**ТЕХНОЛОГИЧНО УЧИЛИЩЕ “ЕЛЕКТРОННИ СИСТЕМИ”**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

Тема: WordPress framework за потребителски планове и разплащания в Multisite среда.

Дипломант: Научен ръководител:

*Пламен Павлов Марио Пешев*

СОФИЯ

2020

Дата на заданието: 15.11.2019 г. Утвърждавам:..............................

Дата на предаване: 15.02.2020 г. /проф. д-р инж. Т. Василева/

**ЗАДАНИЕ**

**за дипломна работа**

на ученика Пламен Пламенов Павлов 12А клас

1.Тема: WordPress framework за потребителски планове и разплащания в Multisite среда

2.Изисквания:

1) Поддръжка на онлайн разплащания през PayPal към съществуващ WordPress уебсайт

2) Регистрация на нов WordPress сайт в Multisite среда спрямо плащане по предварително зададени планове

3) Конфигуриране на активни плъгини спрямо регистрирания план

4) Настройки на плана от админ панела

5) Динамично изтегляне на цените от sales сайт

3.Съдържание 3.1 Обзор

3.2 Същинска част

3.3 Приложение

Дипломант :...........................................

Ръководител:..........................................

/ Марио Пешев /

Директор:................................................

/ доц. д-р инж. Ст. Стефанова /

# Отзив на научния ръководител и предложение за рецензент

За изготвянето на дипломната си работа, Пламен се запозна с нови програмни езици (PHP и JavaScript), системата за управление на съдържанието WordPress, разработка на Multisite проекти, както и външни системи за разплащане като PayPal.

Работата по нея изискваше управление на две различни системи и интеграцията между тях, както и разработката на системата за разплащания и автоматично създаване на нови сайтове.

Въпреки някои проблеми при изграждането на определени изисквания от заданието, Пламен успя да реализира основните компоненти по интеграцията на системите и създаването на планове за нови сайтове.

Изграждането на самия софтуер позволи на Пламен да получи важен практически опит, който ще му е от полза при разработка на други уеб проекти, особено в контекста на PHP или WordPress.

Научен ръководител:

/Марио Пешев/

# УВОД

WordPress framework най-често се отнася до библиотека код, която прави възможно създаването на теми, плъгини, различни видове системи по по-лесен и удобен начин. Най-общо казано те са основата на една WordPress тема. Обикновено framework съдържа основният код, който обхваща главните функционалности и елементи на дизайна. След като бъде създаден framework, той може да бъде използван като начална точка за нова WordPress тема. Правят се нови промени във функционалностите и дизайна, за да може да бъде създаден добър краен продукт. Има два главни вида WordPress theme framework:

* **In-house framework** – този тип са създадени, за да бъдат ползвани от специални компании, които създават теми (като **TeslaThemes** например). Всяка тема е създадена на базата на техния framework.
* **Всички останали –** могат да бъдат използвани от всеки като основа на проект, като варират от безплатни(като **Beans**) до платени(например **Genesis**).[1]

Важно е да се спомене, че при ползване на framework, необходима е и child WordPress тема – причината за това е, че даден framework осигурява необходимите актуализации, подобно на parent темите. При актуализации на framework няма опасност от изгубване на персонализирани промени, които са направени по кода преди това.

Системите за разплащания са важна част от всяка онлайн търговия. Те могат да бъдат определящи във финализирането на дадена покупка. Трябва да бъде избран подходящ метод за разплащане, според клиентите. Най-известната система за разплащания в света е PayPal, докато в България най-популярни са EasyPay и ePay. Една от най-важните особености на една такава система е нейната сигурност – за да бъдат защитени личните данни на клиентите.

**Предимства на електронните плащания:**

* **Спестяване на време –** не е необходимо време в посещаване на банкови институции за осъществяване на паричен превод.
* **Достъпност –** потребителите могат да закупуват продукти по всяко време.
* **Контрол на разходите –** по всяко време може да бъде прегледана историята на плащанията, извършени през Интернет.
* **Повишаване на продажбите** – при наличие на лесен и удобен уебсайт, клиентите се увеличават.[2]

Целите на настоящата дипломна работа са:

* Възможността за поддръжка на разплащания през PayPal към съществуващ WordPress сайт.
* Регистрация на нов WordPress сайт в Multisite среда спрямо плащане по предварително зададени планове.
* Спрямо избраният и платен план да бъдат конфигурирани съответните активни плъгини.
* Настройки на плана от админ панела
* Динамично изтегляне на цените от sales сайт

Тази дипломна работа се фокусира върху улесняването при закупуване на нов WordPress сайт – спрямо избрания и заплатен план(например стандартен и премиум) клиентът получава съответните функционалности и дизайн на сайта, включени в този пакет.

# ПЪРВА ГЛАВА

СЪЩЕСТВУВАЩИ ПОДОБНИ ПРИЛОЖЕНИЯ И СИТЕМИ ЗА РАЗПЛАЩАНИЯ

# Съществуващи решения и подобни реализации

**Stratus5**

**Stratus5** е начело на облачната технология за автоматизация. Компанията е основана през 2008г. и има възможността да захранва следващото поколение cloud приложения и да осигури основата на автоматизация за бъдещето. Със Stratus5 разработчиците на апликации получават всички технически и бизнес cloud възможности, необходими за стартиране, управление, разпространение и осигуряване на приходи от cloud приложения. [3]

Stratus5 предлага на доставчиците и дистрибуторите на софтуерни приложения облачна (cloud) бизнес платформа, която им позволява бързо да стартират и управляват своя SaaS (Software as a Service е модел за лизенциране и доставка на софтуер, при който софтуерът е лизензиран на базата на абонаменти и се хоства централно [4]) бизнес без предварителни разходи. Stratus5 дава възможност на доставчиците на приложения бързо да трансформират всяка еднократна уеб апликация, изградена на каквато и да е технология, в многоинстанционна облачна услуга в рамките само на няколко дни, без да е необходима голяма промяна в продукта.

Облачната Stratus5 бизнес платформа предлага пълно решение, включващо всички бизнес и технически възможности, необходими за работата на SaaS бизнес – по този начин дава възможност на доставчиците на приложения да се съсредоточат напълно върху апликациите и бизнеса си без да се притесняват относно основната облачна технология.

**Акценти на решения:**

* Пълно решение за обслужване на white-label(това е продукт или услуга, произведена от една компания, която бива ребрандирани от други компании, за да изглежда сякаш е тяхна [5]), за да стартирате SaaS бизнеса си без предварителни разходи. Понякога се нарича „софтуер при поискване“, а преди – „софтуер плюс услуги“ от Microsoft.
* Облачно внедряване (всяко приложение във всеки стек) - „SaaS“ потвърждава всяко уеб приложение, изградено на всеки стек, без да се изисква кодиране на приложението.
* Управление на продукта – включва конфигурация на продукта, управление на цените и рекламата.
* Управление на клиентите – пълно управление на жизнения цикъл на клиента, включващо изпробване, преобразуване и текущи комуникации с клиенти.
* Управление на канали – набиране на многостепенни дистрибутори и управление на продажбите.
* Управление на приходите - автоматизирано таксуване, обработка на плащания.
* Управление на клиентската поддръжка – включва управление за поддръжка на клиенти и директен достъп до потребителски инстанции за разрешаване.
* Управление на приложения – включва много облачни услуги за приложения, включително автоматизирано предоставане на приложения, управление на актуализациите, управление на домейна, персонализиране на приложения, open integration framework, управление на производителността и други.
* Бизнес анализ – включва набор от предварително изградени отчети за клиенти, канал, приложения, приходи, поддръжка и други.
* Напълно управлявана услуга за облачна инфраструктура – базирана на мащабируема, надеждна и сигурна облачна инфраструктура, от която водещите доставчици зависят, за да осъществяват своя бизнес. Включва управление на ресурсите, управление на сигурността, услуги за съобщения, архивиране и възстановяване и други облачни услуги.
* Едно единствено табло за управление на всичко – една конзола за наблюдение и управление на инфраструктура, приложения, клиенти, партньори и облачния бизнес.

Stratus5 има десетилетие опит в операции на разработчици (DevOps), облачна автоматизация и организиране. От организиране на контейнери до API, до услуги за автоматизация на бизнеса. [6]

# 1.2. Системи за онлайн разплащания

С всеки изминал ден технологиите навлизат все повече и повече в нашето ежедневие – те имат за цел да улеснят всекидневието ни чрез различен вид помощ. Спестяват ни време, усилия, дори изместват хората в работата им. Именно с това развитие на техниката и Интернет навлизат и промени в системите за разплащания – в днешно време съществуват множество такива услуги. Много от нещата вече се случват онлайн – със съществуването на стотици, хиляди уебсайтове идва и необходимостта от онлайн плащания при покупка на даден продукт, заплащане на услуга или извършване на паричен превод и др. Възможността да контролираме и извършваме плащанията си онлайн ни осигурява голямо удобство – спестяваме си време в посещаване на банки например за извършване на дадено плащане. С тези системи можем и много лесно да наблюдаваме историята на онлайн плащанията ни. Част от най-известните такива системи в България са: EasyPay, ePay.bg, PayPal(която е и най-известната в световен мащаб). Алтернативни продукти на PayPal, които са популярни извън страната ни, са: Payza(AlertPay преди), WePay, Google Checkout, Escrow, Xoom, Payoneer, Skrill / Moneybookers и 2CO.

**ePay.bg**

**ePay.bg** е българска система за електронни плащания. Системата съществува от октомври 1999 година. През ноември същата година е реализирано и първото плащане онлайн в България именно през ePay.bg. Оператор на системата е *Ипей АД*, създадено през 2000 г., част от Датамакс Системс Холдинг АД.

**Цели на системата:**

* Да даде възможност на клиентите си да осъществяват месечните си плащания за битови сметки (ток, вода, парно, телефон, GSM и т.н.), да покриват данъчните си задължения, както и да пазаруват свободно, възползвайки се от предимствата на Интернет пространството – спестяване на време, нерви и средства. Те получават още и възможността да извършват нареждания за плащания през интернет и подробни извлечения по сметките си.
* Да даде възможност на българските търговци да получават плащания по Интернет, чрез национални дебитни и международни кредитни карти. ePay.bg може да се използва както от граждани на страната, така и от чужденци желаещи да закупят стоки или да се възползват от услуги, предлагани от българските фирми. [7]

**PayPal**

**PayPal** е онлайн платежна система на едноименната компания, която стартира самостоятелно през 1998 г., но през 2002 е закупена от eBay Inc.. PayPal дава възможност за трансфер на пари онлайн и представлява електронна алтернатива на традиционните разплащания с чекове и банкови преводи. През 2015 г. компанията става независима.

Компанията PayPal стартира самостоятелно през 1998 г., като сред основателите ѝ е Илон Мъск. През 2002 излиза на борсата и скоро след това е закупена от eBay Inc.. Към края на 2006 PayPal има повече от 123 милиона потребители, които могат да изпращат и получават пари в различни точки от света в 17 вида валута. От 10 май 2007 г. Paypal позволява изпращане на пари от България, а от 1 октомври 2007 г. българските потребители могат и да получават средства.

Чрез PayPal всеки човек, който притежава валиден е-мейл, може да превежда пари на друг човек с е-мейл. Инициаторът на превода трябва първо да си отвори сметка в PayPal и да я захрани със средства. Сметката в PayPal може да бъде обезпечена чрез кредитна, дебитна карта или чрез сметка в банка. За да получи парите, получателят може да изиска чек от PayPal, да си отвори сметка в PayPal или да посочи банкова сметка за превод на сумата. В този смисъл PayPal представлява посредник, който улеснява електронната търговия по света, без да разкрива финансова информация при сделките. Потребители са частни лица, търговци, големи фирми и корпорации (за последните се отчитат такси при паричен трансфер, за останалите услугите са безплатни). [8]

**Payoneer**

**Payoneer** е компания за финансови услуги, която осигурява онлайн парични преводи, услуги за дигитални разплащания.

Хора, притежаващи акаунти, могат да пращат и получават финансови средства, използвайки своя Payoneer e-wallet или презареждаща се предплатена MasterCard дебит карта. Компанията предлага транзакции между 200 страни и територии.

Компании като Airbnb, Amazon, Google и Upwork използват Payoneer за големи разплащания през цял свят. Също така системата се използва и от eCommerce магазини като например Wish.com, Walmart, Rakuten и др. Payoneer има екип от 320 служители, обслужващи 4 милиона клиенти на 70 различни езика, използвайки 150 различни валути. [9]

# ВТОРА ГЛАВА

ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА ПРИЛОЖЕНИЕТО И ПРИЧИНА ЗА ИЗБРАНИТЕ ТЕХНОЛОГИИ

# 2.1.**Функционални изисквания към WordPress базиран framework**

## **2.1.1. Поддръжка на онлайн разплащания през PayPal към съществуващ WordPress уебсайт.**

Необходима е интеграция на PayPal чрез тяхното API.

## **2.1.2. Регистрация на нов WordPress сайт в Multisite среда спрямо плащане по предварително зададени планове.**

В зависимост от избрания и заплатен план(например стандартен или премиум/скъп план) се регистрира нов WordPress сайт със съответните функционалности и плъгини, които се съдържат в избрания план. Например, ако имаме плъгини 1, 2, 3, 4, стандартният план включва пългини 1, 3, а скъпият план включва плъгини 1, 2, 3, 4, ако изберем стандартния план, получаваме съответните плъгини и функционалностите, които предлагат.

## **2.1.3. Конфигуриране на активни плъгини спрямо регистрирания план.**

В зависимост от избрания и платен от потребителя план, се конфигурират съответните плъгини, които дадения план включва.

## **2.1.4. Настройки на плана от админ панела.**

Да се дадат админ права и роля на потребителя и допълнителни настройки.

## **2.1.5. Динамично изтегляне на цените от sales сайт.**

Цените на плановете в приложението се определят в зависимост от цените в други подобни сайтове.

# 2.2. Използвани технологии и аргументация за избора им

## **2.2.1. WordPress.**

WordPress е безплатна и open source система за управление на съдържанието (CMS), написана на PHP и съвместима с MySQL или MariaDB база данни. Функционалностите включват архитектура от плъгини и система от шаблони, познати в WordPress като теми. Първоначално WordPress е създаден като система за публикуване на блогове, но в последствие се развива в подкрепа на други видове уеб съдържание, включващо по-традиционни списъци и форуми за електронна поща, медийни галерии, членски сайтове системи за управление на обучението(LMS) и онлайн магазини. WordPress се използва от повече от 60 милиона уебсайтове, включително 33,6% от топ 10 милиона уебсайтове към април 2019 г. WordPress е едно от най-популярните решения за система за управление на съдържанието в употреба. WordPress се използва и за други приложения, като например широко разпространени дисплейни системи (PDS).

WordPress е бил пуснат на 27 май 2003 г. от своите основатели, американския разработчик Мат Мълленвег и английския разработчик Майк Литъл, като тяхна версия на b2 / cafelog. Софтуерът се пуска под лиценз GPLv2 (или по-нова версия).

За да функционира, WordPress трябва да бъде инсталиран на уеб сървър, или част от интернет хостинг услуга като WordPress.com, или компютър, работещ със софтуерен пакет WordPress.org, за да служи сам като хост на мрежата. Локален компютър може да се използва за тестване и обучение на един потребител.

„WordPress е фабрика за правене на уеб страници“ е честа аналогия, показваща функциите на WordPress: запазва съдържание и позволява на потребителя да създава и публикува уеб страници, изискващо нищо повече от домейн и хост услуга.

WordPress има уеб шаблонна (template) система, използваща шаблонен процесор. Архитектурата на WordPress е водещ контролер, насочващ всички заявки за нестатични URI към един PHP файл, който преобразява URI и идентифицира target страницата. Това позволява поддръжката на повече разчетими от човека пермалинкове.

**Темите в WordPress**

Потребителите на WordPress могат да инсталират и превключват между различни теми. Темите позволяват на потребителите да променят външния вид и функционалността на уебсайта си, без да променят основния код или съдържанието му. Всеки уебсайт на WordPress изисква да има поне една тема и всяка тема трябва да бъде проектирана спрямо WordPress стандартите със структуриран PHP, валиден HTML (HyperText Markup Language) и CSS(Cascading Style Sheets). Темите могат да бъдат директно инсталирани с помощта на инструмента за администриране на WordPress "Appearance" в таблото за управление или папки с темите могат да бъдат копирани директно в директорията с темите, например чрез FTP. PHP, HTML и CSS, намиращи се в темите, могат да бъдат директно променени, за да променят поведението на темата, или темата може да бъде child тема, която наследява настройките от друга тема и избирателно отменя функциите. Темите на WordPress обикновено се класифицират в две категории: безплатна и премиум. Много безплатни теми са изброени в директорията с теми на WordPress (известна също като хранилище), а премиум темите са достъпни за закупуване от пазарни и отделни разработчици на WordPress. Потребителите на WordPress могат също да създават и разработват свои собствени персонализирани теми. Безплатната тема Underscores, създадена от разработчиците на WordPress, се е превърнала в популярна основа за нови теми.

**Плъгини**

WordPress плъгин архитектурата позволява на потребителите да разширяват функциите и функционалността на уебсайт или блог. Към януари 2020 г. WordPress.org разполага с 55 487 приставки, от които предлага персонализирани функции и такива, позволяващи на потребителите да приспособяват сайтовете си към техните специфични нужди. Това обаче не включва премиум плъгините, които са на разположение (приблизително 1 500+), които може да не са изброени в хранилището на WordPress.org. Тези персонализации варират от оптимизация за търсачки (SEO), до клиентски портали, използвани за показване на частна информация за влезли потребители, до системи за управление на съдържанието, до функции, показващи съдържание, като добавяне на джаджа (widget) и навигационни ленти. Не всички налични плъгини(приставки) винаги са в крак с ъпгрейдите и в резултат на това може да не функционират правилно или да не функционират изобщо. Повечето плъгини са достъпни чрез самите WordPress, или чрез изтеглянето им или инсталирането на файловете ръчно чрез FTP или чрез таблото за управление на WordPress. Въпреки това много трети страни предлагат плъгини чрез свои собствени уебсайтове, много от които са платени пакети. Разработчиците на уеб приложения, които искат да разработват приставки (плъгини) трябва да научат hook системата на WordPress, състояща се от над 300 екшън и филтър куки (hooks).

**Мобилни приложения**

Съществуват мобилни апликации за WordPress за Android, WebOS, iOS (iPhone, iPod Touch, iPad), Windows Phone и BlackBerry. Тези приложения, създадени с Automattic, имат опции като добавяне на нови блог постове и страници, коментиране, отговаряне на коментари, модериране на коментари.

**Достъпност**

Екипът по достъпност на WordPress е работил, за да подобри достъпността до налични теми и пр. Той осигурява продължаваща образователна поддръжка относно уеб достъпност и включващия дизайн.

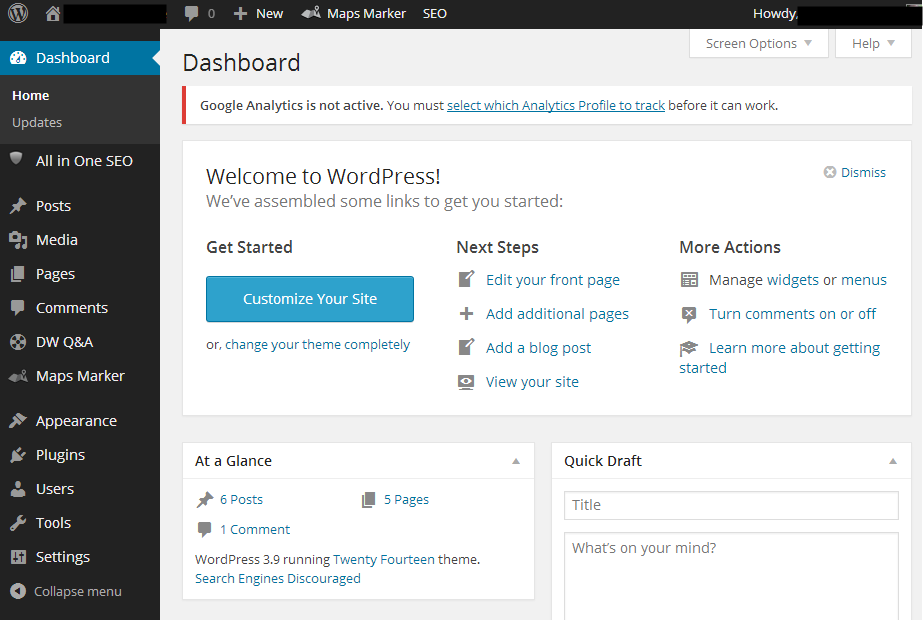
Стандартите за кодиране на достъпността на WordPress твърдят, че „Всички нови или актуализирани кодове, пуснати в WordPress, трябва да съответстват на Указания за достъпност на уеб съдържанието 2.0 на ниво АА“.

**Други функционалности**

WordPress разполага и с интегрирано управление на връзки; чиста структура за постоянна връзка, удобна за търсачката; възможността за присвояване на няколко категории на публикации (постове); и поддръжка за маркиране на публикации. Автоматичните филтри също са включени, осигурявайки стандартизирано форматиране и стилизиране на текст в публикации (например преобразуване на обикновени кавички в интелигентни кавички). WordPress също така поддържа стандартите Trackback и Pingback за показване на връзки към други сайтове, които са свързани към публикация или статия. Публикациите в WordPress могат да бъдат редактирани в HTML, с помощта на визуалния редактор или с помощта на една от редица приставки, позволяващи разнообразие от персонализирани функции за редактиране.

**Multi-user и multi-blogging**

Преди версия 3 WordPress е поддържал един блог за инсталация, въпреки че множество паралелни копия могат да се изпълняват от различни директории, ако са конфигурирани да използват отделни таблици на база данни. WordPress Multisites (по-рано наричан WordPress Multi-User, WordPress MU или WPMU) е била версия на WordPress, създадена да позволи на много блогове да съществуват в рамките на една инсталация, но е в състояние да бъде администрирана от централизиран сервиз. WordPress MU дава възможност на тези, които имат уебсайтове, да хостват свои собствени блогови общности, както и да контролират и модерират всички блогове от едно табло. WordPress MS добавя осем нови таблици с данни за всеки блог. [10]



Фигура 2.1. Таблото на WordPress, чрез което потребителите управляват и редактират сайтовете си

**Причини за избор на WordPress**

* **Лесно и удобно за ползване**
* **Спестява много време**
* **Не е нужно да се работи по дизайна на сайт – съществуват стотици вече готови и безплатни теми**
* **Технологията съдържа много полезни приставки (плъгини), които също спестяват много време, като добавят дадена функционалност към сайта ни, без да ни се налага да разработваме сами код**
* **Над 56% от всички уеб страници в Интернет пространството са разработени и се поддържат от WordPress (най-голям дял на пазара)**

## **2.2.2. XAMPP.**

XAMPP е безплатен и open-source кросплатформен стек от пакети за уеб сървър, разработен от Apache Friends, състоящ се главно от HTTP сървъра Apache HTTP, базата данни MariaDB и интерпретатори за скриптове, написани на езиците за програмиране на PHP и Perl. Тъй като повечето действителни съществуващи уеб сървъри използват същите компоненти като XAMPP, това прави възможно преминаването от локален тестов сървър към реален сървър.

**Необходимо за инсталация**

XAMPP изисква само един zip, tar, 7z или exe файл, за да бъде изтеглен и стартиран, и се изисква малка или въобще никаква конфигурация на различните компоненти, съставляващи уеб сървъра. Версията на XAMPP за Windows изисква Microsoft Visual C ++ 2017 Redistributable.

**Функционалности**

XAMPP редовно се актуализира до най-новите версии на Apache, MariaDB, PHP и Perl. Той също така идва с редица други модули, включително OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress и други. На един компютър могат да съществуват самостоятелни, множество инстанции на XAMPP и всеки даден екземпляр може да се копира от един компютър на друг. XAMPP се предлага както в пълна, така и в стандартна версия (по-малка версия).

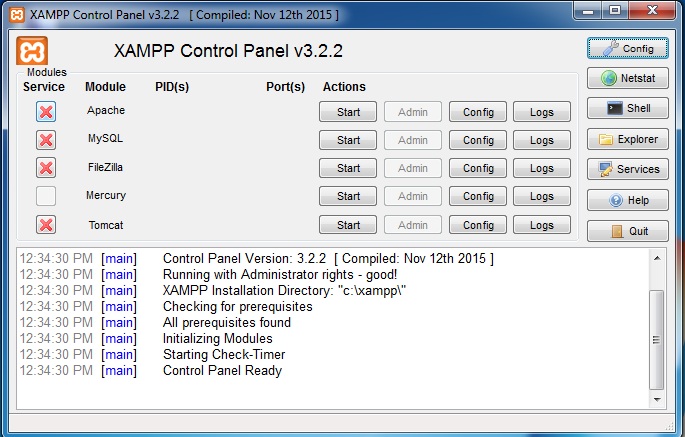
**Употреба**

Най-очевидната характеристика на XAMPP е лекотата, при която стека на уеб сървър на WAMP може да бъде разгърната и инстанцирана. По-късно някои битови приложения, които лесно могат да бъдат инсталирани, са били предоставени от Bitnami.

Официално дизайнерите на XAMPP възнамеряват да го използват само като инструмент за разработка, за да позволят на дизайнерите на уебсайтове и програмистите да тестват работата си на собствените си компютри без никакъв достъп до Интернет. За да се направи това възможно най-лесно, много важни функции за защита са деактивирани по подразбиране. XAMPP има възможност за обслужване на уеб страници в World Wide Web (www). Предвиден е специален инструмент за защита на паролата на най-важните части от пакета.

XAMPP също така осигурява поддръжка за създаване и манипулиране на бази данни в MariaDB и SQLite.

След като XAMPP е инсталиран, е възможно localhost да се третира като отдалечен хост чрез свързване с помощта на FTP клиент. Използването на програма като FileZilla има много предимства при инсталиране на система за управление на съдържанието (CMS) като Joomla или WordPress. Възможно е също така свързване с localhost чрез FTP с HTML редактор. [11]



Фигура 2.2. Табло за упраление на XAMPP

Чрез таблото за управление може да се стартира уеб сървъра (чрез натискане на бутона „Start“ на Apache и MySQL), който се използва, както и да се достъпят базите дании на сайта (чрез избиране на „Admin“ на MySQL) – XAMPP използва модула phpMyAdmin по подразбиране. След като са изпълнени тези стъпки, може да се достъпи WordPress сайта чрез въвеждане на „localhost/<<site\_name>>“. За Windows най-популярните възможности за технология за Web Server са XAMPP и WampServer, а за MacOs – MAMP.

**Причини за избор на XAMPP**

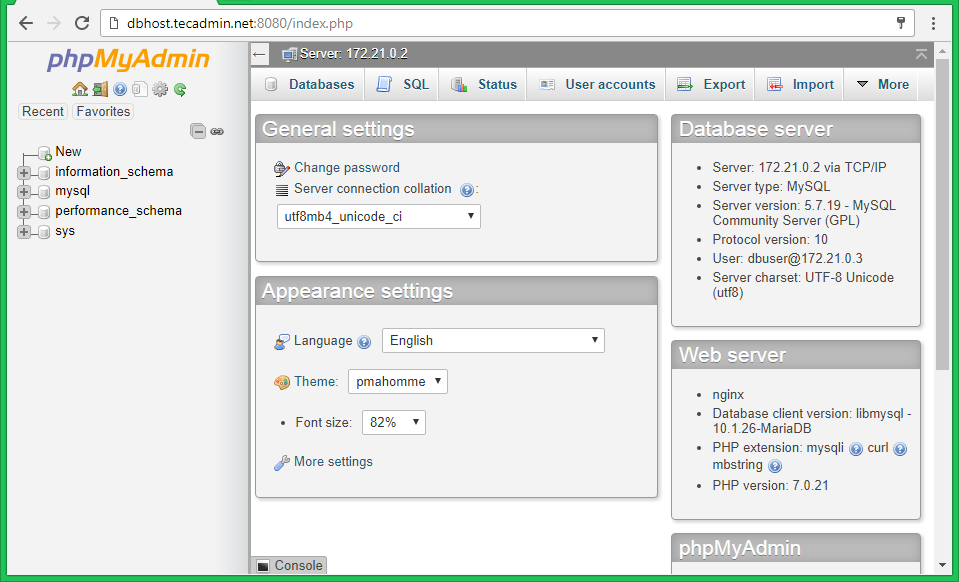
* **Лесен за използване**
* **Осигурява програма по подразбиране за управление на базата данни**
* **Най-популярният избор (подобна на XAMPP технология е WampServer)**

## **2.2.3. phpMyAdmin**

phpMyAdmin е безплатен и open-source за администриране на MySQL и MariaDB. Като преносимо уеб приложение, написано главно на PHP, то се е превърнало в един от най-популярните инструменти за администриране на MySQL, особено за уеб хостинг услуги. Пуснато е в употреба на 9-ти септември 1998г.

**Функционалности**

1. Уеб интерфейс
2. Управление на MySQL и MariaDB база данни
3. Импортиране на данни от CSV и MySQL
4. Експортиране на данни в различни формати: CSV, SQL, XML, PDF (чрез библиотеката TCPDF)
5. Администриране на много сървъри
6. Създаване на PDF графика на оформлението на базата данни
7. Създаване на сложни заявки с помощта на пример със заявка (QBE)
8. Търсене в световен мащаб в база данни или подмножество от нея
9. Преобразуване на съхранени данни във всеки формат с помощта на набор от предварително зададени функции, като показване на BLOB данни като изображение или връзка за изтегляне
10. Графики на живо за наблюдение на MySQL сървърната дейност като връзки, процеси, използване на процесор / памет и т.н.
11. Работа с различни операционни системи
12. Опростява сложни SQL заявки [12]



Фигура 2.3. Табло за управление на phpMyAdmin

**Причини за избор на phpMyAdmin**

* Лесно за използване – създаване на нови база данни, редактиране на вече съществуващи такива, импортване и експортване
* Модула за база данни на XAMPP по подразбиране – не се налага инсталирането на друга апликация за управление на база данни
* Една от най-популярните възможности

## **2.2.4. php**

PHP е сървърен скриптов език и мощен инструмент за създаване на динамични и интерактивни уеб страници.

PHP е широко използвана, безплатна и ефективна алтернатива на конкурентите като ASP на Microsoft.

PHP 7 е най-новата стабилна версия. [13]

PHP кодът може да бъде изпълнен с интерфейс на командния ред (CLI), вграден в HTML код или използван в комбинация с различни системи за уеб шаблони, системи за управление на уеб съдържание и уеб рамки (frameworks). PHP код обикновено се обработва от PHP интерпретатор, реализиран като модул в уеб сървър или като изпълним CGI (Common Gateway Interface). Уеб сървърът извежда резултатите от интерпретирания и изпълнен PHP код, който може да бъде всякакъв тип данни, като генериран HTML код или данни от двоично изображение. PHP може да се използва за много програмиране на задачи извън уеб контекста, като самостоятелни графични приложения и роботизирано управление на дронове.

Стандартният PHP интерпретатор, захранван от Zend Engine, е безплатен софтуер, пуснат под лиценза на PHP. PHP е широко преносим и може да бъде разгърнат безплатно на повечето уеб сървъри на почти всяка операционна система и платформа.

Езикът на PHP се развива без писмена формална спецификация или стандарт до 2014 г., като оригиналното внедряване действа като фактически стандарт, който другите изпълнения целят да следват. От 2014 г. продължава работата по създаването на официална спецификация на PHP.

От февруари 2020 г. над половината сайтове в мрежата, използващи PHP, все още са преустановени / „EOLed“ версия 5.6 или по-стара; и над 55% от всички уебсайтове в света работят с версии преди 7.2, като не са официално подкрепени от екипа за развитие на PHP, докато поддръжката за сигурност се предоставя от трети страни, като Debian (до юни 2020 г. за PHP 5). [14]

**Причини за избор на php**

* Open-source
* Достъпно за различни системи за управление на съдържанието
* Ефективни frameworks
* Висока производителност [15]
* Бързо зарежда уеб страници
* Използва се за разработка на WordPress приставки (плъгини)

# 2.3. База данни на WordPress по подразбиране

**Таблиците**

**wp\_posts**

Таблицата с публикации е може би най-важната таблица в базата данни на WordPress. Името понякога заблуждава хора, които вярват, че съдържа чисто техните публикации в блога. Въпреки това, макар и с лошо име, това е изключително мощна таблица, която съхранява различни видове съдържание, включително публикации, страници, елементи от менюто, прикачени файлове в медиите и всякакви персонализирани типове публикации, които сайт използва.

Гъвкавият характер на съдържанието на таблицата се предоставя от колоната „post\_type“, която обозначава дали редът е публикация, страница, прикачен файл, nav\_menu\_item или друг тип. Но тази гъвкавост също затруднява обсъждането и описанието. По същество таблицата съдържа редове от съдържателни обекти с различни типове.

* ID – уникален номер, присвоен на всяка публикация.
* post\_author – уникалния номер (ID) на потребителя, който я е създал.
* post\_date – време и дата на създаване.
* post\_date\_gmt – GMT време и дата на създаване. GMT времето и датата се запазват, за да няма зависимост от часовата зона на уебсайт в бъдеще.
* post\_content – съдържа цялото съдържание за пост, включително HTML, shortcodes and др.
* post\_title – заглавието на поста.
* post\_excerpt – персонализирано интро или кратка версия на съдържанието.
* post\_status – състояние / статус на поста - ‘draft’, ‘pending’, ‘private’, ‘publish’. Също така страхотен WordPress сайт за новини.
* comment\_status – дали са позволени коментари.
* ping\_status – дали постът разрешава пинг и проследявания.
* post\_password – незадължителна парола, използвана за преглед на поста.
* post\_name – URL на заглавието на поста.
* to\_ping – лист от URLs, на които WordPress трябва да прати pingbacks при актуализация.
* pinged – лист от URLs, на които WordPress е пратил pingbacks.
* post\_modified – време и дата на последната модификация.
* post\_modified\_gmt – GMT време и дата на последната модификация.
* post\_content\_filtered – използва се от приставки (плъгини) за кеширане на версия на post\_content, обикновено преминаваща през филтъра „the\_content“. Не се използва от самото ядро ​​на WordPress.
* post\_parent – използва се за създаване на връзка между тази публикация и друга, когато тази публикация е ревизия, прикачен файл или друг тип.
* guid – Global Unique Identifier, постоянният URL за публикацията, а не permalink версията.
* menu\_order – съдържа номера за показване на страници и други видове, които не са публикувани.
* post\_type – идентификаторът на типа съдържание.
* post\_mime\_type – използва се само за прикачени файлове, тип MIME на качения файл.
* comment\_count – общ брой на коментари, pingbacks и проследявания.

wp\_postmeta

Тази таблица съдържа всякаква допълнителна информация за отделните публикации. Това е вертикална таблица, използваща двойки ключ / стойност, за да съхранява своите данни, техника, която WordPress използва в редица таблици в цялата база данни, позволяваща на WordPress ядро, приставки (плъгини) и теми да съхраняват неограничени данни.

* meta\_id – уникален номер, присвоен на всеки ред от таблицата.
* post\_id – уникалният номер (ID) на публикацията, към която се отнасят данните.
* meta\_key – идентификационен ключ за данните.
* meta\_value – данните.

**wp\_comments**

Всяка публикация, която позволява дискусия, може да има коментари, публикувани към нея. Тази таблица съхранява тези коментари и някои конкретни данни за тях. Допълнителна информация може да се съхранява в wp\_commentmeta.

* comment\_ID – уникален номер, присвоен на всеки коментар.
* comment\_post\_ID – уникален номер (ID) на публикацията, към която се отнася дадения коментар.
* comment\_author – име на коментиралия автор.
* comment\_author\_email – имейл на коментиралия автор.
* comment\_author\_url – URL за коментиралия автор.
* comment\_author\_IP – IP Address на коментиралия автор.
* comment\_date – времето и датата на публикуване на коментара.
* comment\_date\_gmt – GMT време и дата на публикуване на коментара.
* comment\_content – текстът, който коментарът съдържа.
* comment\_karma – неизползван от WordPress ядрото, може да бъде използвано от приставки (плъгини) за помощ в управлението на коментарите.
* comment\_approved – ако коментарът е бил одобрен.
* comment\_agent – откъде е публикуван коментара, т.е. уеб браузър, операционна система и др.
* comment\_type – вид на коментара: коментар, pingback или проследяване.
* comment\_parent – отнася се до друг коментар, ако е отговор.
* user\_id – ID на коментиралия автор, ако е регистриран потребител в уебсайта.

**wp\_commentmeta**

Тази таблица съхранява каквато и да е бъдеща информация, свързана с коментара.

* meta\_id – уникален номер, присвоен на всеки ред от таблицата.
* comment\_id – уникалният номер (ID) на поста, към който се отнасят данните.
* meta\_key – идентификационен ключ за данните.
* meta\_value – данните.

**wp\_terms**

Термините са елементи от таксономия (наука за класификация), използвани за класифициране на обекти. WordPress позволява елементи като публикации и персонализирани типове публикации да бъдат класифицирани по различни начини. Например, когато се създава публикация в WordPress, по подразбиране може да се добави категория и някои тагове към нея. И „Категория“, и „Маркер“ са примери за таксономия, основно начин за групиране на нещата заедно.

* term\_id – уникален номер, присвоен на всеки термин.
* name – името на термина.
* slug – URL на името.
* term\_group – възможност за теми или приставки (плъгини) да групират термини заедно, за да използват псевдоними. Не се попълва от самото ядро ​​на WordPress.

**wp\_term\_taxonomy**

Структурата на тази таблица позволява да се използва един и същ термин в различни таксономии. Например „База данни“ може да се използва като категория за публикации и като термин от персонализирана таксономия за персонализиран тип публикация. Терминът на базата данни съществува веднъж в wp\_terms, ноима два реда в wp\_term\_taxonomy за всяка таксономия.

* term\_taxonomy\_id – уникален номер, присвоен на всеки ред от таблицата.
* term\_id – уникалният номер(ID) на съответният термин.
* taxonomy – може да бъде направено във всяка таксономия или вече съществуваща такава, ползвайки register\_taxonomy().
* description – описание на термина в посочената таксономия.
* parent – уникален номер (ID) на родителски (parent) термин.Използва се за йерархични таксономии като „Категории“.
* count – брой на обектите, зададени за дадената таксономия.

**wp\_term\_relationships**

* object\_id – уникалният номер (ID) на обектите на публикацията.
* term\_taxonomy\_id – уникалният номер (ID) на двойката термин / таксономия.
* term\_order – позволява подреждане на термини за даден обект, които не се използват напълно.

**wp\_users**

Тази таблица е основата на управлението на потребителите на WordPress

* ID – уникален номер, присвоен на всеки потребител.
* user\_login – уникално име за потребителя.
* user\_pass – паролите на потребителя.
* user\_nicename – името на потребителя, което се показва.
* user\_email – имейл адрес на потребителя.
* user\_url – URL на потребителя, адреса на уебсайта.
* user\_registered – време и дата на регистриране на потребителя.
* user\_activation\_key – използва се за възстановяване на пароли.
* user\_status – било е използвано в Multisite pre WordPress 3.0, за да идентифицира спам потребител.
* display\_name – желаното име, което да се използва публично в сайта, може да бъде user\_login, user\_nicename, име или фамилия, определени в wp\_usermeta.

**wp\_usermeta**

Тази таблица съхранява каквато и да е бъдеща информация, свързана с потребителите.

* umeta\_id – уникален номер, присвоен на всеки ред от таблицата.
* user\_id – уникален номер (ID) на дадения потребител.
* meta\_key – идентификационен ключ за данните.
* meta\_value – данните.

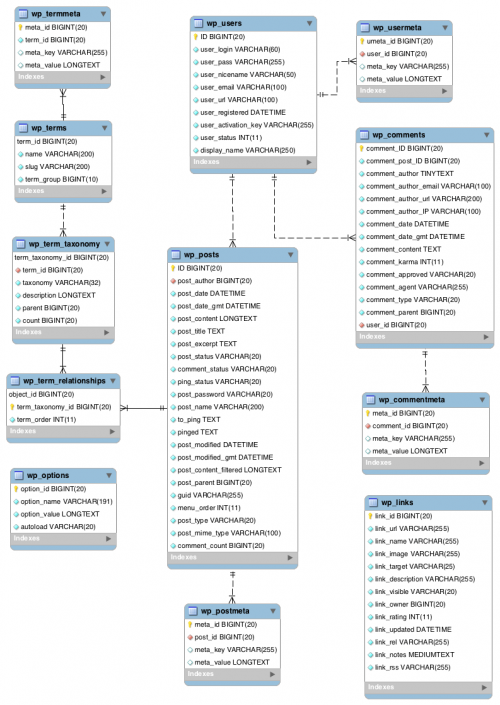
**wp\_options**

Таблицата с опции е мястото, където се съхранява цялата конфигурация на уебсайта, включително данни за темата, активни приставки (плъгини) джаджи (widgets) и временни кеширани данни. Обикновено там, където другите плъгини и теми съхраняват техните настройки.

* option\_id – уникален номер, присвоен на всеки ред от таблицата.
* option\_name – идентификационен ключ за данните.
* option\_value – данните. Те често са сериализирани, така че трябва да се внимава с тях.
* autoload – контролира дали опцията е зарадена автоматично с функцията wp\_load\_alloptions.

**wp\_links**

* link\_id – уникален номер, присвоен на всеки ред от таблицата.
* link\_url – URL на връзка (линк).
* link\_name – име на връзката.
* link\_image – URL на снимка, отнасяща се до връзката.
* link\_target – целевата рамка на връзката, т.е. \_blank, \_top, \_none.
* link\_description – описание на връзката.
* link\_visible – контрол на връзката, ако е публична или частна.
* link\_owner – уникален номер (ID) на потребителя, създал връзката.
* link\_rating – добавя оценка на линка между 1-10.
* link\_updated – време и дата на актуализация на връзката.
* link\_rel – връзка на линка.
* link\_notes – бележки за връзката.
* link\_rss – RSS за връзката (линка). [16]



Фигура 2.4. Връзка между таблиците в базата данни на WordPress

# ТРЕТА ГЛАВА

ПРОГРАМНА РЕАЛИЗАЦИЯ НА WORDPRESS FRAMEWORK ЗА ПОТРЕБИТЕЛСКИ ПЛАНОВЕ И РАЗПЛАЩАНИЯ В MULTISITE СРЕДА

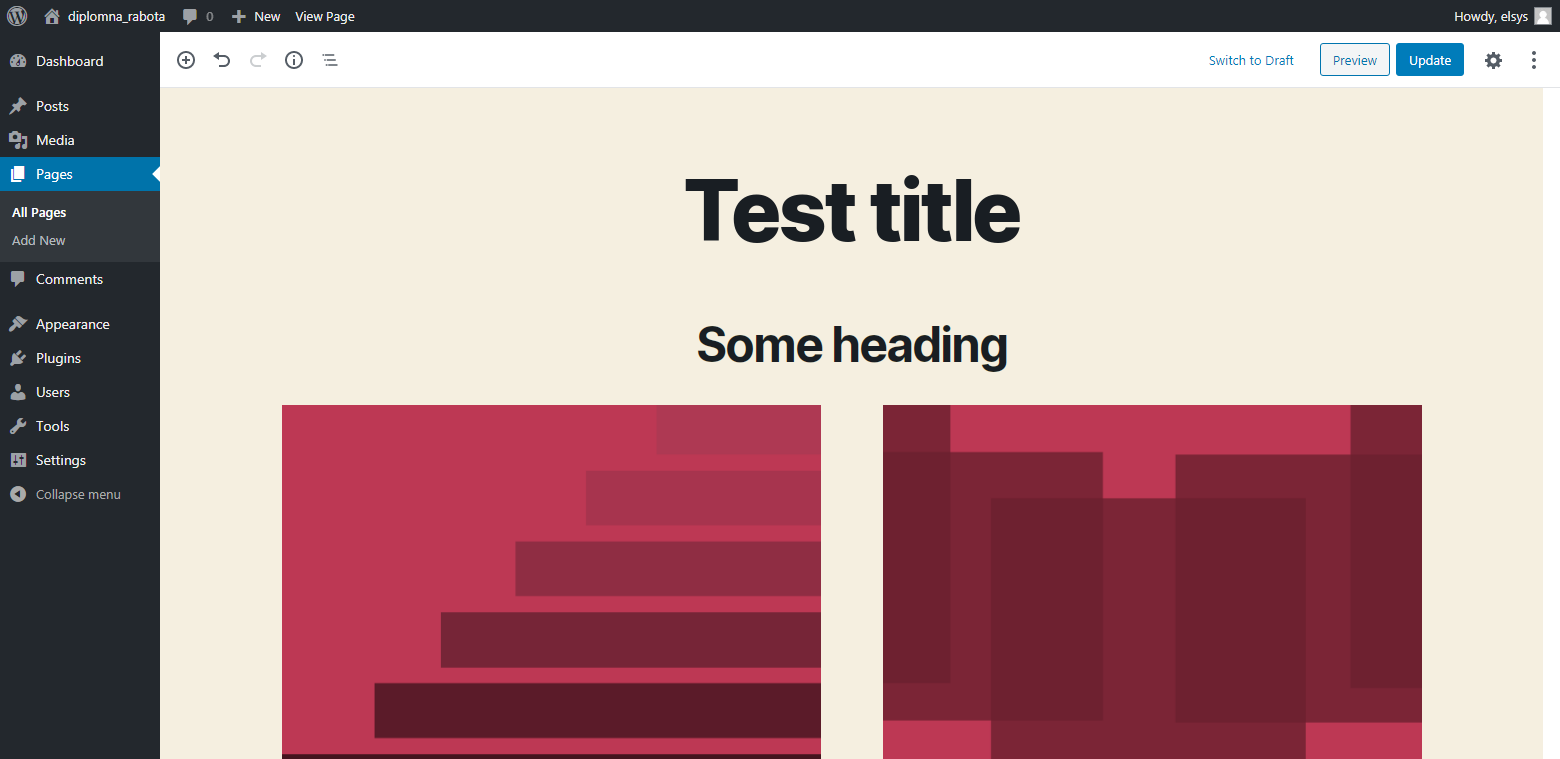
Първата задача за изпълнението на тази дипломна работа е инсталирането на необходимите технологии – WordPress, XAMPP. PhpMyAdmin не изисква допълнителна инсталация, тъй като то е модулът за база данни на XAMPP по подразбиране.

Следващата стъпка е регистрация в WordPress и създаване на нов уебсайт. Инсталацията на WordPress е лесна и отнема само няколко минути. Попълват се полета с името и паролата на базата данни, с която се да свърже новият сайт (ако има парола на базата данни) и друга информация като име на потребителя, парола на потребителя, име на уебсайта, както и хост на уебсайта (ако има такъв, ако ли не се използва localhost). След като е инсталиран WordPress, се избира тема за уеб страницата, а WordPress създава сайт по подразбиране, в зависимост от избраната тема.



Фигура 3.1. Инсталация на WordPress

За тази дипломна работа е избрана темата „Twenty Twenty“ на WordPress.

Фигура 3.2. Първоначален вид на сайта в WordPress

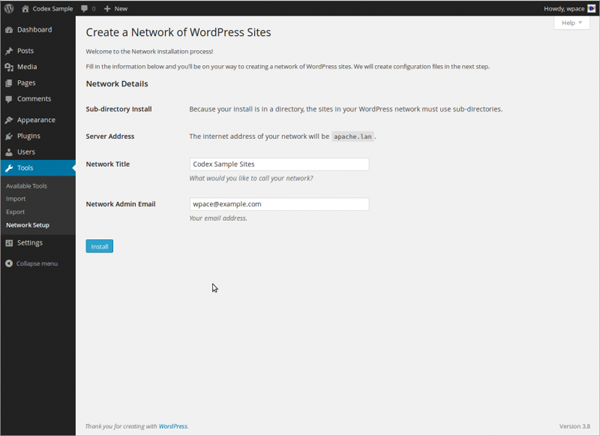
След като вече има примерен сайт за основа, е време да се създаде multisite среда – да се превърне уебсайта в multisite. За тази цел първо трябва да се постави следният код в самото начало на wp-config.php файла, намиращ се в xampp / htdocs / diplomna\_rabota :

/\* Multisite \*/

define( 'WP\_ALLOW\_MULTISITE', true );

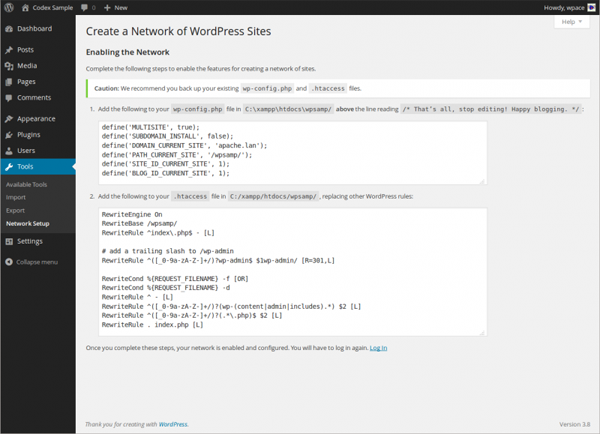
След това трябва да се влезе в админ таблото на уебсайта. Ако има инсталирани плъгини, те трябва да се деактивират преди създаването на multisite средата. След това могат да бъдат активирани отново.

От страничното меню на админ таблото се избира полето „Tools“, а в него – „Network Setup“. Там се въвеждат името на мрежата от WordPress сайтове и email адреса на админа. След като това е изпълнено се натиска бутона „Install“.



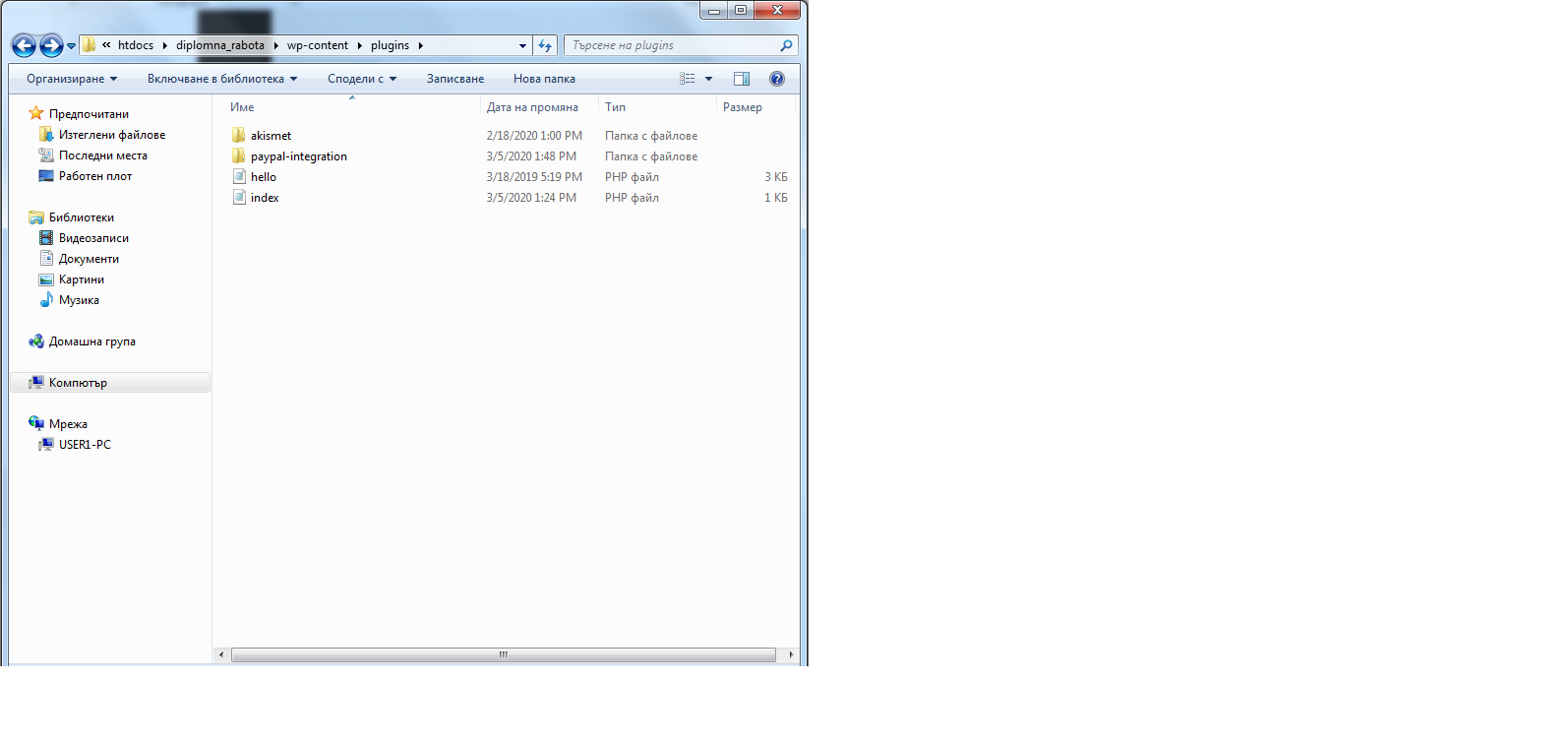
Фигура 3.3. Конфигуриране на multisite среда в админ панела на WordPress

След като тези стъпки са изпълнени, WordPress дава инструкции с код, който трябва да се поставят в началото на файловете wp-config.php и .htaccess, намиращи се в xampp / htdocs / diplomna\_rabota .



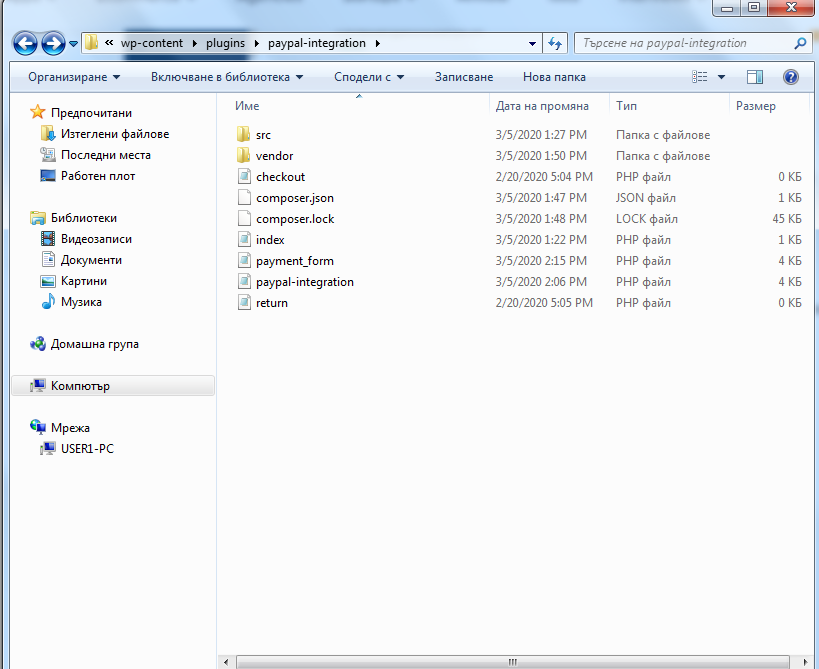
Фигура 3.5. Инструкции за завършване на multisite среда

Следва интеграция на системата за онлайн разплащания „PayPal“ в уебсайта. Това е изпълнено чрез добавяне на нов плъгин към уебсайта. За тази цел е нужно да се създаде нова папка с името на новия плъгин в diplomna\_rabota/ wp-content / plugins. Папката с новия плъгин е кръстена paypal\_integration.



Фигура 3.6. Създаване на нов плъгин

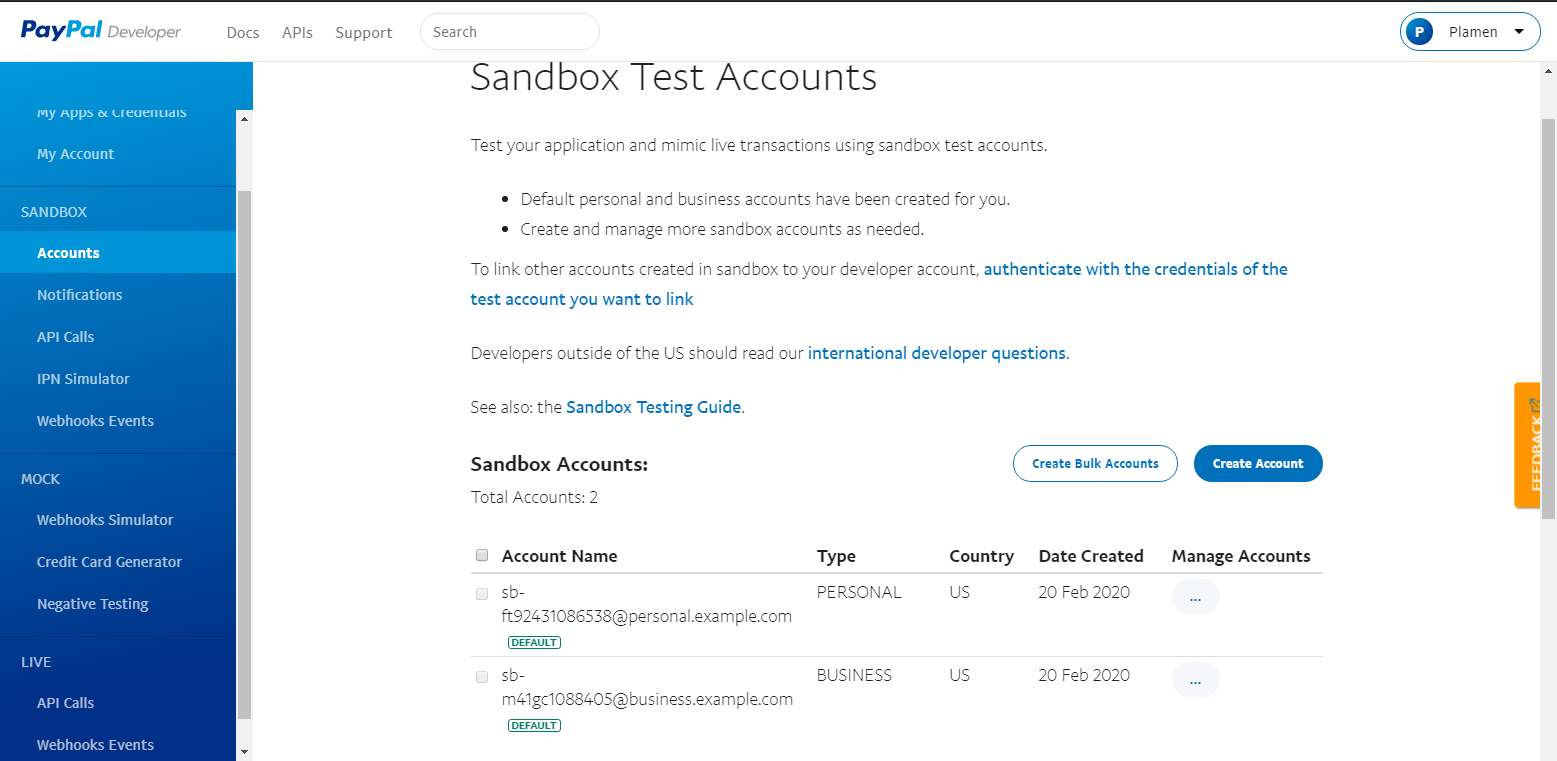
В самата папка на плъгина е нужна структура с файлове и папки, както е посочено по-долу.



Фигура 3.7. Структура на новия плъгин

Преди всичко е нужен PayPal акаунт. Ако няма такъв, той може да бъде създаден много бързо и лесно. След наличието му всеки потребител получава по поне 2 Sandbox акаунта – това е изолирана среда за тестване, която позволява на потребителите да стартират програми или да изпълняват файлове, без да засягат приложението, системата или платформата, на която те се изпълняват. Софтуерните разработчици го използват за тестване на нов програмен код, а специалистите по киберсигурност – за тестване на потенциално зловреден софтуер. [17]

За целта на тази дипломна работа Sandbox акаунтите ще се използват за тестване на плащанията, вместо да бъдат използвани реални парични средства.



Фигура 3.8. Таблото за управление на PayPal Developer акаунт, където могат да бъдат намерени и Sandbox акаунтите на потребителя

За интеграцията на PayPal е необходимо да бъдат инсталирани Composer – това е инструмент за управление на зависимостта в PHP. То позволява да се декларират библиотеките, от които зависи даден проект, като Composer ще ги управлява (инсталира / актуализира) автоматично. [18]

След като е инсталиран composer, е необходимо да бъде изпълнена тази команда в терминала:

composer require omnipay/PayPal

Това ще инсталира пакета и ще може да се конфигурира PayPal по-лесно. Omnipay - това е библиотека за обработка на плащания за PHP. Тя е проектирана на базата на идеи от Active Merchant, плюс опит в прилагането на десетки gateways за CI Merchant. Има ясен и последователен API [19] и дава възможност за конфигурирането на следните опции на PayPal:

* PayPal\_Express (PayPal Express Checkout)
* PayPal\_ExpressInContext (PayPal Express In-Context Checkout)
* PayPal\_Pro (PayPal Website Payments Pro)
* PayPal\_Rest (PayPal Rest API)

След това трябва да се създаде файл composer.json, в който да се сложи следния код:



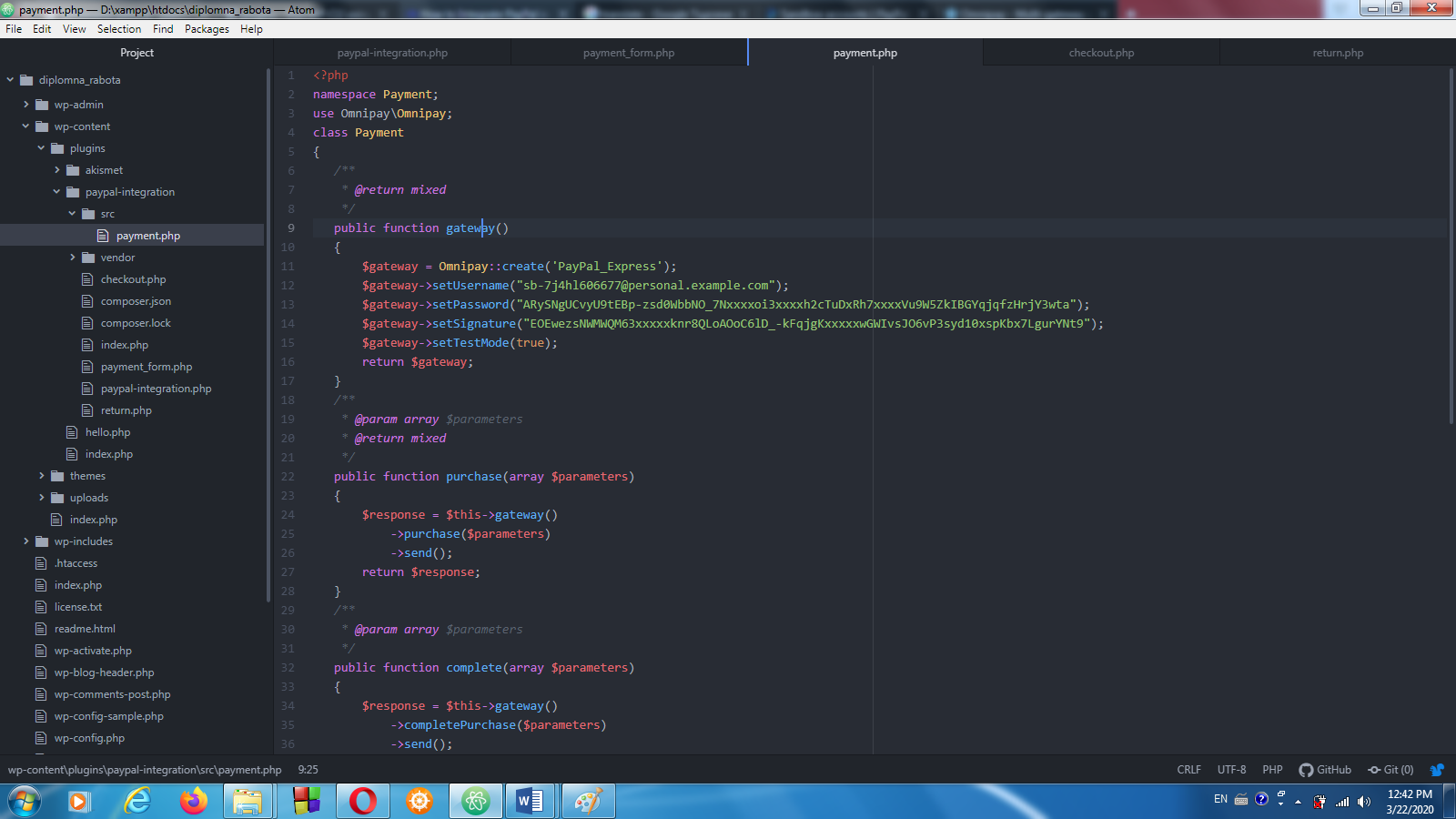
Фигура 3.9. Необходимият код, който трябва да се постави във файла composer.json

и да се изпълни дадената команда в терминала:

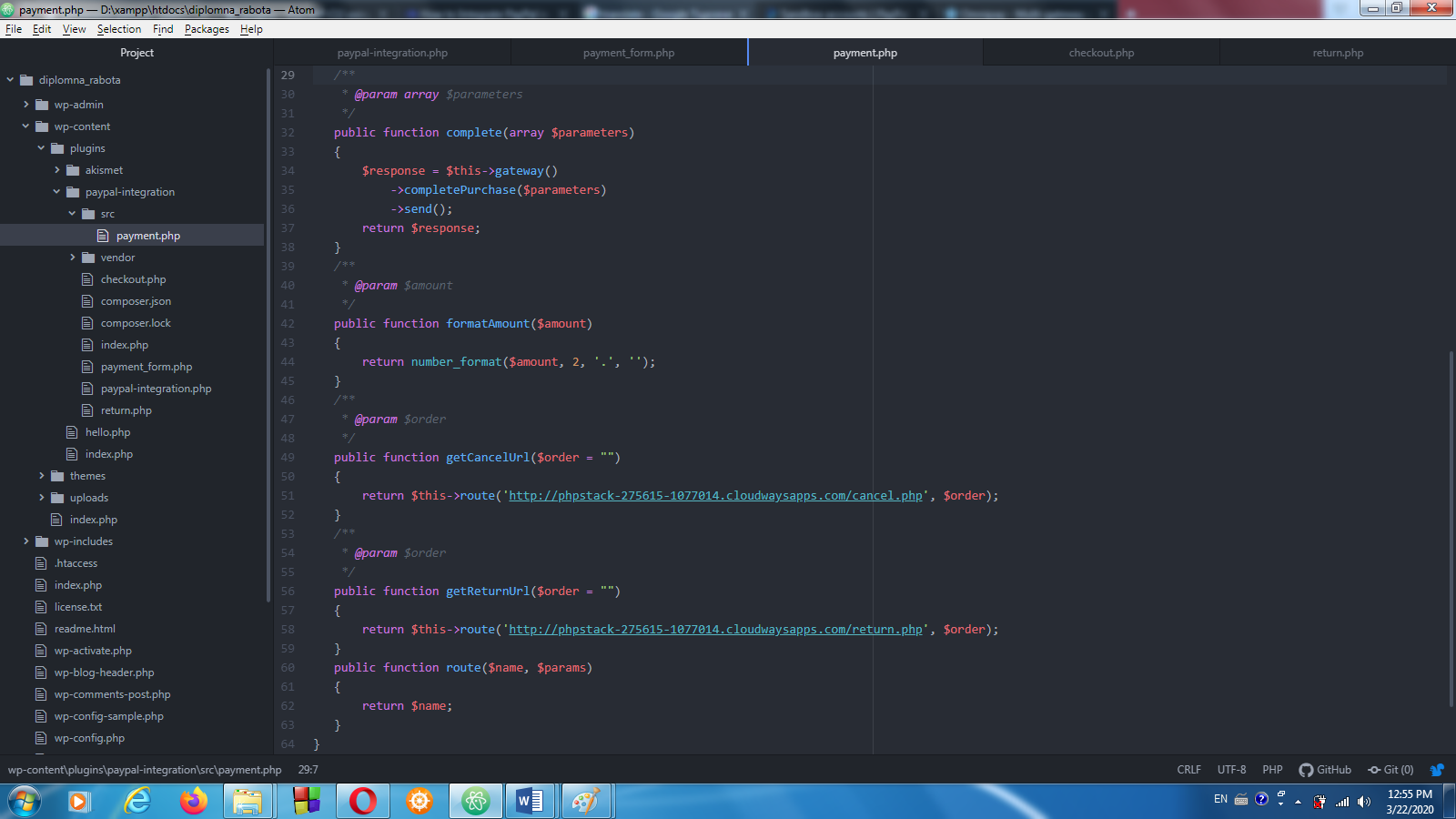
composer install

Посоченият по-горе код инсталира PayPal чрез Omnipay.

В папката src на новия плъгин (diplomna\_rabota / wp-content / plugins / paypal-integration / src) трябва да се създаде нов файл payment.php



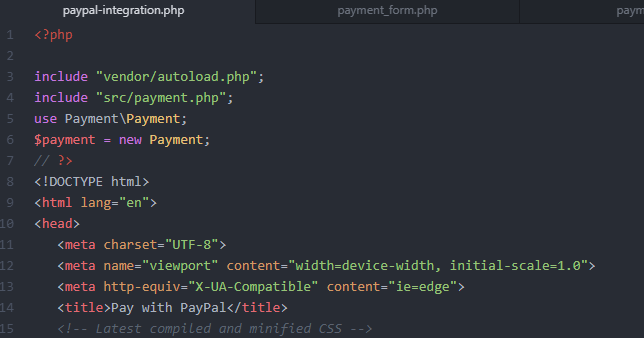
Фигура 3.10. Кодът, който трябва да бъде в payment.php



Фигура 3.11. Кодът в payment.php

Посоченият код позволява извършването на самото плащане чрез PayPal. Вече може да започне използването на предварително изградени методи на PayPal за покупка, формат на сумата, връщащ URL адрес и т.н.

Следва да се създаде форма за уебсайта, която да изпраща плащанията към PayPal. За тази цел се създава файл с името paypal-integration.php, който съдържа следния код:



Фигура 3.12. paypal-integration.php

(От 16-ти до 21-ви ред):

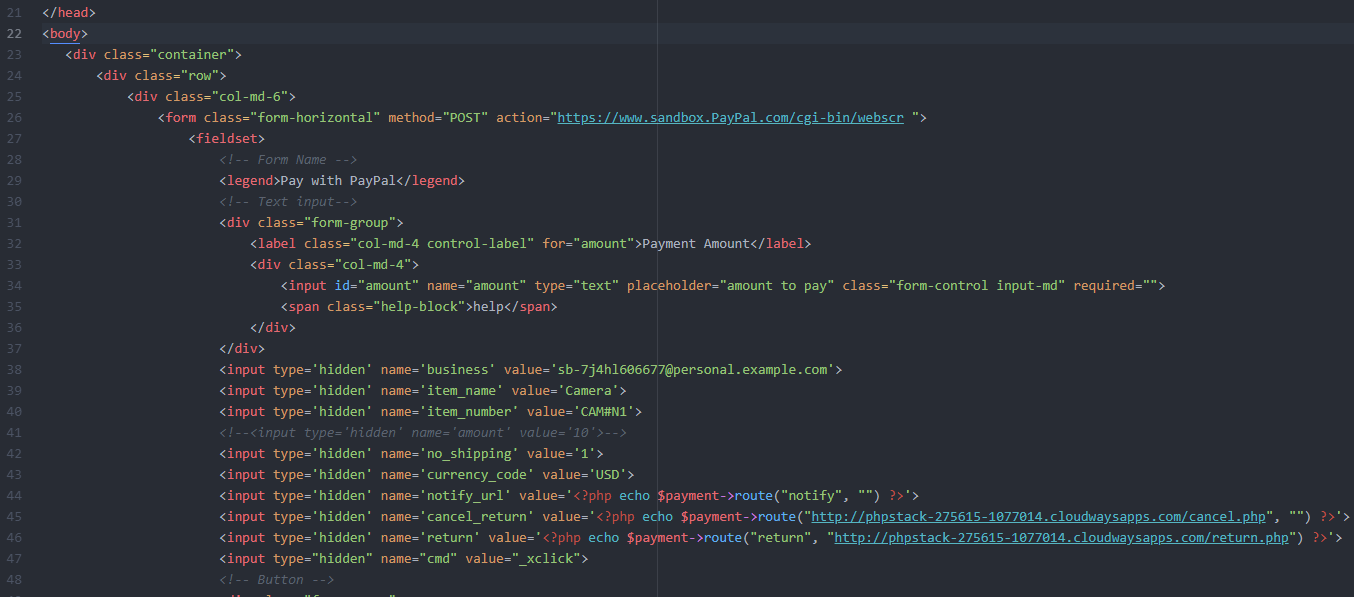
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Optional theme -->

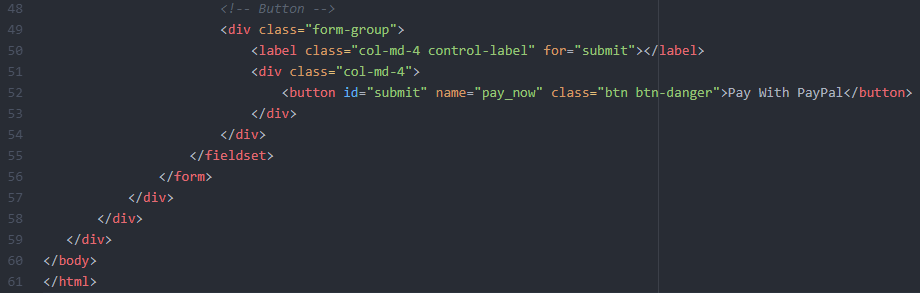
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>



Фигура 3.13. paypal-integration.php

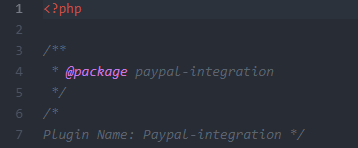


Фигура 3.14. paypal-integration.php

Даденият код е написан на html и php. Той създава форма за извършване на плащанията, като използва файла payment.php (чийто код обработва самите плащания през PayPal), а за дизайна си - готова тема от bootstrap. paypal-integration.php свързва вече имплементирания код за разплащания през PayPal с html форма, която се визуализира на уебсайта.

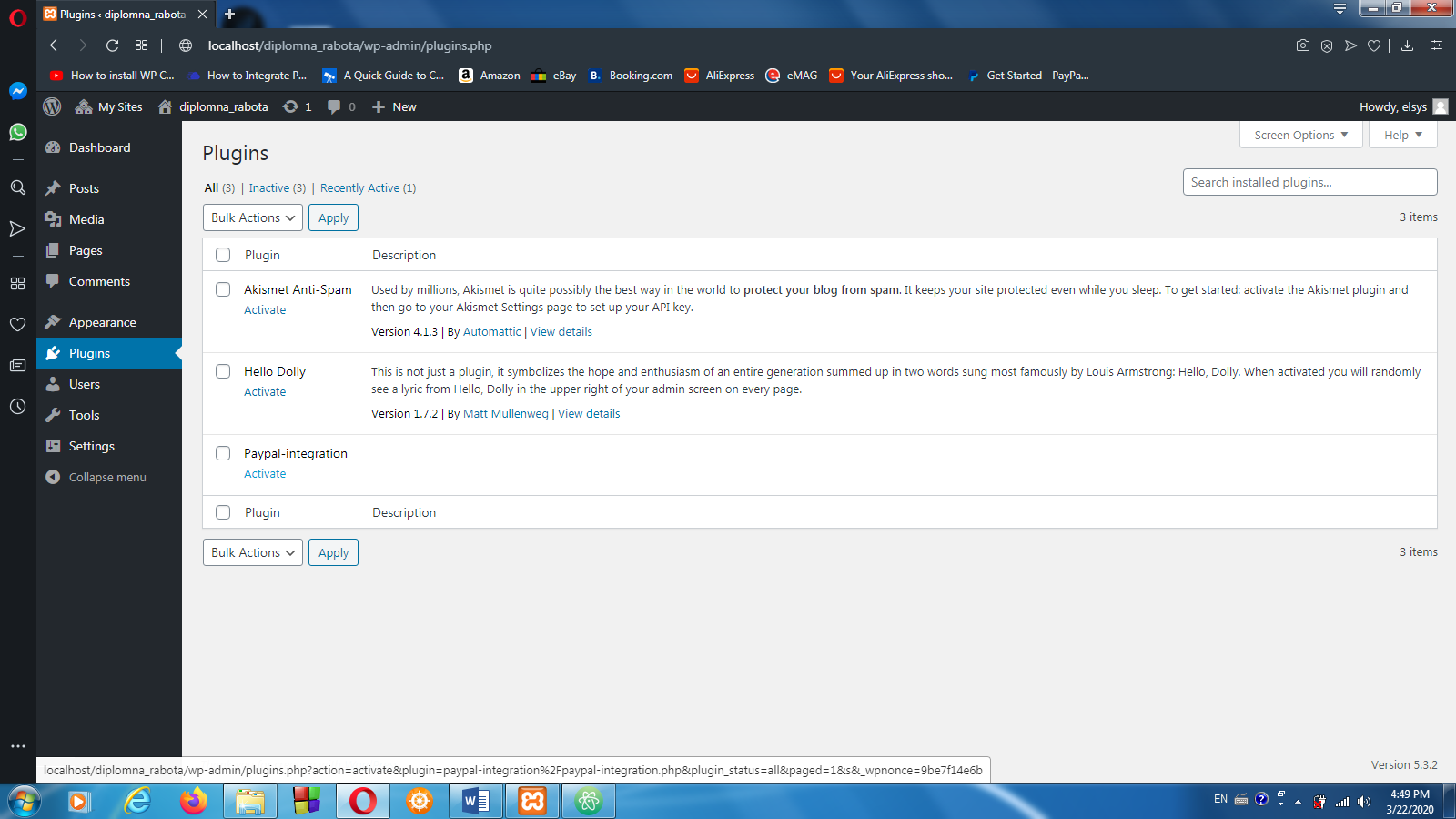
Понеже искаме да имаме два възможни абонаментни плана – Стандартен (Standard) и Скъп(Premium) трябва да имаме две такива форми във файла.

За да бъде качен плъгинът на уеб страницата, е необходимо в paypal-integration.php да се сложи следният код в началото на файла, след „<?php“:

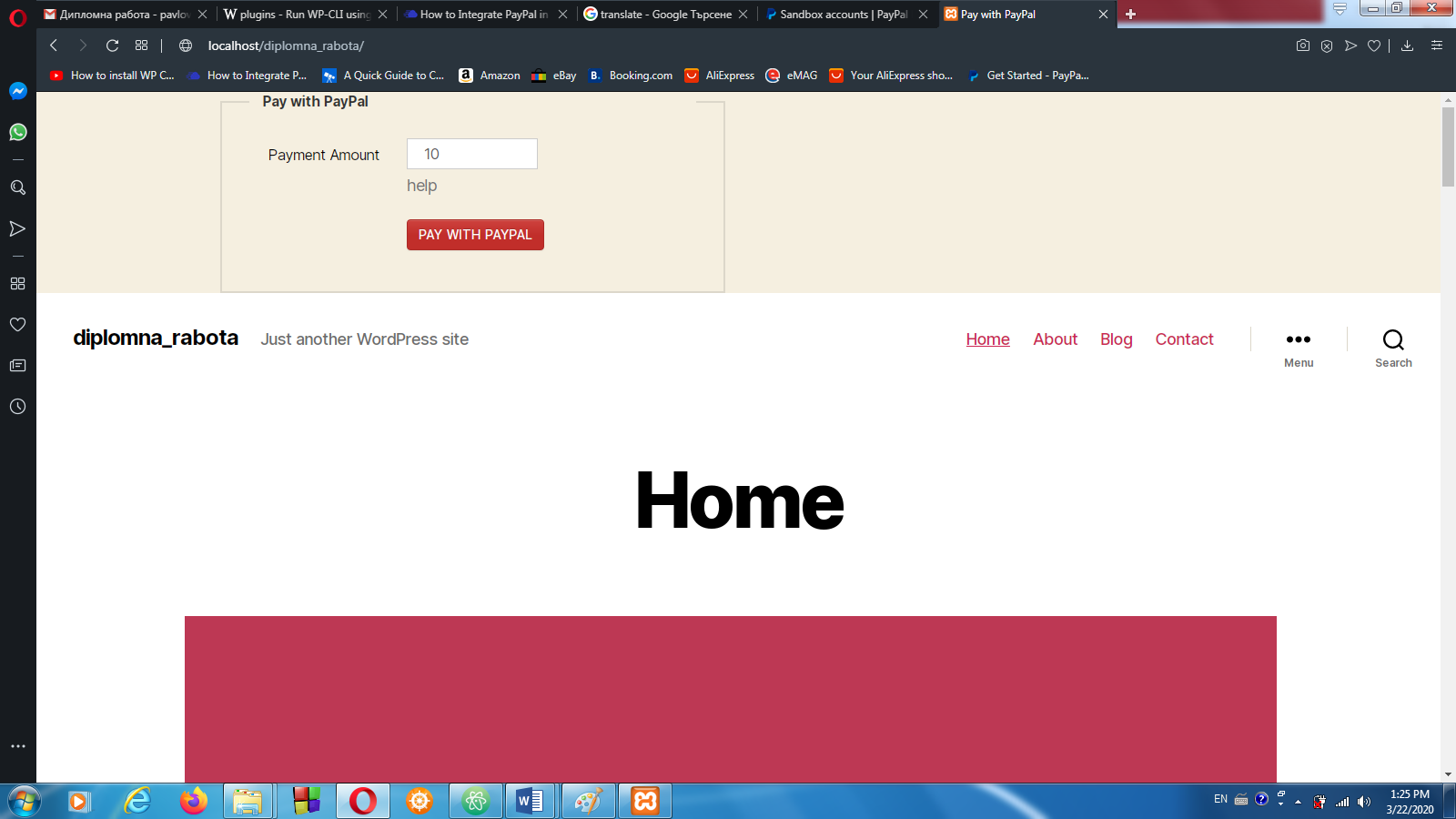


Фигура 3.15. Коментар в началото на paypal-integration.php

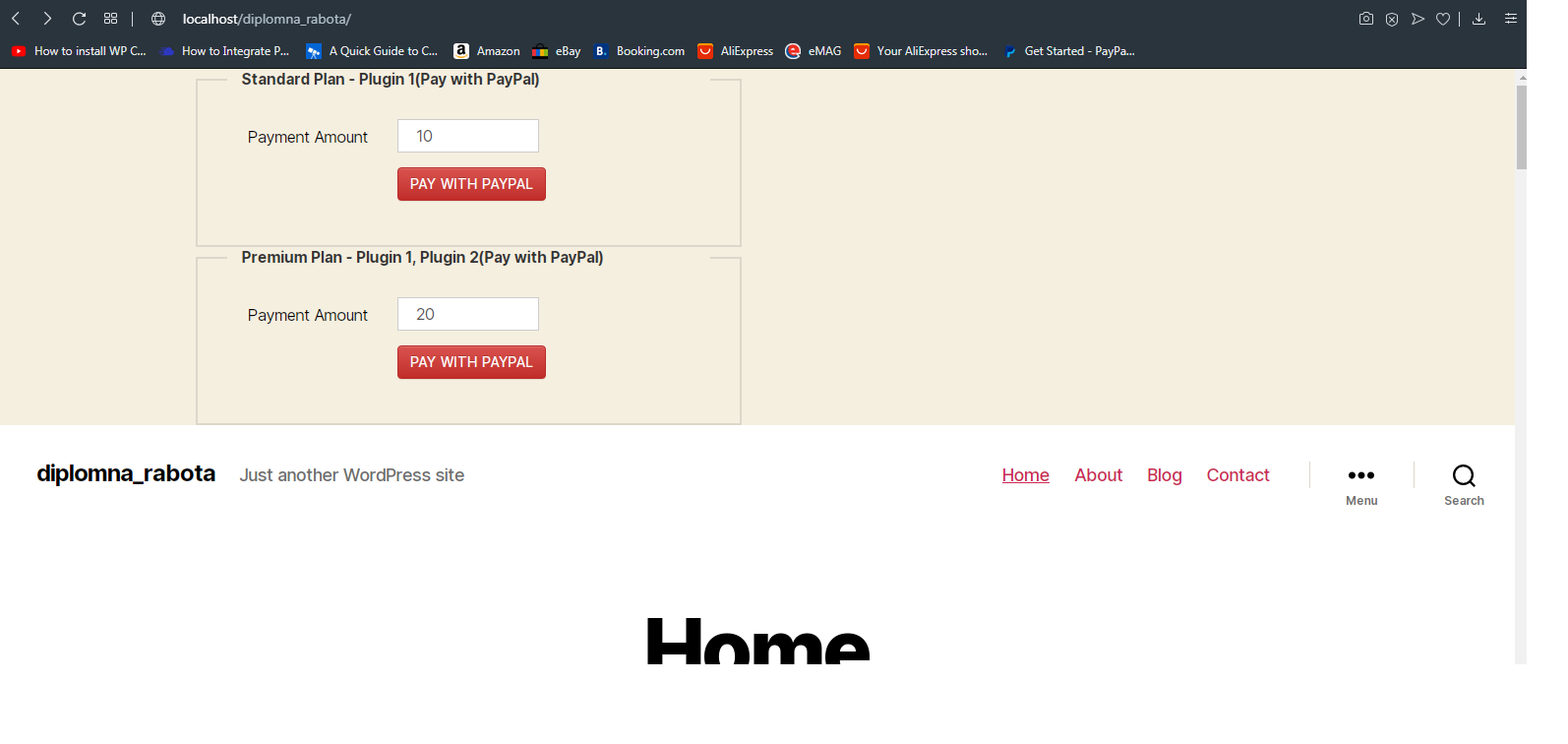
Тези редове „казват” на WordPress, че paypal-integration е приставка. След като уебсайтът се презареди, новият плъгин може да бъде намерен в таблото за управление в секция „Plugins“, а чрез натискане на бутона „Activate“ под името на плъгина, той започва да работи.



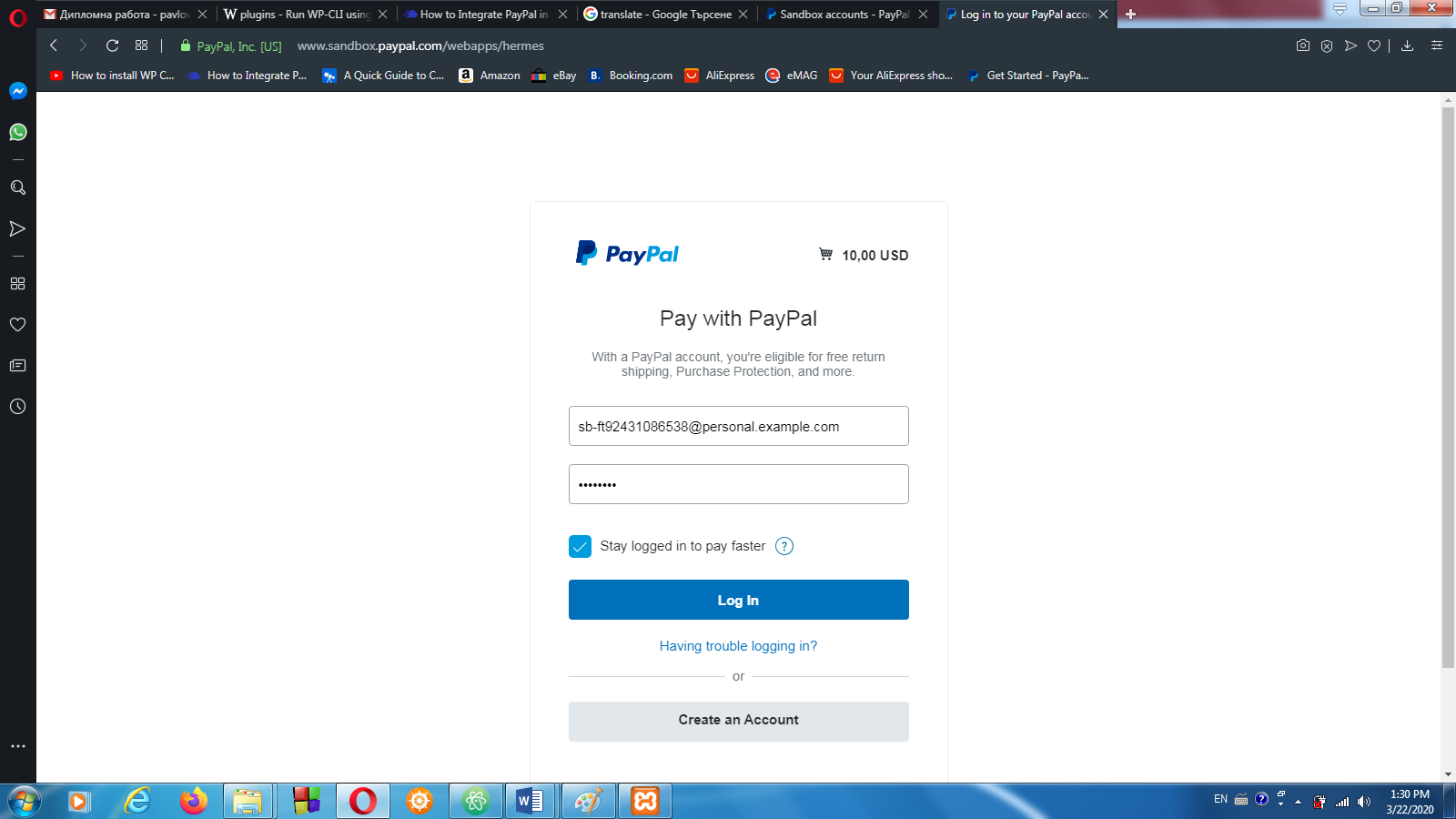
Фигура 3.16. Активиране на новия плъгин



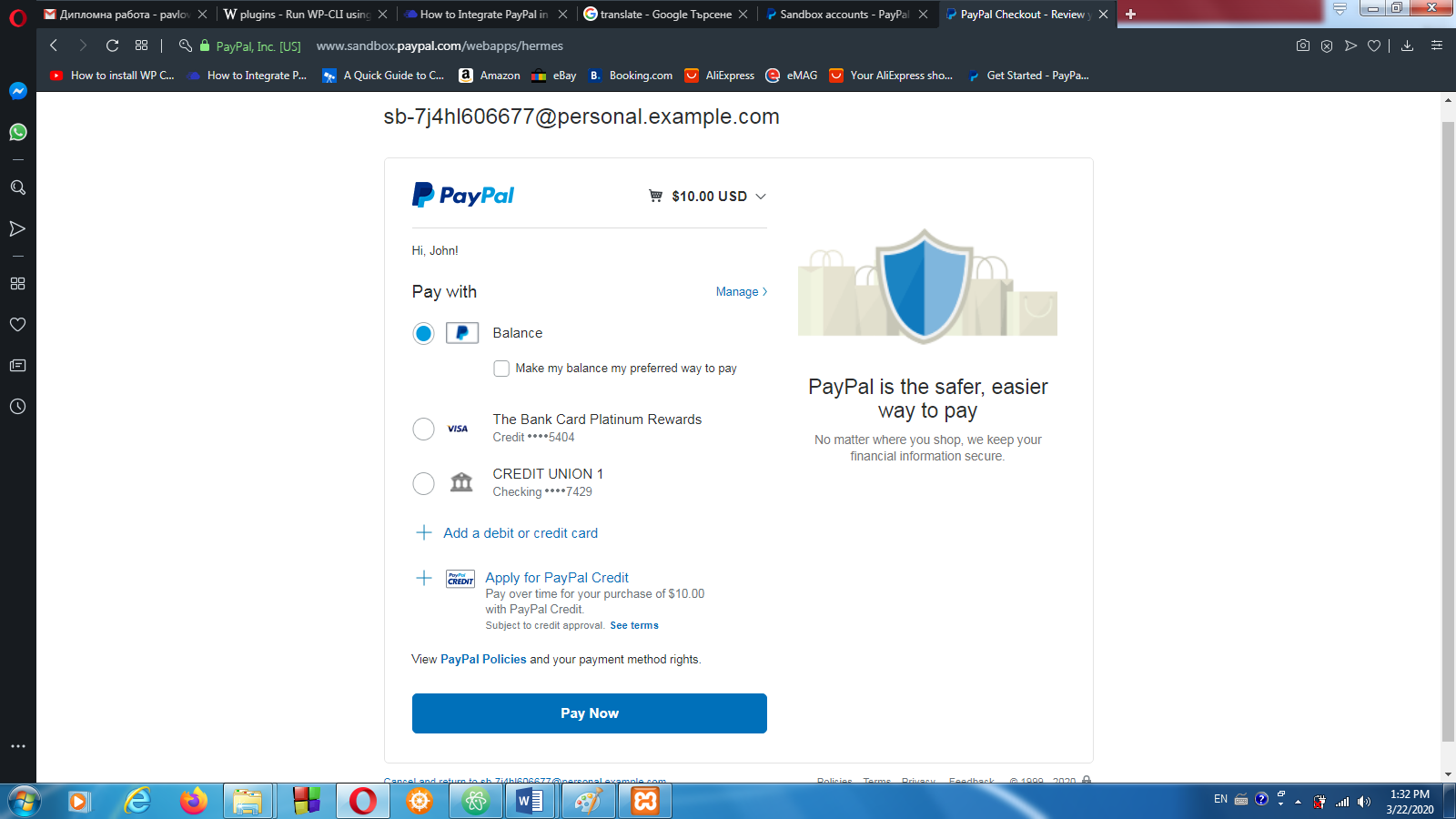
Фигура 3.17. HTML формата, която прави плащането през PayPal възможно



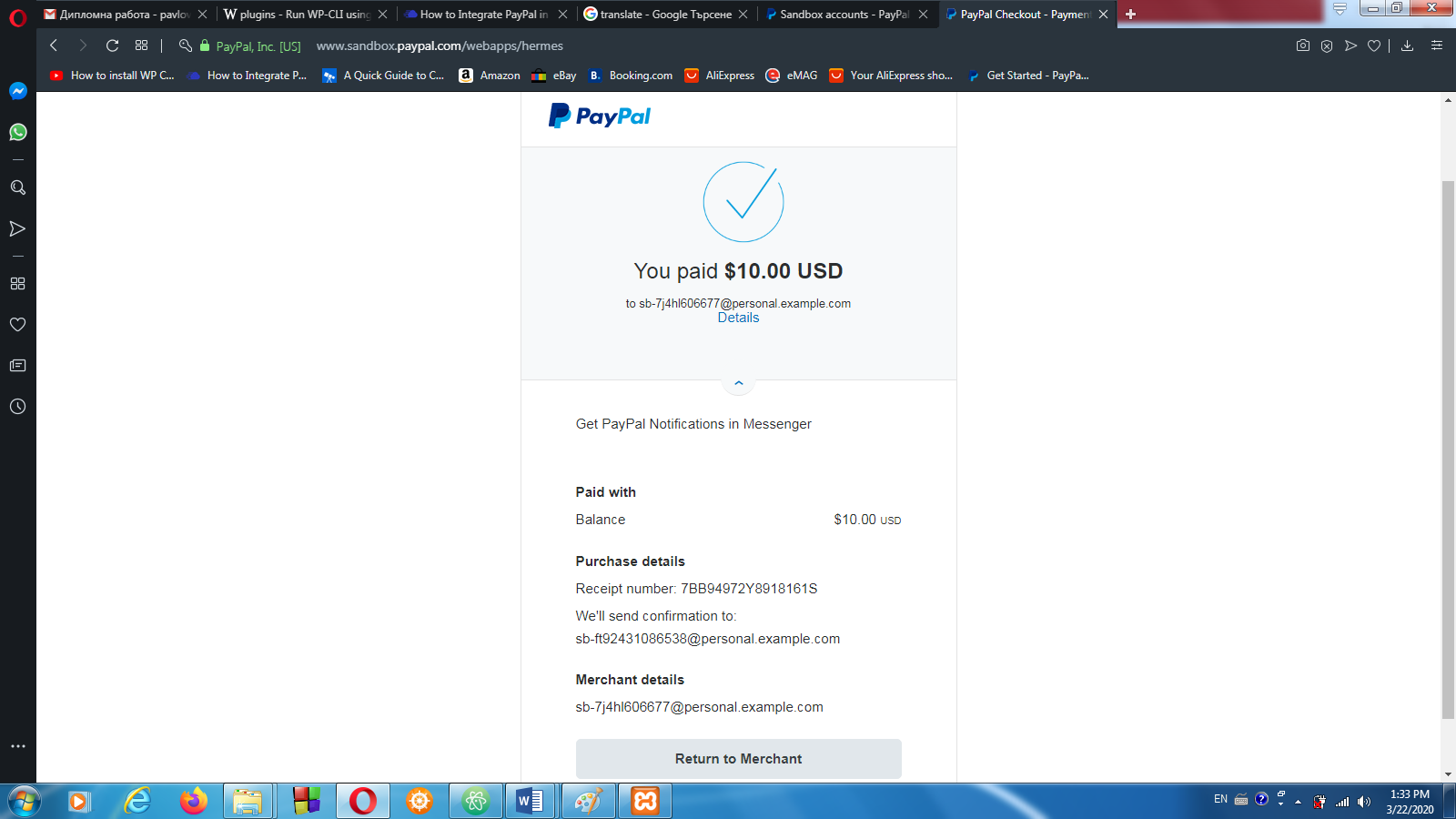
Фигура 3.18. Наличие на вече двата абонаментни плана



Фигура 3.19. След натискане на бутона за плащане е необходимо въвеждането на имейла и паролата на Sandbox акаунта



Фигура 3.20. След въвеждането на информацията се изисква избиране на начин на плащане



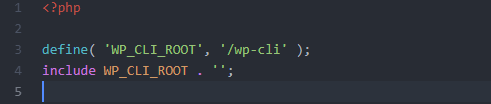
Фигура 3.21. След натискане на бутона „Pay Now“ плащането е завършено, като заплатената сума се отразява на баланса на акаунта

[19]

За създаване на нов subsite в multisite мрежата, конфигуриране на плъгини според заплатения план и създаване на нов потребител, който да има подобни права на админа върху събсайта – такива права, че да не може да прави промени при конфигурираните плъгини, ще се използват команди от WP-CLI. Това е интерфейс на WordPress за терминала. Чрез него могат да се конфигурират плъгини, създава нов subsite в multisite среда и още други неща без използване на уеб браъзур. [20] Командите на WP-CLI се използват през терминала. За да бъдат включени в php код, се използва функцията shell\_exec() – тя изпълнява командата през терминала (shell) и връща крайният резултат като стринг. [21]

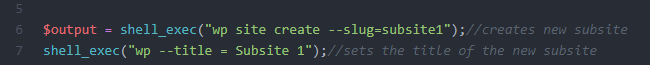
За създаването на нов subsite, на нов потребител (user), който да управлява новия събсайт и конфигурирането на приставките се създава нов файл subsite\_model.php, намиращ се в xampp / htdocs / diplomna\_rabota / wp-content / plugins / paypal-integration / subsite\_model.php.

Той е php скрипт, ползващ функцията shell\_exec(), чрез която успява да изпълнява командите на WP-CLI в терминала. В началото на третият и четвъртият ред включват WP-CLI в кода.



Фигура 3.22. subsite\_model.php

Командата за създаване на нов subsite е „wp site create --slug=<slug>“, където „slug“ е път за новия сайт, поддомейн за инсталации на поддомейни, директория за поддиректории. В случая „slug“ ще е „subsite1“. Цялата команда се запазва, за да може да бъде изведена по-късно на екрана в терминала. След като е оказан пътят на новия subsite, следва да му бъде сложено заглавие. Това се извършва чрез командата „wp --title=<title>“, в която „title“ е заглавието на сайта. За да бъдат изпълнени тези команди през php кода, те трябва да бъдат подадени като аргумент на функцията shell\_exec(). [22]



Фигура 3.23. Задаване на пътя и заглавието на нов subsite

За конфигурирането на плъгини се използва „wp plugin activate <plugin\_name>“, където plugin\_name е името на плъгина. [23]

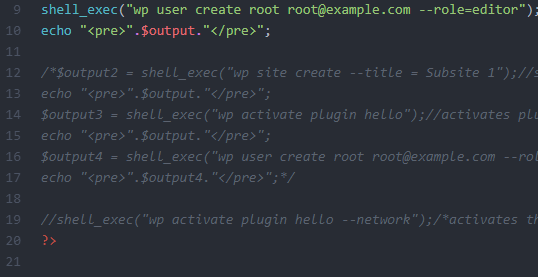


Фигура 3.24. Активиране на плъгин

Командата за създаване на нов потребител е „wp user create <user-login> <user-email> --role=<role>“. <user-login> е потребителското име, <user-email> - имейлът на новия потребител, който създаваме. „<role>“ е ролята, която даваме на новия потребител, като в зависимост каква е тя, са позволени определени права. [24] В WordPress има 5 роли по подразбиране:

1. Administrator – може да инсталира, редактира и изтрива приставки, както и теми. Най-важното е, че той може да добавя нови потребители в сайта, да променя информация за съществуващите потребители, включително техните пароли, както и да изтрива всеки потребител (и други администратори също). Тази роля е основно запазена за собствениците на сайтове и дава пълния контрол върху WordPress сайт.
2. Editor – няма достъп до натройките на сайта, не може да изтрива или да добавя нови плъгини или потребители.
3. Author – може единствено да пише, променя, добавя и изтрива постове.
4. Contributor – може да добавя и да редактира постовете си, но не може да ги публикува или изтрива.
5. Subscriber – може да влезе в профила си в WordPress сайта и да редактира профила си – име, парола и т.н., но няма достъп до нищо от администраторската част на уебсайта. [25]

Тъй като на новия потребител се иска да му се даде най-високата роля в йерархията след тази на администратора, за да няма достъп до настройките на сайта и плъгините, той получава ролята „Editor“.



Фигура 3.25. Създаване на нов потребител на събсайта

Накрая резултата от командите се принтира в терминала.

Изпълнените вече по-горе команди, трябва така да се обединят с интеграцията на PayPal, че при заплащане на определен план да се създава нов събсайт, да се създаде нов потребител за него и да се конфигурират съответните плъгини (приставки), включени в заплатения абонаментен план.

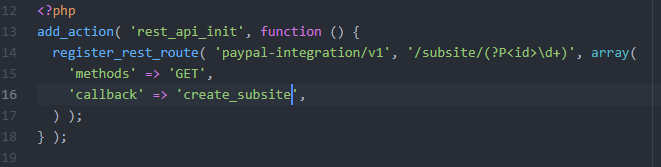
За тази цел трябва да се създаде нов WordPress Custom Endpoint. Той трябва да изисква subsite\_model.php, където се намира и кодът с функциите, които създават нов събсайт в Multisite средата, създават нов потребител и активират нужните плъгини. Точно този endpoint се иползва като callback за PayPal интеграцията, с други думи – при направено успешно плащане, се изпълнява кодът от файла subsite\_model.php.

За изпълняването на това трябва да се регистрира нов route (път). Това „казва“ на API (Application Program Interface/Програмен Интерфейс на Приложение) да отговори на дадена заявка с командите от subsite\_model.php. За тази цел се използва функцията register\_rest\_route(), която трябва да бъде извикана в callback на друга функция – rest\_api\_init(). По този начин се избягва извършването на допълнителна работа, когато API не е заредено.

На register\_rest\_route() е необходимо да се подадат три неща като аргументи:

* Името (Namespace)
* Пътят (Route), който искаме
* Опциите (Options)

На namespace (име) се въвежда името на приставката – в случая ‘paypal-integration’. Прието е след името на плъгина да се пише и версията на API. В този случай namespace ще е – ‘paypal-integraiton/v1’. На route се подава URL адресът - в случая ‘/subsite1’, като може и да се направи ‘/subsite1/(?P<id>\d+)’, като по този пътят ще съвпада с всичко, което съдържа ‘subsite’ ((?P<id>\d+) е ID на събсайта). На options се подава масив, като се въвежда и callback функция (може да се казва create\_subsite() например).



Фигура 3.26. WordPress Custom Endpoint

В този случай функцията, подадена като callback на мястото на Options аргумента трябва да обработва заявката (плащане на абонаментен план) и да изпълнява кода на файла с командите subsite\_model.php – да създаде нов събсайт.

# ЧЕТВЪРТА ГЛАВА

**РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ**

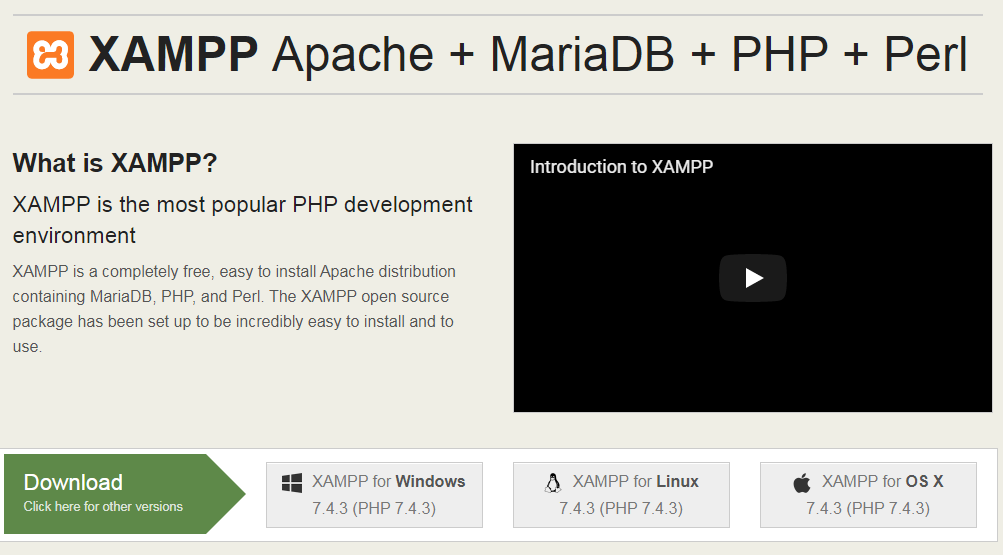
Софтуерните изисквания на проекта са:

* Операционна система – WordPress може да работи и на Windows, и на Linux, и на MacOS
* Инсталиран XAMPP – може да бъде изтеглен безплатно от официалния сайт

За инсталирането на уебсайта няма високи хардуерни изисквания. Минималните необходими са:

* Процесор: 1.0GHz+
* RAM (Оперативна памет): 512MB+
* Дисково пространство: 1GB+

Първоначално трябва да бъде изтеглен XAMPP, който отговаря за сървърната част. Това може да бъде направено от официалния сайт на приложението - <https://www.apachefriends.org/index.html>



Фигура 4.1. официалния сайт на XAMPP

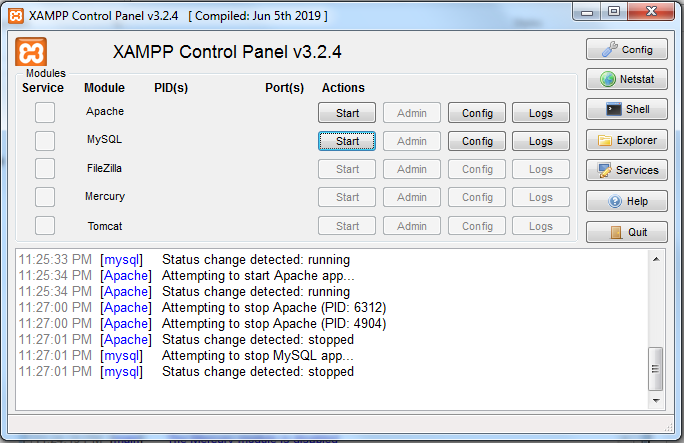
Според операционната система, се избира и съответната версия на апликацията. След като свалянето на файла е приключило, се натиска за инсталиране. Следват няколко обичайни стъпки, като избиране на директория, където да бъде инсталиран XAMPP и други.

Папката с проекта може да бъде изтеглена от репозиторито в GitHub, като това може да бъде направено от следния уеб адрес: <https://github.com/papavlov/Diplomna_Rabota>

Репозиторито е публично и може да бъде видяно и изтеглено от всеки. За да бъде свалено, трябва да се натисне бутона „Download“, а под него „Download ZIP“. След като това приключи, файлът трябва да бъде разархивиран в xampp / htdocs.

Необходимо е да бъде импортната базата данни на проекта. За тази цел трябва да се стартира xampp-control, намиращо се в папка xampp.

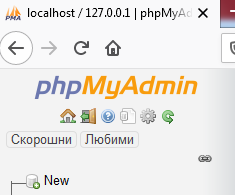
* Натискат се бутоните „Start“ на полетата Apache и MySQL



Фигура 4.2. Табло за управление на XAMPP

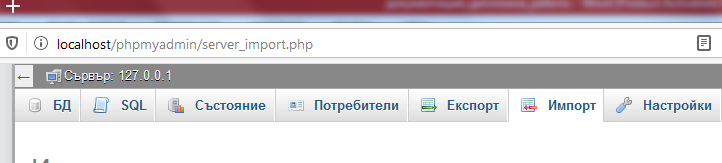
* Избира се „Admin“ на полето MySQL

Това ще отвори нов раздел на браузър, който отвежда в phpMyAdmin. От менюто отляво се натиска „New“, за да се създаде нова база данни.



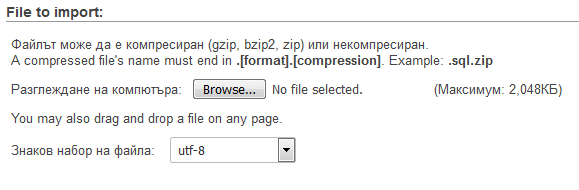
Фигура 4.3. Създаване на нова база данни

* От навигационното меню отгоре се избира „Import“ / „Импорт“



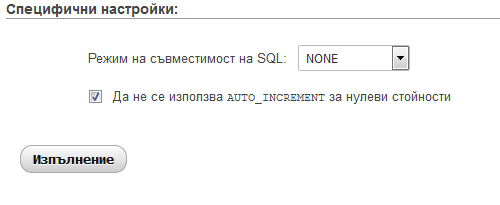
Фигура 4.4. Навигационно меню на phpMyAdmin

* Натиска се бутона „Browse“, след което се избира файлът diplomna\_rabota.sql, намиращ се в xampp / htdocs.



Фигура 4.5. Избиране на файла с датабазата на проекта

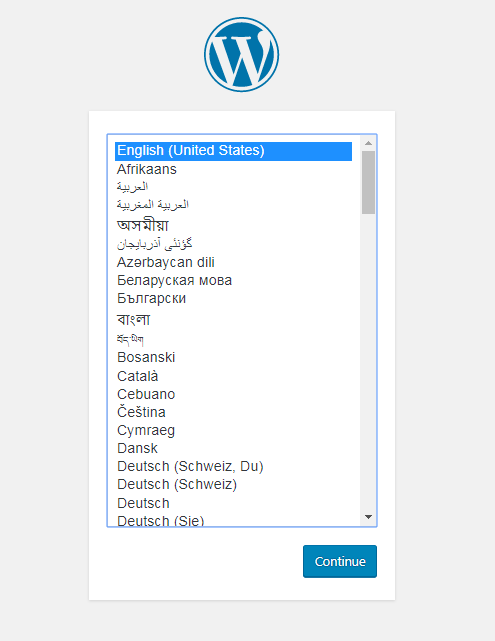
* Процесът се завършва чрез натискане на бутона „Изпълнение“, намиращ се най-отдолу на страницата



Фигура 4.6. Завършване на създаването на датабазата на проекта

След като тези стъпки биват изпълнени, следва да бъде инсталиран WordPress. Това става чрез отваряне на уебсайта в браузъра (за тази цел обаче трябва преди това през XAMPP да бъдат стартирани Apache и MySQL). В полето за адреси на уеб браузъра се въвежда <http://localhost/diplomna_rabota>, след което се визуализира инсталационният екран на WordPress. Процесът на инсталация е лесен и бърз.

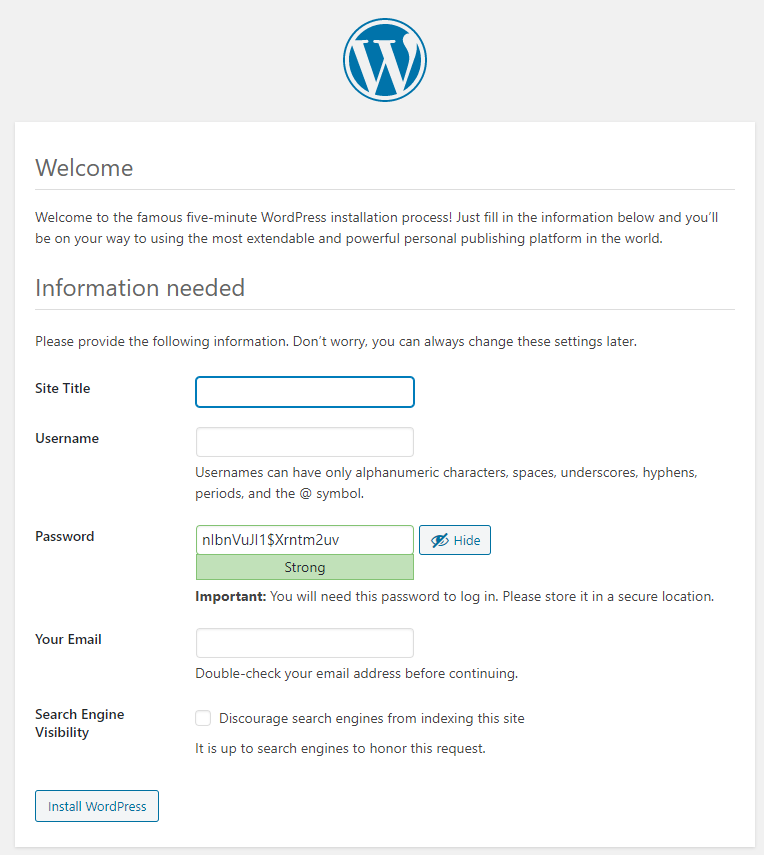
* Избира се желаният език



Фигура 4.7. Избиране на език на WordPress

Въвеждат се:

* Име на сайта – diplomna\_rabota
* Потребителско име – elsys
* Парола – elsys2020
* Email – [pavlovplamen01@gmail.com](mailto:pavlovplamen01@gmail.com)



Фигура 4.8. Въвеждане на информация

* Натиска се бутона „Install WordPress“
* Въвежда се името на базата данни – diplomna\_rabota
* Потребителско име – root
* Парола (това поле се оставя празно)

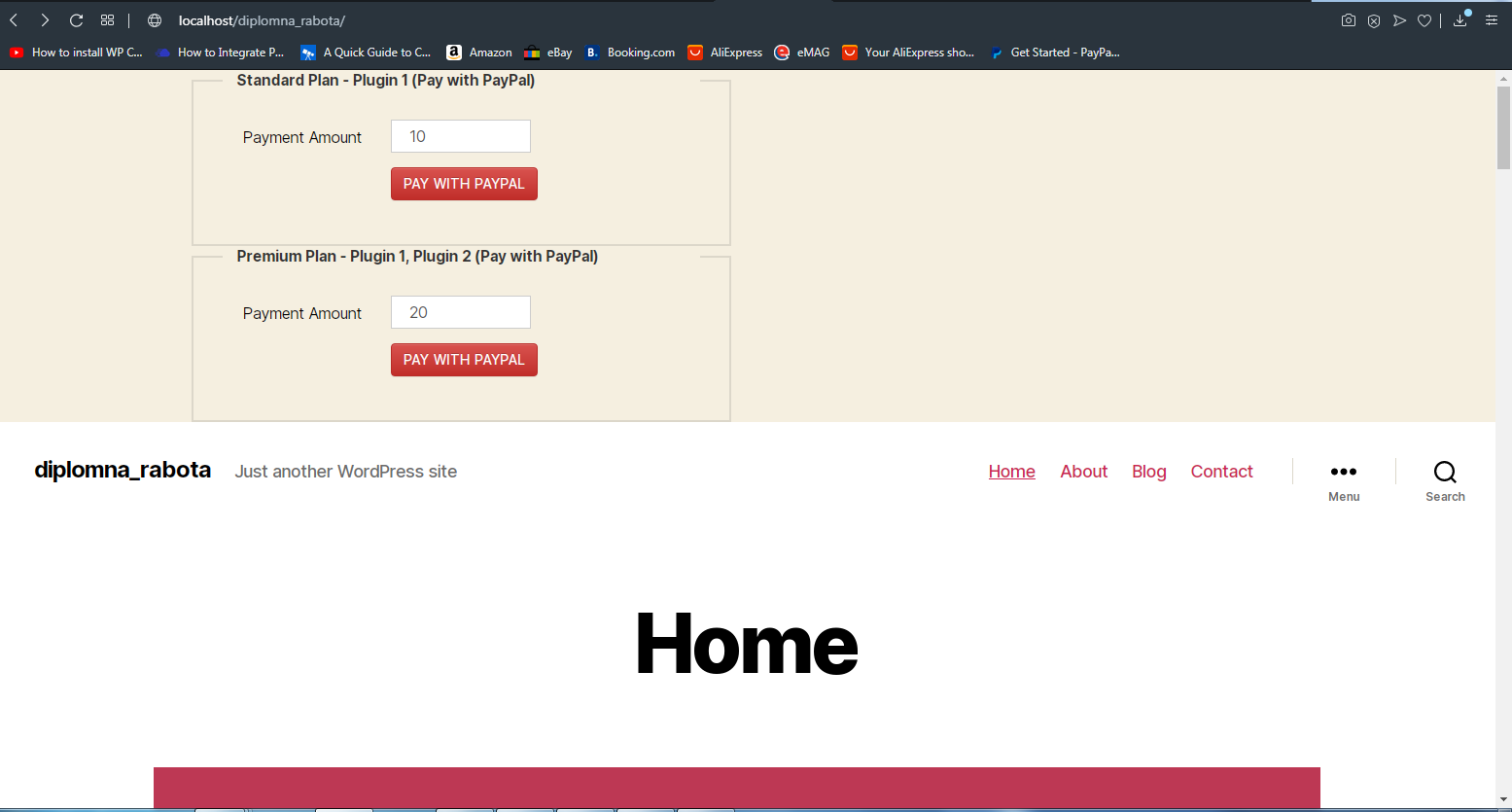
По подразбиране за хостване на уебсайта се използва localhost (в полето „Database Host“ се пише „localhost“, ако вече не е автоматично въведено).



Фигура 4.9. Завършване на инсталацията на WordPress

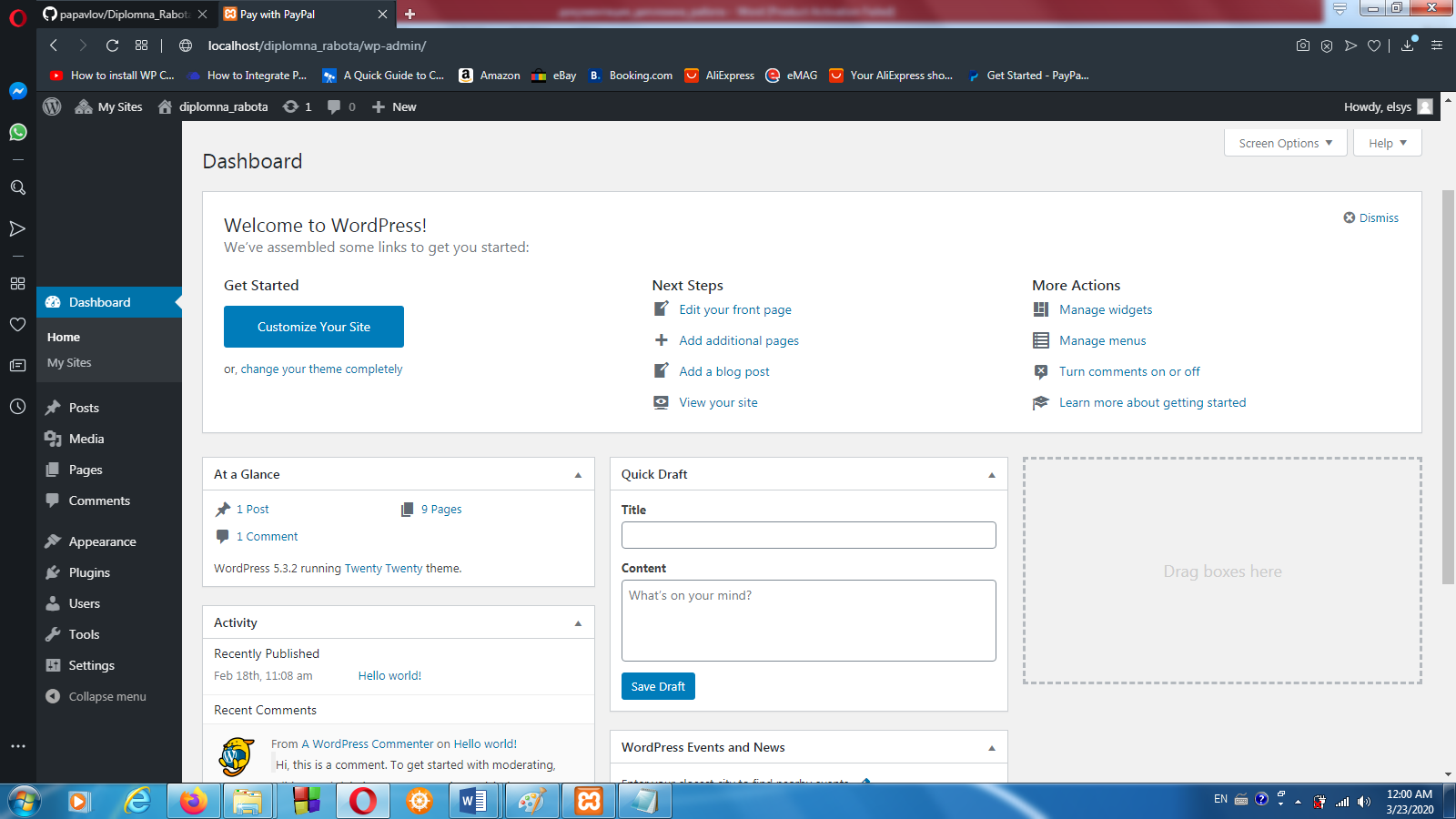
След като посочените стъпки бъдат изпълнени, се натиска „Submit“ и инсталацията е готова.

Уебсайтът може да бъде достъпен чрез адреса localhost/diplomna\_rabota, който се въвежда в браузъра. Той отвежда към началната страница на уеб страницата.



Фигура 4.10. Началната страница

На адреса localhost/diplomna\_rabota/wp-admin може да се достъпи таблото за управление на WordPress.



Фигура 4.11. Табло за управление на WordPress

**Подменюта:**

* Home -> My Sites – чрез избиране на това подменю могат да се видят всички налични уебсайтове към дадения момент в Multisite средата.
* Posts – виждат се вече съществуващите постове в уебсайта (ако има такива). Могат да се създават нови публикации, както и да се редактират.
* Media – използва се за добавяне и изтриване на снимки и видеа от уебсайта.
* Pages – в това подменю могат да се видят всички съществуващи към момента страници в уебсайта. Те могат да бъдат редактирани, изтривани, както е възможно и да се създават нови страници.
* Comments – тук се съдържат всички коментари направени от потребители в уебсайта (обикновено се иползва при блогове, където под статии има коментари).
* Appearance – в това подменю може да се променя дизайна на уеб страницата – да се смени темата, да се променят цветовете и др.
* Plugins – всички инсталирани приставки към дадения момент на уебсайта. Могат да се активират, да се изключват, както и да се инсталират нови плъгини (собствени custom или такива, които WordPress предоставя).
* Users – съдържа всички потребители в уебсайта, като може както да се добавят нови, така и да се редактират и изтриват вече създадени.
* Tools – в това подменю могат да се импортират (от други) и експортират публикации и коментари и т.н.
* Settings - основни настройки на уеб сайта. В това подменю може да се редактират името и описанието на сайта, имейлът на администратора му и др.

# ЗАКЛЮЧЕНИЕ

Изпълнението на този проект има минимални софтуерни и хардуерни изисквания. Поради тази причина може да бъде инсталиран на почти всякаква компютърна машина.

В тази дипломна работа успешно се реализира WordPress framework за потребителски планове и разплащания в Multisite среда. Плащанията към съществуващ WordPress уебсайт се извършват чрез системата PayPal, чрез скрипт, написан на php, в който е включена и библиотеката за плащания Omnipay. Направена е и multisite среда в WordPress, в която може да се създават нови събсайтове, които могат да се управляват от администратора, като въведена промяна в дизайна например може да важи както само за един от сайтовете в мрежата, така и за всички. Относно създаването на нов събсайт и потребител с такива права, че той да не може да влиза в настройките на приставките на сайта и да ги променя, както и за конфигурирането на съответните плъгини, включени в даден абонаментен план, са използвани команди от WP-CLI. За интегрирането им в кода е използвана функцията shell\_exec(). Остава да се довърши функционалността цените на абонаментните планове да се изтеглят динамично от sales сайт – това означава те да бъдат съобразявани според тези на конкурентни подобни приложения.

В бъдеще могат да се вземат в предвид и подобрения в дизайна на дипломната работа, както и да се създадат повече абонаментни планове, като всеки от тях да включва повече приставки. По-този начин предлаганите уебсайтове ще имат повече функционалности, което ще подобри качеството на продукта.

# ИЗПОЛЗВАНА ЛИТЕРАТУРА

1. WordPress frameworks - <https://digital.com/blog/wordpress-frameworks/>

2. Платежни системи - <https://mdzero.com/payment-systems/>

3. Stratus 5 - <https://stratus5.com/about-2/>

4. SaaS - <https://en.wikipedia.org/wiki/Software_as_a_service>

5. white-label - <https://en.wikipedia.org/wiki/White-label_product>

6. Stratus5 - <https://stratus5.com/solutions/>

7. ePay.bg - <https://bg.wikipedia.org/wiki/EPay.bg>

8. PayPal - <https://bg.wikipedia.org/wiki/PayPal>

9. Payoneer - <https://en.wikipedia.org/wiki/Payoneer>

10. WordPress - <https://en.wikipedia.org/wiki/WordPress>

11. XAMPP - <https://en.wikipedia.org/wiki/XAMPP>

12. phpMyAdmin - <https://en.wikipedia.org/wiki/PhpMyAdmin>

13. php - <https://www.w3schools.com/php/>

14. php - <https://en.wikipedia.org/wiki/PHP>

15. Причини за избор на php - <https://d5creation.com/php-for-web-development/>

16. Базата данни по подразбиране на WordPress - <https://deliciousbrains.com/tour-wordpress-database/?fbclid=IwAR0qj6MT8Yb4C003i3aEZHD1SxODMCHqqKUaPc5zChjz9lBgYZC_I99x8ss>

17. Sandbox - <https://searchsecurity.techtarget.com/definition/sandbox>

18. Composer - <https://getcomposer.org/doc/00-intro.md>

19. PayPal Integration - <https://www.cloudways.com/blog/paypal-integration-in-php/>

20. WP-CLI - <https://wp-cli.org>

21. shell\_exec() - <https://www.php.net/manual/en/function.shell-exec.php>

22. Create site WP-CLI commands - <https://developer.wordpress.org/cli/commands/site/create/>

23. Configure plugins WP-CLI commands - <https://developer.wordpress.org/cli/commands/plugin/activate/>

24. Create user WP-CLI commands - <https://developer.wordpress.org/cli/commands/user/create/>

25. WordPress roles - <https://www.wpbeginner.com/beginners-guide/wordpress-user-roles-and-permissions/>

**СЪДЪРЖАНИЕ**

[Отзив на научния ръководител и предложение за рецензент 3](#_Toc40980260)

[УВОД 4](#_Toc40980261)

[ПЪРВА ГЛАВА 6](#_Toc40980262)

[1.1. Съществуващи решения и подобни реализации 6](#_Toc40980263)

[1.2. Системи за онлайн разплащания 8](#_Toc40980264)

[ВТОРА ГЛАВА 12](#_Toc40980265)

[2.1.Функционални изисквания към WordPress базиран framework 12](#_Toc40980266)

[**2.1.1. Поддръжка на онлайн разплащания през PayPal към съществуващ WordPress уебсайт.** 12](#_Toc40980267)

[**2.1.2. Регистрация на нов WordPress сайт в Multisite среда спрямо плащане по предварително зададени планове.** 12](#_Toc40980268)

[**2.1.3. Конфигуриране на активни плъгини спрямо регистрирания план.** 12](#_Toc40980269)

[**2.1.4. Настройки на плана от админ панела.** 12](#_Toc40980270)

[**2.1.5. Динамично изтегляне на цените от sales сайт.** 12](#_Toc40980271)

[2.2. Използвани технологии и аргументация за избора им 13](#_Toc40980272)

[**2.2.1. WordPress.** 13](#_Toc40980273)

[**2.2.2. XAMPP.** 18](#_Toc40980274)

[**2.2.3. phpMyAdmin** 20](#_Toc40980275)

[**2.2.4. php** 22](#_Toc40980276)

[2.3. База данни на WordPress по подразбиране 23](#_Toc40980277)

[ТРЕТА ГЛАВА 33](#_Toc40980278)

[ЧЕТВЪРТА ГЛАВА 55](#_Toc40980279)

[ЗАКЛЮЧЕНИЕ 65](#_Toc40980280)

[ИЗПОЛЗВАНА ЛИТЕРАТУРА 66](#_Toc40980281)

**ПРИЛОЖЕНИЕ**