрез:

- Custom Post Types (https://codex.wordpress.org/Custom_Post_Types) – служи за подреждане и надграждане на функционалностите в Post страниците
- Custom Taxonomies (https://codex.wordpress.org/Custom_Taxonomies) – създаваме Категории и Тагове, групираме постове или страниците по определени признаци. Например ако имаме пост с Категория „IT“, тагове са “CMS”, “WordPress” и др.
- Custom Fields (https://codex.wordpress.org/Custom_Fields) – добавяме допълнително съдържание, различно от стандартното: Заглавие, Описание, Снимка, User и дата, което не е задължително да е категоризирано, а може да е уникално за базата, която създаваме
- Meta boxes
(https://codex.wordpress.org/Plugin_API/Action_Reference/add_meta_boxes) – кутии, с които можем да групираме Custom Fields (например SEO пългините добавят доста Meta boxes)
- Page templates (<https://codex.wordpress.org/Pages>) – използва се, когато искаме някои от страниците да имат различна визуализация
- Load plugin text domain
(https://codex.wordpress.org/Function_Reference/load_plugin_textdomain) – функции, с които можем да зареждаме набор от текстове от интернационализацията (превода на WordPress на над 40 езика), така че пългинът или темата, която използваме да може да се преведе.
Използваме го, когато искаме да разберем и преведем тема или пългин на език, който не разбираме и не можем сами да преведем.

За работа с бази от данни трябва да разберем:

- WP Query (https://codex.wordpress.org/Class_Reference/WP_Query) – клас, който позволява да се задават заявки към базата от данни. Филтрира също така колоната от бази данни и ни връща заявка, която можем да ползваме
- Get posts (https://codex.wordpress.org/Template_Tags/get_posts) – прави заявка към базата от данни и връща масив с постове
- \$wpdb (https://codex.wordpress.org/Class_Reference/wpdb) – използва се за по-сложни заявки, работи на ниско ниво с базата

Пример:

```
$myrows = $wpdb->get_results( "SELECT id, name FROM mytable" );
```

- Pre get posts hook
(https://codex.wordpress.org/Plugin_API/Action_Reference/pre_get_posts) –извиква се след създаването на променлив обект на заявката, но преди действителното, заявката се изпълнява
- Posts where hook
(https://codex.wordpress.org/Plugin_API/Filter_Reference/posts_where) – този филтър се прилага, когато искаме да ограничим видимостта на нашите постове, те ще се показват в различни области на сайта, според условия, които зададем

Скриптове и стилове:

- Wpenqueue script
(https://codex.wordpress.org/Plugin_API/Action_Reference/wp_enqueue_scripts) – позволява да се качат всички скриптовете на определени хукове в WordPress. Използва се, когато използваме много плъгини, за да извикаме определен скрипт да се изпълни преди друг или в определена последователност
- Wpenqueue style
(https://developer.wordpress.org/reference/functions/wp_enqueue_style/)

– позволява да се закачат всички стилове на определени хукове в WordPress. Използва се, когато използваме много плъгини, за да извикаме определен стил да се изпълни преди друг или в определена последователност

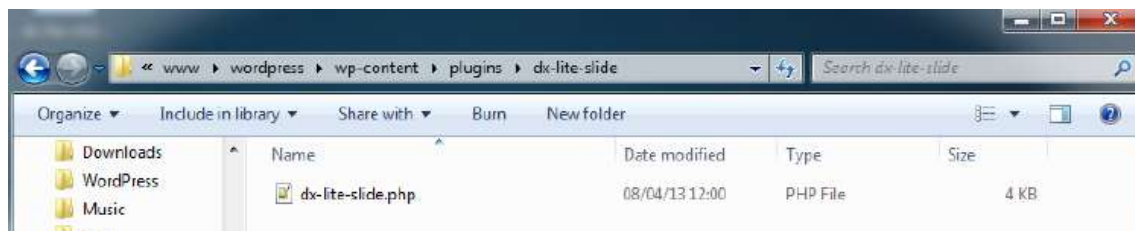
- Admin enqueue scripts

(http://codex.wordpress.org/Plugin_API/Action_Reference/admin_enqueue_scripts)

3.2 Да създадем плъгин от тип Слайдер за изображения

- Какво ще прави нашия плъгин, защо е по-добре да създадем отделен плъгин Слайдер, отколкото да го правим във functions.php на темата – защото при смяна на темата слайдовете няма да са видими, но ще бъдат в базата и излишно ще тежат в съдържанието. Затова по-практично е да използваме по-леки теми, към които да добавяме множество плъгини и функционалности.
- Да си направим план
- Да следваме най-добрите практики за писане на плъгини
- Да имаме инсталиран и работещ XAMPP / PHP, MySQL, Apache
- Да инсталираме чрез него WordPress

3.2.1 Първа стъпка – Дефиниране на съдържанието – В основната папка на инсталирания WordPress на нашия компютър (C:\xampp\htdocs\silvia-cms\wp-content\plugins) трябва да създадем папка с името на плъгина, който ще създадем (Фигура №43). В съответна среда, в която ще работим (Eclipse), създаваме файл dx-lite-slide.php, в който ще програмираме нашия слайдер.



Фиг. №43 Създаване на папка dx-lite-slide

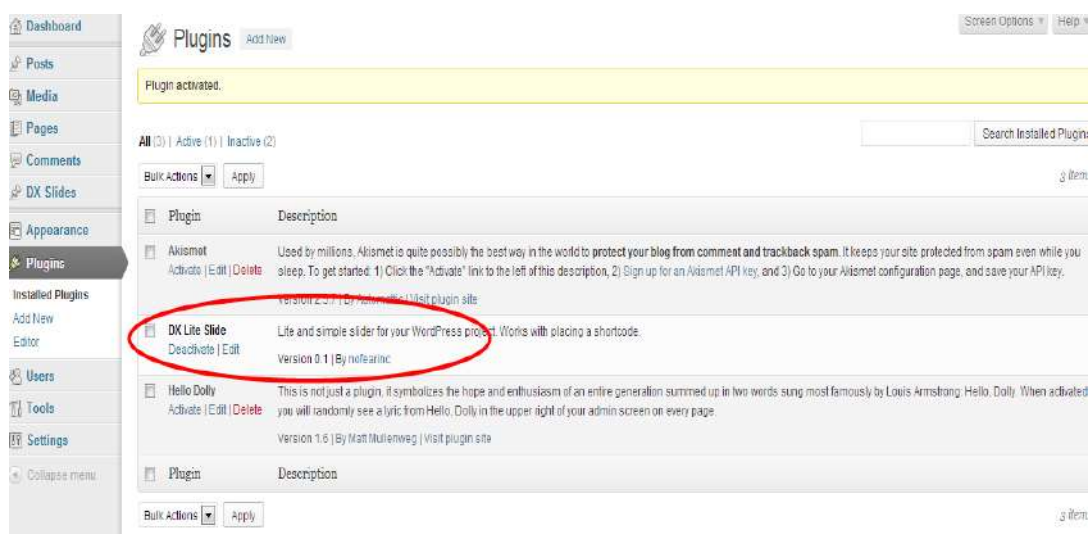
Започваме със заглавие, описание, авторство и версия на пългина, които поставяме в коментар, както е показано на Фигура №44.

```
<?php
/**
 * Plugin Name: DX Lite Slide
 * Description: Lite and simple slider for your WordPress project. Works with placing a shortcode.
 * Author: Silvia Gancheva
 * Author URI: http://silviagancheva.com/
 * Version: 0.4
 * License: GPLv2 or later

```

Фиг. №44 Начален код на пългина

Вече в нашия админ панел на WordPress на виртуалната машина се е появил пългинът, който създадохме (<http://localhost/silvia-cms/wp-admin/>) – Фигура №45.



Фиг. №45 Пългинът в админ панела на WordPress

Следваща стъпка е да дефинираме съдържанието на пългина, който искаме да създадем, което става чрез **custom_post_types**, **custom_taxonomies** и **custom_fields**. Регистрираме функция, която кръщаваме по определен начин и извикваме функция `register_post_type`, чрез която дефинираме име на типа, разни имена (labels) – името на пългина, единично име на пългина, какво се добавя при добавяне на нов пългин, описание на типа данни и други параметри:

```

public function register_dx_slides_cpt() {
    register_post_type( 'dx_slide', array(
        'labels' => array(
            'name' => __( 'DX Slides', 'dxls' ),
            'singular_name' => __( 'Slide', 'dxls' ),
            'add_new' => _x( 'Add New', 'pluginbase', 'dxls' ),
            'add_new_item' => __( 'Add New Slide', 'dxls' ),
            'edit_item' => __( 'Edit Slide', 'dxls' ),
            'new_item' => __( 'New Slide', 'dxls' ),
            'view_item' => __( 'View Slide', 'dxls' ),
            'search_items' => __( 'Search Slide', 'dxls' ),
            'not_found' => __( 'No slides found', 'dxls' ),
            'not_found_in_trash' => __( 'No slides found in Trash', 'dxls'
        ),
    ),
    'description' => __( 'Slides for the DX Lite Slide', 'dxls' ),
    'public' => true,
    'publicly_queryable' => true,
    'query_var' => true,
    'rewrite' => true,
    'exclude_from_search' => true,
    'show_ui' => true,
    'show_in_menu' => true,
    'menu_position' => 40, // probably have to change, many plugins
    use this
    'supports' => array(
        'title',
        'editor',
        'thumbnail',

```

```

        'custom-fields',
        'page-attributes',
    ),
    ));
}

```

С тази функция ние регистрираме определен тип данни с определени имена (labels), които са видими и поддържат следните функционалности – име, име на редактор, картинка (thumbnail или featured image), поддържа потребителски полета (custom-fields) и атрибути (page-attributes), ако са ни необходими.

3.2.2 Втора стъпка – Категоризация на слайдовете – това е необходимо, за да може да визуализираме слайдове от различни категории (напр. слайдер от категория „пейзажи“, друг от категория „животни“ и пр.)

```

public function register_dx_slides_tax() {
    register_taxonomy( 'dx_slider', array( 'dx_slide' ), array(
        'hierarchical' => true,
        'labels' => array(
            'name' => _x( 'Sliders', 'taxonomy general name', 'dxls' ),
            'singular_name' => _x( 'Slider', 'taxonomy singular name',
            'dxls' ),

            'search_items' => __( 'Search Sliders', 'dxls' ),
            'popular_items' => __( 'Popular Sliders', 'dxls' ),
            'all_items' => __( 'All Sliders', 'dxls' ),
            'parent_item' => null,
            'parent_item_colon' => null,
            'edit_item' => __( 'Edit Slider', 'dxls' ),
            'update_item' => __( 'Update Slider', 'dxls' ),

```

```

        'add_new_item' => __( 'Add New Slider', 'dxls' ),
        'new_item_name' => __( 'New Slider Name', 'dxls' ),
        'separate_items_with_commas' => __( 'Separate Sliders with
commas', 'dxls' ),
        'add_or_remove_items' => __( 'Add or remove Slider', 'dxls'
),
        'choose_from_most_used' => __( 'Choose from the most
used Slider', 'dxls' )
    ),
    'show_ui' => true,
    'query_var' => true,
    'rewrite' => true,
));

register_taxonomy_for_object_type( 'dx_slider', 'dx_slide' );
}

```

Тук имаме функция **register_taxonomy**, която регистрира категория “dx_slider”, която е свързана с нашия “dx_slider” пост тип данни и има йерархия (категоризация) . Позволен е юзер интерфейс, който да използваме и чрез “rewrite”, може да виждаме опциите, преглеждайки в браузера.

След което дефинираме следната функция-конструктор, където закачаме функциите за регистрирането на пост типовете и на категориите:

```

public function __construct() {
    add_action( 'init', array( $this, 'register_dx_slides_cpt' ) );
    add_action( 'init', array( $this, 'register_dx_slides_tax' ) );
}

```

За целта използваме най-популярните хукове `,init‘` или `,admin_init‘`, в които можем добавяме нова функционалност (widgets, post_type, taxonomies, custom_fields), които да се регистрират при цялостната инициализация на WordPress.

3.2.3 Трета стъпка – Добавяне на опции – да създадем файл `dx-lite-slide-options.php` – където ще направим всички опции за нашия слайдер. Опциите се създават за връзка между WordPress темите и плъгините или за връзка между плъгин и плъгин или ако искаме плъгинът да задава (контролира) нещо в нашата тема. Във функцията `<?php $dxls_options = get_option('dxls_slide_options'); ?>`, `,get_option‘` се свързва с `,wp_options‘` папката на нашия `localhost/php myadmin` и взима стойността, която искаме да достъпим; `,add_option‘` – добавя нови стойности; `,update_option‘` – ъпдейтва нови стойности.

С кода изписан по-долу добавяме кутийки с определена функционалност и визуализация:

```
<form method="post" action="">
    <label for="width">Slider width (use metrics too - px, em, %)</label>
    <input type="text" id="width" name="width" value="<?php if ( ! empty(
$dxls_options['width'] ) ) esc_attr_e( $dxls_options['width'] ); ?>" /><br />
    <label for="height">Slider height (use metrics too)</label>
    <input type="text" id="height" name="height" value="<?php if ( ! empty(
$dxls_options['height'] ) ) esc_attr_e( $dxls_options['height'] ); ?>" /><br />
    <label for="interval">Slider Interval (in milliseconds)</label>
    <input type="text" id="interval" name="interval" value="<?php if ( ! empty(
$dxls_options['interval'] ) ) esc_attr_e( $dxls_options['interval'] ) ?>" />
    <span>* Default interval is 4000</span><br />
    <label for="order">Slider Order</label>
```

```

<select id="order" name="order">
    <?php
    if ( $dxls_options['order'] == 'desc' ) {
        echo '<option value="desc" selected="selected">DESC</option>';
        echo '<option value="asc">ASC</option>';
    } else {
        echo '<option value="desc">DESC</option>';
        echo '<option value="asc" selected="selected">ASC</option>';
    }
    ?>
</select>

<p class="submit">
    <input type="submit" name="dxls_slide_options_submit" class="button-
primary" value="<?php _e('Save Changes', 'dxls'); ?>" />
</p>
</form>

```

3.2.4 Четвърта стъпка – Визуализиране на страница с опции – да свържем страницата за опциите с нашия основен файл dx-lite-slide.php

```

public function register_dx_slide_options_page() {
    add_submenu_page( 'edit.php?post_type=dx_slide', 'DX Slides Options',
'DX Slides Options', 'edit_themes', 'dx_slides_options', array( &$this,
'dx_options_submenu_page_callback' ) );
}

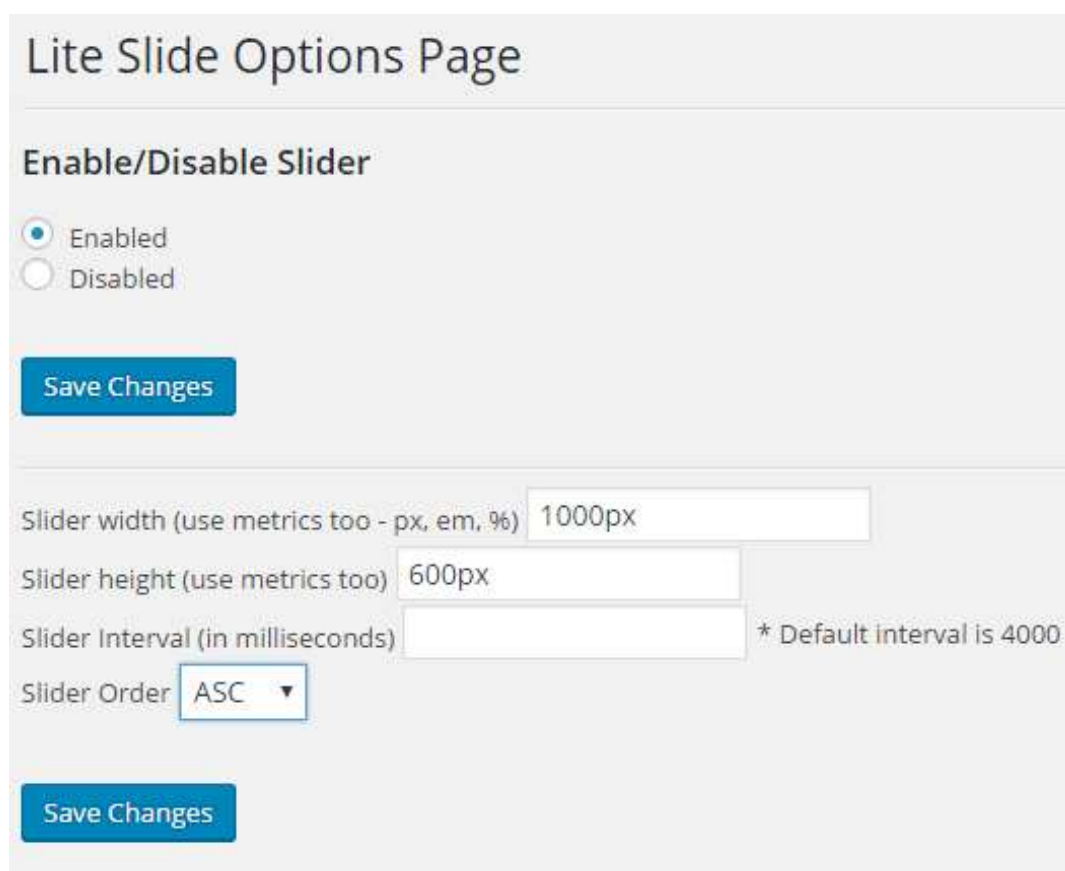
public function dx_options_submenu_page_callback() {
    include_once 'dx-lite-slide-options.php';
}

```

Имаме функция, която регистрира меню страница и друга, с която извикваме файл, в който са опциите. Тази функция ще закачим за **admin menu** хука чрез следния код:

```
add_action( 'admin_menu', array( $this, 'register_dx_slide_options_page' ) );
```

На Фигура №46 е представено как ще изглежда страницата с опциите в WordPress:



Фиг. №46 Lite Slide Options Page

Имаме опция Активиране/Деактивиране на слайдера, опция за промяна на ширината и височината на слайдера, опция за промяна на интервала за смяна на изображенията и дали да се показват в низходящ, или възходящ ред.

3.2.5 Пета стъпка – Добавяне на съдържание – да добавим някакво съдържание (снимки) на нашия слайдер в localhost/silvia-cms/wp-admin.

Добавям пейзажни снимки в Категория ‚Nature‘ и няколко снимки на птици в Категория ‚Birds‘. Можем да създаваме много и различни категории слайдери, които после ще визуализираме на различни места в страниците ни или на заглавната страничка.

3.2.6 Шеста стъпка – Създаване на слайдер – да направим файл `display-slideshow.php`, който ще съдържа `shortcode`, който чете данните, които имаме и ги визуализира в слайдшоу. В този файл взимаме данните, които сме записали във файла с настройките и ги използваме по определен начин.

Тук, например, задаваме относително статични аргументи:

```
$dxls_slide_args = array(
    'post_type'      => 'dx_slide',
    'post_status'    => 'publish',
    'posts_per_page' => -1,
    'orderby'        => 'date',
    'order'          => 'DESC'
);
```

Задаваме ‚`post_type`‘ да е ‚`dx_slide`‘, ‚`post_status`‘ да е ‚публикувано‘, ‚`post_per_page`‘ задава колко поста да се показват на страница (-1 показва, че не искаме странициране, искаме да виждаме всички данни, които на нас ни трябват), ‚`orderby`‘ показва, че ще ги подреждаме по дата, ‚`order`‘ сочи, че ще е в намаляващ ред.

Ще разгледаме и следващата част от кода:

```
if ( $dxls_options['dxls_slide_status'] == 'enabled' ) :
```

```
    $dxls_slides_list = new WP_Query( $dxls_slide_args );
```



```

        if ( $dxls_slides_list->have_posts() ) :
            echo '<div class="dx-slideshow" style="display: inline-block; float: ' .
            $args["position"] . '; width: ' . $args["width"] . '; height: ' . $args["height"] . ';">';
            while ( $dxls_slides_list->have_posts() ) : $dxls_slides_list-
            >the_post();
                $dxls_img_url = wp_get_attachment_url(
                get_post_thumbnail_id( $post->ID ) );
                echo '<div>';
                echo ' ';
                echo '</div>';
            endwhile;
            echo '</div>';
        endif;
        wp_reset_query();
    endif;

```

Правим проверка: ако от admin-а тази опция 'dxls_slide_status' е 'enabled' – да покажем слайдера, в противен случай – да не го показваме. Показваме го като извикваме `WP_Query` с тези параметри `$dxls_slide_args`. Тогава ако имаме някакви слайдове, показваме `<div class="dx-slideshow"` с определени параметри, които контролираме. След това създаваме цикъл: докато има слайдове, ще се показват един след друг по `$post->ID`. С този код взимаме пътя на картинката `$dxls_img_url = wp_get_attachment_url(get_post_thumbnail_id($post->ID));`, а с този - `echo '';` го слагаме в картинка с определени параметри за всички от тях, независимо от първоначалните им размери.

След което, трябва да регистрираме shortcode-а в нашата основна страница dx-lite-slide.php :

```
public function add_dx_slides_shortcode() {  
    add_shortcode( 'dx_display_slideshow', array( $this,  
'dx_display_slideshow' ) );  
    add_shortcode( 'dx_widget_display_slideshow', array( $this,  
'dx_widget_display_slideshow' ) );  
}
```

И да го извикаме в нашия конструктор в същия файл:

```
add_action( 'init', array( $this, 'add_dx_slides_shortcode' ) );
```

3.2.7 Седма стъпка – JavaScript и CSS – да добавим CSS и JavaScript, за да стилизираме слайдера. Добавяме в основния код следните функции :

```
public function dx_enqueue_style_css() {  
    wp_enqueue_style( 'style.css', plugins_url( '/styles/style.css', __FILE__ ) );  
    wp_enqueue_style( 'style.css' );  
    public function dx_enqueue_slider_script() {  
        wp_enqueue_script( 'dx_slide', plugins_url( '/js/dx-slide.js', __FILE__ ), array(  
'jquery' ) );  
    }
```

След което ги добавяме в основния конструктор :

```
add_action( 'wp_enqueue_scripts', array( $this, 'dx_enqueue_style_css' ) );  
add_action( 'wp_enqueue_scripts', array( $this, 'dx_enqueue_slider_script' ) );
```

Add_action прави следното : някъде в кода слагаме празно пространство, където `add_action` може да прави, каквото сме му задали (да извиква `div`, `php` – които генерират неща и др.)

Следваща стъпка е да създадем на нашия компютър две папки ‘`styles`’ и ‘`js`’, в които да създадем съответно файлове `style.css` и `dx-slide.js`.

Стиловете са на нашия основен компонент `dx-slideshow`:

```
.dx-slideshow {  
    display: inline-block;  
    position: relative;  
    width: 100%;  
    max-width: 960px;  
    height: 350px;  
    margin: 50px auto;  
    padding: 10px;  
    box-shadow: 0 0 20px rgba(0, 0, 0, 0.4);  
}
```

Позицията е `relative` (относителна), за да може да я променим след това; ширина : 100%; максимална ширина : 960px; височина : 350px; отстояние и сянка на изображението. След това задаваме параметри на кутията (`div`), в която ще се помести снимката:

```
.dx-slideshow > div {  
    position: absolute;  
    top: 10px;  
    left: 10px;  
    right: 10px;  
    bottom: 10px;
```

```
}
```

Отново позицията е относителна, за да можем да я променяме и от всички страни на изображението оставяме по 10px до кутията.

JavaScript казва следното: ако имаме повече от едно изображение, тогава да започва анимация на самия JavaScript – изображенията ще се сменят едно след друго с лек елемент на избледняване; ако изображението е само едно – ще се показва само то:

```
jQuery(document).ready(function($)  
if ( jQuery( '.dx-widget-slideshow img ' ).length > 1 ) {  
    jQuery( '.dx-widget-slideshow > div:gt(0)' ).hide();  
    setInterval( function() {  
        jQuery( '.dx-widget-slideshow > div:first' )  
            .fadeOut( 1000 )  
            .next()  
            .fadeIn( 1000 )  
            .end()  
            .appendTo( '.dx-widget-slideshow' );  
        }, 4000);  
    }  
});
```

3.2.8 Осма стъпка – Добавяне на пост/страница със слайдера – създаваме пост или страница или Widget в WordPress, в който поместваме слайдера.

Създаваме пост или страница или Widget в WordPress, в който поставяме следния shortcode:

[dx_display_slideshow]

В различните страници, постове и widgets може да задаваме и различни компоненти на слайдера. Например в widget искаме да е с по-малки размери:

[dx_widget_display_slideshow width="150px" height="150px"]

Можем да променяме ширината и височината на слайдера благодарение на този код в **display-slideshow.php**, в който на мястото на **width** и **height** се обръщаме към аргументите, които сме задали:

```
echo '';
```

3.2.9 Девета стъпка – Лицензиране (Стандарти) – Прието е да се следва стандартната заглавната част с информация за лицензиране при разработване и публикуване на плъгин в WordPress. Повечето плъгини използват GPL2 License или такива, които са сходни с GPL2 - <http://www.gnu.org/licenses/license-list.html>. За да отбележим, че използваме лиценз GPL2, трябва да включим редовете от Фигура №47 в нашия плъгин:

```
<?php  
/* Copyright YEAR PLUGIN_AUTHOR_NAME (email : PLUGIN_AUTHOR_EMAIL)  
  
This program is free software; you can redistribute it and/or modify  
it under the terms of the GNU General Public License, version 2, as  
published by the Free Software Foundation.  
  
This program is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
GNU General Public License for more details.  
  
You should have received a copy of the GNU General Public License  
along with this program; if not, write to the Free Software  
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA  
*/  
?>
```

Фиг. №47 Стандарт за лицензиране

3.2.10 Десета стъпка – Визуализация – плъгинът е готов!

Можем да го използваме в WordPress с желаните от нас изображения в избраната тема – Фигура №48.





Фиг. №48 Визуализация на Слайдера

Извод

В Глава трета кратко са описани основните компоненти, необходими за проектирането на плъгини. Тъй като WordPress е система с отворен код, тя предлага приложно-програмен интерфейс (API), генериращ код, който позволява да планираме изграждането на приложения по-бързо, без да се изучават слоевете на операционната или софтуерната система. WordPress съдържа Shortcode API, Widgets API, Plugin API, Options API и др., чрез които се постига ефективност и бързина при адаптирането на нови софтуерни технологии. В Главата са обяснени причините за дефиниране на данните в

WordPress, принципите при работа с бази от данни и начините за добавяне на скриптове и стилове. На базата на тези познания е програмиран плъгин **Слайдер**, който визуализира сменящи се едно след друго изображения. Добавени са различни ефекти, категоризация, опции и widget слайдер, които подобряват функционалността на плъгина. **Слайдерът** е независим от темата, която използваме и може да се използва към произволна такава. Показани са стъпките, по които е направен плъгинът, както и крайният резултат при използването му.

Заклучение

Разработената дипломна работа на тема: “Приложение на плъгините /разширенията/ в CMS платформата WordPress” представлява цялостен и завършен проект, отразяващ процеса на анализ на CMS системата и нейните разширения, проектиране и реализация на плъгин за изображения. Работата премина през няколко етапа.

В Глава първа са обяснени предимствата на CMS ситемите като: ниски разходи по поддръжка, съгласуваност, по-добра навигация, повече сигурност и др., които могат да доведат до внедряване и използване на CMS. В главата са обяснени понятията: Система за управление на съдържанието; Жизнен цикъл на електронното съдържание; Основни роли при управление на съдържанието. WordPress има трислойна архитектура, състояща се от Система за управление на бази данни, Изпълними скриптове и Шаблони. Използва скриптов езици като PHP, Perl, ASP, Python и др. за взаимодействие с данните и преобразуването им в HTML страници или други документи. В Главата е направен сравнителен анализ на съществуващите CMS системи – WordPress, Joomla, Python, Drupal, OpenCart и Magento, на базата на който е избран

WordPress, заради лесната употреба, интуитивния интерфейс и бързата инсталация. Показано е постъпково инсталиране на WordPress на определен сървър и на нашия компютър, базистно боравене с директориите и инсталиране на тема (интерфейс на системата).

В Глава втора е изяснено, че програма или група от функции, написани на PHP, която добавя допълнителни функционалности към WordPress сайт или блог, се нарича WordPress плъгин. Постъпково е показано как се инсталира плъгин в WordPress през wp-admin панел. Направено е описание на популярните и добавящи най-добри подобрения плъгини, а именно: Jetpack, WordPress SEO by Yoast, Contact form 7, W3 Total Cache и Hyper Cache Extended, Woo Commerce, YARP, qTranslate и WPLM. Подробното описание на плъгините е полезно за базовото им използване и разширяване функционалността и потреблението на сайта.

В Глава трета е програмиран плъгин **Слайдер**, който визуализира сменящи се едно след друго изображения. Добавени са различни ефекти, категоризация, опции и widget слайдер, които подобряват функционалността на плъгина. **Слайдерът** е независим от темата, която използваме и може да се използва към произволна такава. Показани са стъпките, по които е направен плъгинът, както и крайният резултат при използването му.

В дипломната работа се описват някои основни концепции при програмирането на плъгини за WordPress. Посочват се част от възможностите на системата с отворен код както и се следват добрите практики за програмиране. Заключение, което може да се направи е, че целта на дипломната работа е изпълнена. Изработен е функционален плъгин за WordPress, от типа Слайдер за изображения. Използван е семпъл дизайн, подходящ за съвременен сайт, като предоставя лесен за използване интерфейс и работи добре с различни теми на WordPress.

Дипломната работа може да послужи на хора, занимаващи се с WordPress, начинаещи разработчици на плъгини и теми за системата, както и на WordPress програмисти с повече опит, които биха искали да обогатят знанията си.

Насоки за бъдещо развитие и усъвършенстване:

Осигуреният плъгин предоставя само основните функции подходящи за една такава система. За бъдещо развитие могат да се разработят следните функционалности:

- Модул за SEO оптимизация на снимките на **Слайдера**
- Възможност за добавяне на линкове към изображенията при задържане на мишката върху тях
- Добавяне на динамични плочки със стрелки за движение в ляво и дясно на изображенията
- Добавяне динамични визуални елементи

Използвана литература:

Печатни издания:

- [1] Гутманс, А., С. Баккен, Д. Ританс. PHP 5 Професионално програмиране. 2005.
- [2] Ернандес, М. Проектиране на бази от данни. 2004.
- [3] Уелинг, Л., Л. Томсън. Разработване на проекти за Web PHP и MySQL. 2004.

Интернет адреси:

- [4] Сайтът на WordPress - <https://wordpress.org/>
- [5] All articles about CMS - <http://cmsarticles.awardspace.com/>
- [6] A review of open source content management systems - <http://www.openadvantage.org/articles/oadocument.2005-04-19.0329097790>
- [7] CMS Watch: Content Management, Enterprise Search and Portal Reports - <http://www.cmswatch.com/>
- [8] CMS Wiki - <http://www.cmswiki.com/tiki-index.php>
- [9] Drupal – <https://www.drupal.org/>
- [10] How to choose a web CMS - <http://livestoryboard.com/CMS-Resources/How-to-choose-a-web-CMS.html>
- [11] How to evaluate a content management system - http://www.steptwo.com.au/papers/kmc_evaluate/
- [12] Joomla – <https://www.joomla.org/>
- [13] Magento – <https://magento.com/>
- [14] MySQL AB :: The world's most popular open source database - <http://mysql.com/>
- [15] OpenCart – <http://www.opencart.com/>
- [16] PHP: Hypertext Preprocessor. - <http://php.net/>
- [17] So, What Is CMS - http://www.steptwo.com.au/papers/kmc_what/index.html
- [18] The Content Management Comparison Tool - <http://www.cmsmatrix.org/>