МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«АНАЛІЗ РЕАЛІЗАЦІЙ ТИПОВИХ КОМПОНЕНТІВ САЙТУ З ВИКОРИСТАННЯМ ФРЕЙМВОРКІВ CODEIGNATER, LARAVEL ТА YII2»**

Виконав(ла): студент(ка) 2 курсу, групи 8.1219
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
Д.В. Синєпольський
(ініціали та прізвище)

Керівник професор кафедри програмної інженерії, доцент, д.т.н. Чопоров С.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач і доцент кафедри комп'ютерних наук, доцент, к.т.н. Борю С.Ю.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент
Лісняк А.О.
(підпис)

« » 2020 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ(СТУДЕНТЦІ)

Синєпольському Дмитру Володимировичу
(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Аналіз реалізацій типових компонентів сайту з використанням фреймворків CodeIgnater, Laravel та Yii2

керівник роботи (проекту) Чопоров Сергій Вікторович, д.т.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 20 » травня 2020 року № 576-с

2. Строк подання студентом роботи 30.11.2020

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Аналіз реалізацій типових компонентів сайту

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 28.04.2020

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	28.04.2020	Виконано
2.	Збір вихідних даних.	30.05.2020	Виконано
3.	Обробка методичних та теоретичних		
	джерел.	25.06.2020	Виконано
4.	Розробка першого та другого розділу.	20.07.2020	Виконано
5.	Розробка третього та четвертого розділу.	20.09.2020	Виконано
6.	Оформлення та нормоконтроль		
	кваліфікаційної роботи.	04.11.2020	Виконано
7.	Захист кваліфікаційної роботи.	16.12.2020	Виконано

Студент _____ Д.В. Синєпольський
(підпис) (ініціали та прізвище)

Керівник роботи _____ С.В. Чопоров
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.В. Кудін
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Аналіз реалізацій типових компонентів сайту з використанням фреймворків CodeIgnater, Laravel та Yii2»: 82 с., 16 рис., 1 табл., 28 джерел, 3 додатки.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ІНТЕРНЕТ-МАГАЗИН, LARAVEL ФРЕЙМВОК, YII2 ФРЕЙМВОРК, CODEIGNITER ФРЕЙМВОРК, VUEJS ФРЕЙМВОК.

Об’єктом дослідження є WEB-додаток.

Мета роботи: аналіз реалізацій типових компонентів сайту.

Методи дослідження: спостереження, системний аналіз, порівняння, узагальнення.

Отримані висновки та їх новизна: у роботі проаналізовано та систематизовано вільне програмне забезпечення, що має можливість збереження і додавання публікацій в базу даних MySQL та реактивного відображення за допомогою JavaScript.

Результати випускної роботи мають науково-теоретичне та практичне значення. Висновки, пропозиції й рекомендації, що містяться у роботі щодо основних напрямів здійсненого дослідження, можуть бути використані у цілях розробки простого інтернет-магазину, застосовуватися у практичній діяльності, під час вивчення інженерії програмного забезпечення.

SUMMARY

Master’s Qualification Thesis «Analysis of typical site components implementations using CodeIgnater, Laravel and Yii2 frameworks»: 82 pages, 16 figures, 1 tables, 28 references, 3 supplements.

SOFTWARE, ONLINE STORE, LARAVEL FRAMEWORK, YII2 FRAMEWORK, CODEIGNITER FRAMEWORK, VUEJS FRAMEWORK.

The object of the study is the WEB-application.

The aim of the study is analysis of implementations of types of site components.

The methods of research are observation, system analysis, comparison, generalization.

The obtained conclusions and their novelty: in the work analyzed and systematized free software that has the ability to save and add publications to the database in the MySQL environment and reactive mapping using JavaScript.

The results of the graduation work have a scientific-theoretical and practical significance. Conclusions, suggestions and recommendations contained in the work on the main directions of the research carried out can be used to develop a simple online store, to be applied in practice, while studying software engineering.

ЗМІСТ

ВСТУП

Популярність створення вебресурсів сприяла розробці різних систем і програм, які спрощують процес написання сайту. Також вони допомагають підвищити ефективність роботи, а також дозволяють розробнику сфокусуватися на основну логіку програми.

Фреймворк – програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту.

Для запуску веб-додатка, мені знадобиться веб-сервер з підтримкою PHP версії 7.3.

Для розробників, які бажають використовувати YII2, вкрай корисним буде розуміння концепції об'єктно-орієнтованого програмування (ООП), так як YII2 – це строго об'єктно-орієнтований фреймворк.

Метою роботи є аналіз реалізацій типових компонентів сайту на PHP-фреймворках та використання мови програмування PHP в створенні додатка на YII2, Codeigniter та Laravel.

Практична цінність. Був реалізований практично приклад на основі якого зроблені відповідні висновки. Робота може бути корисна студентам, програмістам і розробникам сайтів.

Робота містить вступ, 3 розділи, висновок і бібліографію.

У першому розділі описується що собою являє фреймворк, його концепція і випадки, в яких можна застосувати PHP фреймворк.

У другому розділі описана практична частина роботи – встановлення самого фреймворка і його використання.

У третьому розділі описана реалізація на обраних фреймворках.

1 PHP-ФРЕЙМВОРК

1.1 Фреймворк програмної системи

Фреймворк – це набір всіляких бібліотек (інструментів) для швидкої розробки повсякденних завдань.

Фреймворк (англ.framework, син. каркас) – в інформаційних системах структура програмної системи; програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту [9]. На відміну від бібліотек, які об'єднують набір підпрограм близькою функціональності, фреймворк містить в собі велику кількість різних за призначенням бібліотек. Вживається також слово каркас, а деякі автори використовують його в якості основного, в тому числі не базуючись взагалі на англомовному аналогу. Можна також говорити про каркасному підході як про підхід до побудови програм, де будь-яка конфігурація програми будується з двох частин: перша, постійна частина – каркас, не змінний від конфігурації до конфігурації і несе в собі гнізда, в яких розміщується друга, змінна частина – змінні модулі (або точки розширення).

1.2 Концепція PHP-фреймворків

PHP-фреймворки за останнім часом набрали популярність, і стали базовою платформою для розробки веб-додатків. Іншими словами можна сказати, що вони забезпечують основну структуру програми. Використання PHP-фреймворків, дозволяє економити велику кількість часу, зменшити навантаження на процес розробки, позбавляючи від проблеми повторюваного коду, і швидко створювати додатки. Без використання PHP-фреймворків,

набагато складніше створювати веб-додатки, супроводжувати і модернізувати їх. Тим часом, використання PHP фреймворків робить процес створення програми набагато легшим і функціональним.

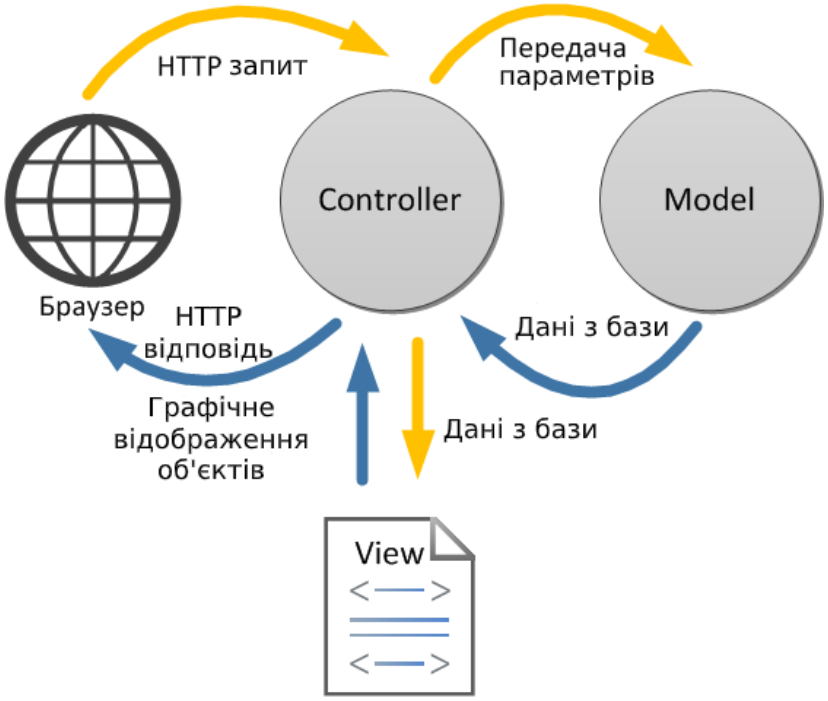

Рисунок 1.1 – Концепція PHP-фреймворків

Зараз більшість PHP проектів побудовані за допомогою архітектури Model View Controller (MVC). MVC – це архітектурний шаблон проектування, який використовується в більшості мов програмування і дозволяє відокремити бізнес-логіку від призначеного для користувача інтерфейсу, а так само виділити область логіки яка виробляє обмін інформацією між базою даних і призначеним для користувача інтерфейсом [15]. Таким чином можна змінити логіку додатка, не зачіпаючи інтерфейсної частини, або навпаки, що дуже добре для дизайнерів і верстальників. Це дозволяє уникнути плутанини і спрощує весь процес розробки. Коли йдеться про MVC, то мається на увазі: Model – та частина архітектури, яка взаємодіє з базою даних, View – представляє ту частину, яку безпосередньо бачить користувач, тобто графічний інтерфейс, і Controller – це область логіки, яка

контролює і управляє всіма її складовими і даними. Більшість сучасних фреймворком беруть за основу саме архітектуру MVC. Так само в сучасному фреймворку використовується шаблон проектування Front Controller, який, в залежності від запиту, перенаправляє його на потрібний контролер. Без Front Controller розробка із застосуванням фреймворка не мала б сенсу.

1.3 Випадки використання PHP-фреймворків

Для того щоб скористатися всіма можливостями фреймворка, потрібен чималий багаж знань в розробці додатків. PHP-фреймворки можуть допомогти усунути дуже часту помилку при програмуванні додатків, а саме повторення коду, а також систематизувати процес розробки. Фреймворки є потужним інструментом для динамічного мови програмування як PHP, які допоможуть організувати ваш код.

Кожна людина має різні вподобання і потреби. Для одного розробника використання PHP-фреймворків може допомогти в прискоренні процесу програмування, а для іншого це може здатися марною тратою часу. У більшості випадків це залежить від рівня професіоналізму, але, в загальному, PHP-фреймворки призначені, щоб заощадити час і абстрагуватися від рутинних завдань.

В основному, PHP-фреймворки застосовуються для розробки проектів складніше ніж 2-х сторінковий сайт з текстовими сторінками.

1.4 Огляд сучасних фреймворків

Після проведення невеликого аналізу і читання відгуків про фреймворками від великого числа розробників різних рівнів, з точки зору

зручності розробки, швидкості, стабільності, було виділено 3 популярних PHP-фреймворків, які відповідають більшості вимог.

1.4.1 Codeigniter

Codeigniter – популярний MVC фреймворк з відкритим вихідним кодом, написаний на мові програмування PHP, для розробки повноцінних веб-систем і додатків. Розроблено компанією EllisLab, а також Ріком Еллісом (Rick Ellis) і Полом Бурдик (Paul Burdick) [23].

Codeigniter відрізняє простота, яка досягається завдяки наступним факторам:

1. Гарна документація;
2. Розвинуте співтовариство;
3. Фреймворк дає свободу програмісту, не створюючи будь-яких структурних обмежень і конвенцій;
4. Програмісту не потрібно вчитися користуватися генераторами коду з командного рядка;
5. CodeIgniter працює практично на будь-якому хостинговому плані, який має підтримку PHP версії 5.1 і вище;
6. CodeIgniter вважається одним з найшвидших і не вимогливих до ресурсів фреймворків;
7. Підтримка баз даних MySQL, PostgreSQL, MSSQL, SQLite, Oracle.

Недоліки:

1. Підтримка PHP4 тягне за собою зайвий код;
2. Немає вбудованої ORM;
3. Немає вбудованої системи поділу прав;
4. Повільно розвивається;
5. Нестійкість до CSRF-атак.

1.4.2 YII

YII – це високоефективний заснований на компонентної структурі PHP-фреймворк для розробки масштабних веб-додатків. Він дозволяє максимально застосувати концепцію повторного використання коду і може істотно прискорити процес веб-розробки. Назва YII (вимовляється як Yee або [ji:]) означає простий (easy), ефективний (efficient) і розширюваний (extensible). Так само автор фреймворка, Qiang Xue каже що назва фреймворка спочатку означало Yes It Is [23].

Особливості:

1. Висока продуктивність;
2. Інтерфейси DAO і ActiveRecord для роботи з базами даних (PDO);
3. Підтримка інтернаціоналізації;
4. Кешування сторінок і окремих фрагментів;
5. Перехоплення і обробка помилок;
6. Введення та валідація форм;
7. Аутентифікація і авторизація;
8. Використання AJAX і інтеграція з jQuery;
9. Генерація базового PHP-коду для CRUD-операцій (скаффолдинг);
10. Підтримка тем оформлення для їх легкої зміни;
11. Можливість підключення сторонніх бібліотек;
12. Міграції бази даних;
13. Автоматичне тестування;
14. Підтримка REST;
15. Активна російськомовна спільнота.

Недоліки:

1. Фреймворк все ще молодий.

1.4.3 Laravel

Laravel – ідеальне рішення для тих, хто хоче швидко та грамотно створити безпечний, надійний веб-проект, при цьому завжди залишаючись на піку технологій веб-розробки [23].

Laravel має архітектуру MVC (модель-уявлення-контролер) та PHP-фреймворк побудований на базі компонентів Symfony. При необхідності налаштування модулів можна їх підключити у вигляді пакетів-провайдерів.

Код фреймворка відділений від коду розробника, завдяки цьому кожен компонент може легко розширюватися. CSS, JS, HTML-коди сторінок розділені в окремі директорії. Фреймворк Laravel має відмінний шаблонизатор Blade, за його допомогою можна відокремити верстку від PHP-коду.

До основних можливостей PHP-фреймворку Laravel:

1. Великий набір функціонала (для проєкту можна знайти будь-який функціонал);
2. Розміщена зручна адміністративна панель, (можна створити свою унікальну адміністративну частину);
3. Гарний захист бази даних (контроль захисту сайтів від SQL-ін'єкцій);
4. Велика масштабованість (функціонал проєкту можна розширити);
5. Зручна маршрутизація, валідація входять параметрів;
6. Кешування, робота зі сховищами файлів, робота з різними базами даних;
7. Міграції для бази даних, можна змінювати структурну частину бази даних та відміняти зміни;
8. Черги завдань, планувальник завдань, консоль, робота з SSH;

9. Широкий функціонал Eloquent ORM захищає від атак типу SQL Injection, а також завантажувати дані з декількох таблиць (вирішуючи проблему N + 1) або ж обробляти дані з БД частинами;
10. Прискорення роботи фреймворка за допомогою кешування файлів маршрутизації, файлів конфігурації, шаблонів;
11. Електронне листування: пошта, Slack і т.д.;
12. Підтримка WebSockets для створення справжніх інтерактивних додатків;
13. Підтримка багатомовності;
14. Інтерфейс командного рядка artisan, який генерує моделі, контролери, повідомлення, запускати завдання з черги завдань;
15. Laravel Tinker – додатковий пакет в якому можна працювати з кодом проекту з командного рядка;
16. Великий спектр для тестування веб-ресурсу.

Недоліки:

1. Високий рівень входження.

Код PHP-фреймворку Laravel гармонічний, тому він користується великою популярністю серед розробників. Адже кожну деталь продумують багато людей, щоб веб-додаток був взірцем для інших.

1.5 Розробка моделі бізнес-процесів

Модель бізнес-процесу традиційно є основною складовою управління бізнес-процесами. Оскільки об’єктом процесного управління є бізнес-процес, для можливості його розпізнавання, порівняння, аналізу та управління необхідно розділити на безліч ознак, що характеризують кожне властивість або здатність процесу.

Моделювання – це опис бізнес-процесу в заздалегідь обумовлених термінах, за правилами, так званими нотаціями. Модель бізнес процесу може бути як текстова, графічна або інформаційна [24].

Моделювання дозволяє обмінюватися інформацією про об’єкт моделювання без ризику втратити або спотворити інформацію про його внутрішніх властивостях. Модель бізнес-процесу дозволяє сконцентруватися на цільовій і значимій інформації про взаємозв’язки всіх об’єктів процесу. За рахунок цього простіше зрозуміти його хід процесу ніж, наприклад, за його словесним описом.

Моделлю бізнес-процесу називається його формалізований (графічний, табличний, текстовий, символьний) опис, що відображає реально існуючу або передбачувану діяльність підприємства. Модель, зазвичай, містить таку інформацію про бізнес-процеси:

- набір складових бізнес-функцій та порядок їх виконання;
- механізми контролю та управління в рамках бізнес-процесу;
- виконавців кожної бізнес-функції;
- вхідні та вихідні документи/інформація;
- ресурси, які необхідні для виконання кожної бізнес-функції;
- документацію, яка регламентує виконання кожної бізнес-функції;
- параметри, що характеризують виконання бізнес-функцій і процесу в цілому.

На сьогоднішній день існують різні програмні продукти, що дозволяють моделювати бізнес-процеси, що відбуваються на підприємствах. Бізнес-процеси підприємства електронної комерції були змодельовані за допомогою програмного продукту «Allfusion Process Modeler 7.0» (BPwin 7.0). Продукт володіє простим і інтуїтивно зрозумілим інтерфейсом. BPwin 7.0 підтримує три методології – IDEF0, IDEF3 і DFD, кожна з яких вирішує свої специфічні завдання.

IDEF0 – методологія функціонального моделювання і графічного описання процесів. Її ціль – формалізація і опис бізнес-процесів. Особливістю IDEF0 є те, що вона акцентує увагу на ієрархічне представлення об’єктів, що значно полегшує розуміння предметної області. Ідея IDEF0 полягає в тому, що бізнес-процес відображається у вигляді прямокутника, в якій входять і виходять стрілки.

Для IDEF0 має значення сторона процесу та пов’язаною з нею стрілкою:

- ліва сторона – вхід бізнес-процесу – інформація (документ) або ТМЦ, яка буде перетворена в ході виконання процесу;

- права сторона – вихід бізнес-процесу – перетворена інформація (документ) або ТМЦ;

- верхня сторона – управління бізнес-процесу – інформація або документ, який визначає те, як повинен виконуватися бізнес-процес, як має відбуватися перетворення входу у вихід;

- нижня сторона – механізм бізнес-процесу – те, що перетворює вхід в вихід: співробітники або техніка. Вважається, що за один цикл процесу не відбувається зміни механізму.

IDEF3 є стандартом документування технологічних процесів, що відбуваються на підприємстві, і надає інструментарій для наочного дослідження і моделювання їх сценаріїв. Цей метод призначений для моделювання послідовності виконання дій і взаємозалежності між ними в рамках процесів. Моделі IDEF3 можуть використовуватися для деталізації функціональних блоків IDEF0, що не мають діаграм декомпозиції. Діаграми IDEF3 відображають дію у вигляді прямокутника. Всі зв’язки в IDEF3 є односпрямованим і організовуються зліва направо. Метод IDEF3 дозволяє декомпозіровать дію кілька разів, що забезпечує документування альтернативних потоків процесу в одній моделі[5, 31]. Діаграми потоків даних (Data Flow Diagram) використовуються для документування механізмів

передачі і обробки інформації в модельованій системі. Діаграми DFD зазвичай будуються для наочного відображення поточної роботи системи документообігу організації. Найчастіше діаграми DFD застосовують як доповнення моделі бізнес-процесів, виконаної в IDEF0.

До основних об’єктів нотації відносять такі елементи:

- роботи (Activities) – відображають процеси обробки і зміни інформації;

- стрілки (Arrows) – відображають інформаційні потоки;

- сховища даних (Data Store) – відображають дані, до яких здійснюється доступ. Ці дані використовуються, створюються або змінюються роботами [4];

- зовнішня сутність (External entity) являє сутність поза контекстом системи, що є джерелом або приймачем даних системи. Передбачається, що об’єкти, представлені такими вузлами, не повинні брати участь ні в якій обробці.

1.5.1 Порівняння популярних фреймворків

Порівняння 6-ти популярних фреймворків за основними критеріями (необхідний рівень знань, сфера застосування, документація і т.д.).

Таблиця 1.1 – Порівняння популярних фреймворків

	Zend Framework 1	CakePHP	Code Igniter 2	Laravel	Symfony 2	YII 1.1
Опис	Академічно грамотний код. Дуже гнучкий. Вимагає гарного знання PHP і ООП. Доведеться трохи доводити під себе перш, ніж використовувати. Суха, але досить повна технічна документація.	Багато вбудованого функціоналу. Все досить тісно інтегровано. Документація не в кращому стані.	Майже мікрофреймворк. Дуже легкий для вивчення. Відмінна документація. Гнучкий. Легко використовувати сторонній код.	Легке створення веб-додатків з виразним та елегантним синтаксисом. Безкоштовний, з відкритим кодом.	Активно використовує командний рядок, yaml. Потужний ORM, хороша система view, генератори коду, dependency injection для всього. Вивчити дуже непросто, незважаючи на хорошу документацію	Простіше у вивченні, ніж Zend і Symfony. Увібрав краще від Rails. Непоганий AR, хороша система view, генератори коду. Досить тісна інтеграція.
Необхідний рівень знань	PHP5, ООП, шаблони проектування.	PHP, ООП, вміння розбиратися в вихідному коді фреймворка.	PHP, Основи ООП	PHP5, ООП, Консоль. Unity тести	PHP5, ООП, ORM, консоль.	PHP5, ООП.
Передбачувані проекти	Середні - великі	Маленькі - середні	Маленькі - великі	Маленькі - великі	великі	Маленькі - великі
PHP5.2	Так	Так	Так	Так	немає	Так
Жорстка структура каталогів	Ні (рекомендації)	Так	Так	Ні	Так	Ні (рекомендації)
Офіційна підтримка інтернаціоналізації	Так	Так	Так	Так	Так	Так
Складність установки	висока	низька	низька	низька	висока	Середня

Продовження табл. 1.1

Вимагає настройки	багато	трохи	трохи	трохи	багато	трохи
Повна підтримка ORM	немає	Так (не дуже зручна)	Ні (можна використовувати Doctrine)	Так	Так (Propel, Doctrine)	Active Record
Документація і приклади	хороша	є	відмінна	відмінна	У процесі написання	відмінна
Unit-тести для вихідного коду фреймворка	Так	Так	немає	Так	Так	Так
Англомовне співтовариство	Так	Так	Форум, Wiki, туторіали, блоги	Форум, блоги	Так	Так
Російськомовне співтовариство	Так	Майже не активно	документація, форум, блоги	Форум, блоги	немає	документація, форум, блоги
Ліцензія	New BSD	MIT	своя	MIT	MIT	New BSD

1.6 Переваги використання

1. Гнучкість розробки і розвитку проекту;
2. Ефективне використання ресурсів сервера;
3. Відкритий код фреймворка;
4. Легкість і надійність веб-розробок. Фреймворк складається з базових, перевірених, налагоджених функцій і операцій. Побудований на базі об'єктно-орієнтованого програмування;
5. Постійний розвиток і вдосконалення фреймворка;
6. Великий обсяг супровідної документації, прикладів з розробки на

різних мовах;

7. Світова популярність, велика кількість розробників;
8. Легкість супроводу проекту в подальшому, так як розробка із застосуванням фреймворка заснована на певних угодах;
9. Фреймворк дозволяє сконцентруватися на вирішенні архітектурних завдань, а не базових як при розробці без його застосування.

Фреймворк дозволяє вузько вирішувати поставлену задачу.

Одне з головних переваг фреймворка – це зручна розробка нестандартних проектів. Жоден великий нестандартний проект (наприклад, twitter.com, фотобанк з онлайн покупкою фотографій, сайт знайомств і т.д.) не роблять на готової CMS – вони для цього не призначені. Всі оригінальні проекти розробляють на фреймворками.

Веб-проект, розроблений на фреймворку, розвивається разом з вашим бізнесом. Змінюєтеся ви – змінюється сайт, достатньо лише замінити окремий блок (модуль), створити новий розділ або внести новизну в дизайні.

1.7 Недоліки застосування

З недоліків:

- підхід 1 файл = 1 клас;
- дуже багато коду не використовується і лежить мертвим вантажем в проекті;
- складність в освоєнні.

Ці недоліки досить умовні, достоїнств незрівнянно більше.

1.8 Вибір оптимального фреймворка для розробки сайту

При виборі PHP-фреймворку, можна трохи заплутатися з тим, що він повинен робити, і з тим для чого призначений фреймворк і що він виконує. Не кожен фреймворк підтримує ORM-шар для роботи з базами даних, має якісне співтовариство і хорошу документацію. Це може не перешкодити, якщо потрібен простий фреймворк. Однак, якщо необхідний фреймворк який би зручний і простий в освоєнні, то необхідно ретельно підійти до питання вибору фреймворка і зважити всі «за» і «проти».

1.8.1 Підтримка баз даних

Питання підтримки баз даних в PHP-фреймворк дуже важливий. Наприклад, CodeIgniter підтримує MySQL, Oracle і SQLite, а фреймворк Kohana не підтримує Oracle і SQLite. Частина фреймворків мають вбудований ORM-шар, частина – ні. Залежно від використовуваної бази даних для розробки проекту доводиться вибирати той чи інший PHP-фреймворк.

1.8.2 Підтримка спільноти

Для комфортного вирішення проблем повинен мати хороше співтовариство, не тільки з точки зору розміру, але і в якості і в готовності допомогти. Навіть якщо це маленьке співтовариство, але є зворотний зв'язок від спільноти, це можна вважати плюсом. Так само плюсом є наявність російськомовного співтовариства.

1.8.3 Документація

Частина фреймворк мають слаборозвинених, застарілу документацію. Частина не мають російської документації. Тому перед вибором фреймворка необхідно переконатися в тому що документація актуальна, вчасно оновлюється і доповнюється, і що інструкція із застосування проста в розумінні.

1.8.4 Продуктивність

Ключовим фактором при виборі так само може бути продуктивність фреймворка, наприклад, частина фреймворків підтримує кешування на достатньому рівні, частина — ні.

1.8.5 Безпека

Не всі фреймворки стійкі до різного роду атакам, тому перед вибором фреймворка необхідно ретельно проаналізувати активність розвитку, розмір спільноти, а так само наявність вбудованих засобів для захисту від атак.

1.8.6 Поріг входження

Не всі фреймворки прості в освоєнні, це дуже важливо враховувати при виборі, так як на освоєння одного фреймворка може не вистачити і року, а на освоєння іншого – вистачить всього тижня.

1.8.7 Швидкість розробки

Так само слід врахувати той факт, що на одному фреймворку проект розробляється швидше, на іншому – ні. Наприклад, розробки із застосуванням фреймворка zend триває більше ніж із застосуванням YII.

1.8.8 MVC архітектура

Фреймворк також повинен використовувати MVC архітектуру. Якщо цього у вас немає, то швиденько, ще раз погляньте в попередній розділ, для того щоб зрозуміти для чого він потрібен. Більшість хороших PHP-фреймворків мають бібліотеки, плагіни, модулі та розширення. Це дуже добре для того щоб реалізувати велике коло функціоналу та вдосконалити і прискорити процес розробки.

1.8.9 Швидкість розвитку фреймворку

Цей пункт так само дуже важливий, так як деякі фреймворки оновлюються кожні два роки (codeigniter), а деякі раз в пару місяців. Це дозволяє уникнути використання старого, «необкатаного», недопрацьованого коду при розробці.

1.8.10 Зворотня сумісність

Не всі фреймворки назад сумісні, тобто, при оновленні фреймворка в проекті може виникнути необхідність в повній переробці проекту. Частина фреймворків умовно назад сумісна, наприклад, при оновленні молодшої частини версії (minor) все сумісно, а при старшій – немає. Так само великим плюсом є керівництво по переходу на нову версію фреймворку.

1.8.11 Наявність вбудованих JavaScript-бібліотек

Для прикладу, YII та Laravel включає в себе jquery і jquery ui, а так само має вбудовані засоби для контролю підключених скриптів і черговості їх вибопненія, а codeigniter — немає.

1.8.12 Підтримка з боку хостингу

Частина фреймворків вимагають PHP 7.3, а частина буде працювати і на PHP 5. Якщо помилитися у виборі, то проект не буде працювати на хостингу.

1.9 Поширені помилки при виборі фреймворка

Будь-яка людина може помилитися при виборі PHP-фреймворку, однак можна захиститися від такого роду помилок. Тому необхідно переконатися, що обраний фреймворк має достатню функціональність і гарну підтримку, тому що зазвичай невеликі фреймворки створюються особами, знання PHP і навіть дещо обмежені. Це може викликати різні помилки і питання, які в кінцевому рахунку заважають і уповільнюють процес розробки.

Вибираючи PHP-фреймворк необхідно звертати увагу на те, наскільки він легкий в освоєнні і розумінні. Це має дуже важливе значення для малодосвідченого PHP програміста. Також необхідно переконатися, що база даних і веб-сервер сумісні з архітектурою обраного фреймворку.

Якщо не притримувати вищевказаних вимог, то можливі падіння в продуктивності розробки проекту, продуктивності коду і легкості супроводу. Іншою поширеною помилкою є неправильне встановлення

фреймворка. При установці необхідно слідувати інструкції, щоб уникнути помилок.

2 ПРОЕКТУВАННЯ

Як проект, для ознайомлення з можливостями фреймворків, була обрана розробка сайту-каталогу мобільних телефонів, який буде володіти такими особливостями:

1. Товари розміщені на головній сторінці;
2. Можливість швидко замовити товар;
3. Зручна навігація по сайту;
4. Гарний і зручний в користуванні UI/UX інтерфейс;
5. Підтримка високих навантажень (кешування).

Для розробки магазину були обрані такі фреймворки:

- YII2;
- Codeigniter 3;
- Laravel.

Вони мають гарну розширюваність, всі необхідні можливості для створення інтернет магазину а також великі російськомовні спільноти.

Роботу над створенням будь-якого сайту можна розділити на наступні етапи:

1. вибір і встановлення необхідних інструментів;
2. розробка дизайну проекту та верстки;
3. проектування і створення бази даних;
4. створення основи програми та конфігурація;
5. генерація каркаса коду за допомогою кодогенератор Gii;
6. встановлення верстки та доопрацювання каркаса.

Бізнес процеси які реалізують розрозблені магазини однакові, відрізняється лише їх реалізація на програмному рівні, тому описані алгоритми роботи можна застосувати до всіх трьох реалізацій функціоналу.

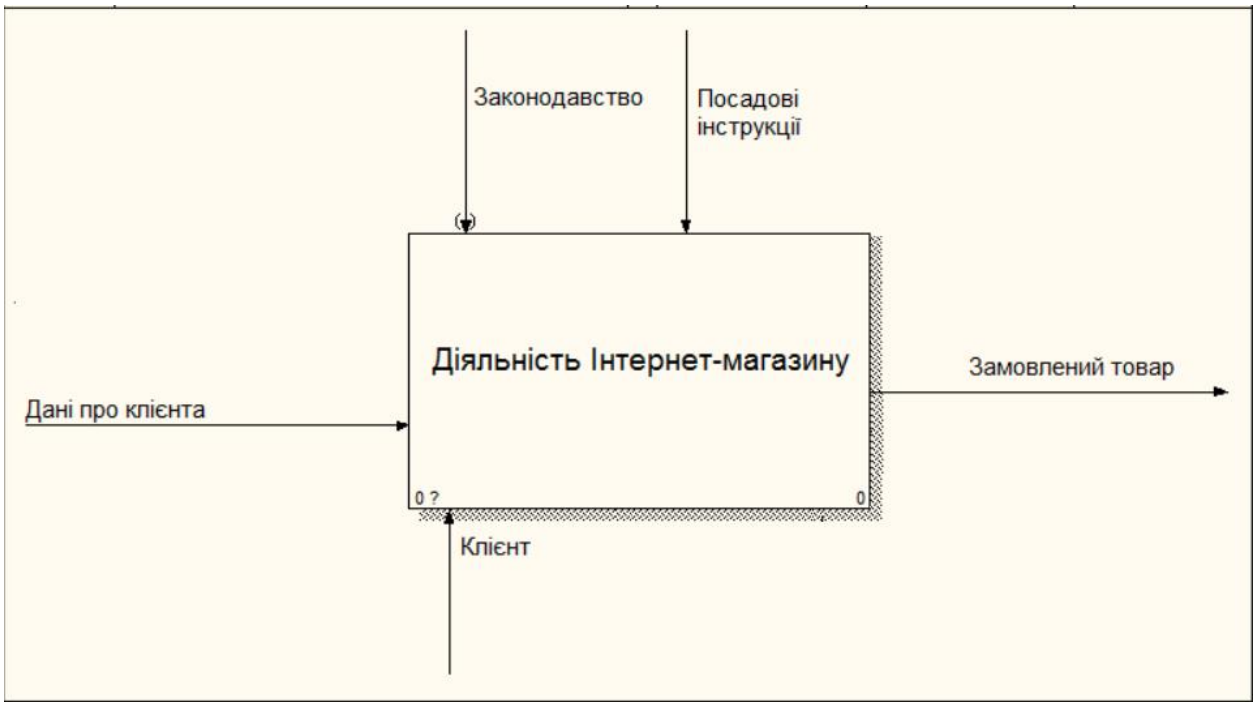

Рисунок 2.1 – Контекстна діаграма «Діяльність Інтернет-магазину»

Процеси, якими управляє менеджер інтернет магазину в системі AS-IS і TO-BE, як уже згадувалося раніше відрізняється наявністю додаткового модуля статистики, який відповідає за процеси збору і обробки статистичної інформації. Тобто ми розглядаємо роботу менеджера інтернет-магазину трохи ширше, ніж зазвичай. За рахунок обрабокі статистичної інформації, збільшується якість роботи магазину в цілому. Тобто після додавання до стандартних процесів, якими управляє менеджер-магазину додаткових процесів обробки статистичної інформації менеджер магазину і його роль переходять на більш високий рівень в системі управління роботою магазину в цілому.

Цей метод опису бізнес процесів побудований на основі IDEF3 стандарту призначений для моделювання послідовності виконання дій і взаємозалежності між ними в рамках процесів. Моделі можуть використовуватися для деталізації функціональних блоків, що не мають діаграм декомпозиції. Діаграми відображають дію у вигляді прямокутника. Всі зв’язки є односпрямованим і організовуються зліва направо.

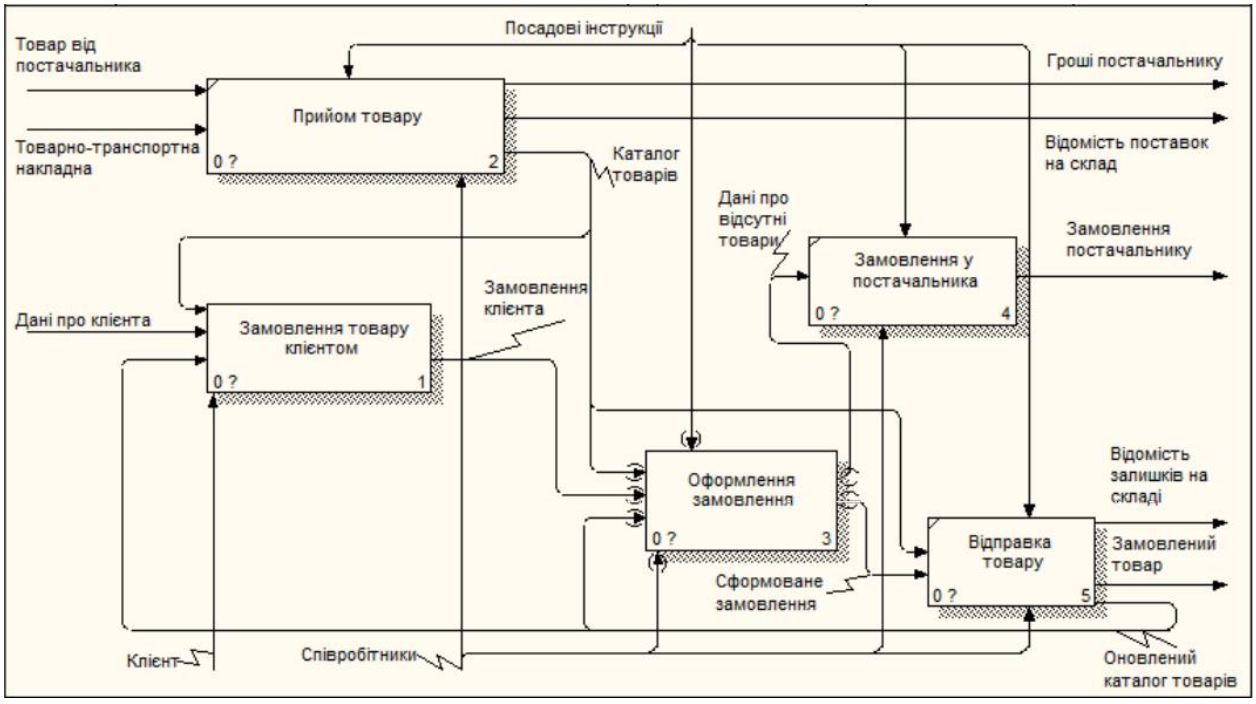

Рисунок 2.2 – Виконання замовлення і алгоритм роботи менеджера

Функціонал замовлення товару у вигляді бізнес процесу зображений на наступній діаграмі. На дій покрово описані дії що відбуваються від час замовлення. Всі кроки реалізовані в інтернет магазинах що були розроблені на різних фрейморках.

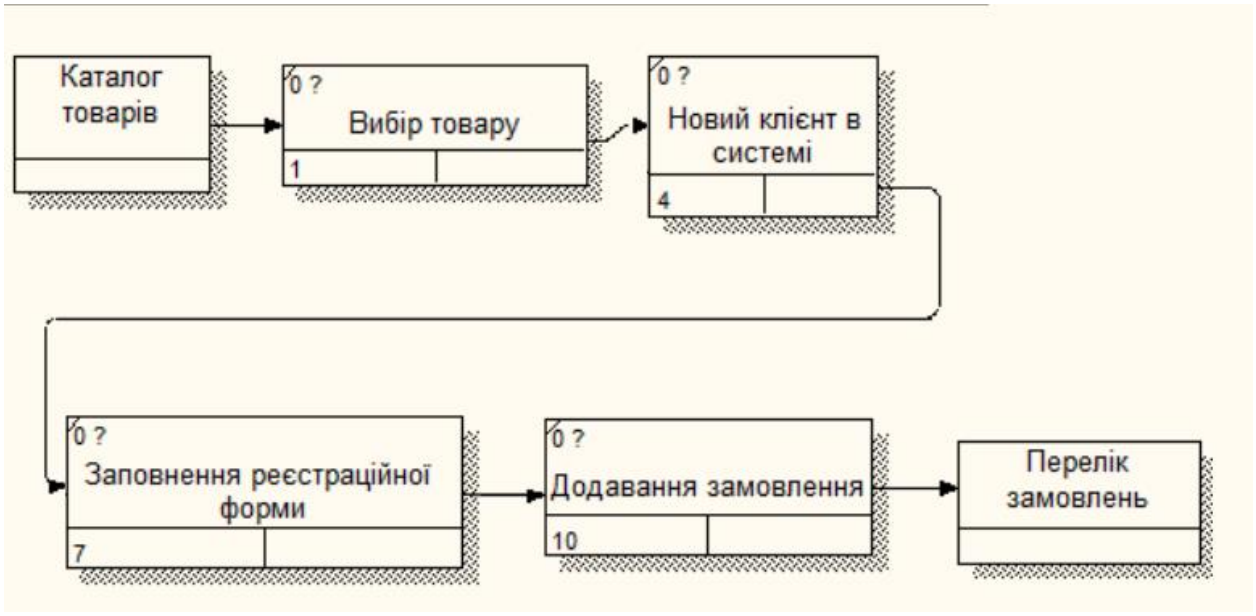

Рисунок 2.3 – Декомпозиція функціонального блоку «Замовлення товару клієнтом»

2.1 Додатки, використані при розробці проекту

Для оптимізації розробки проекту, а також зручності, були використані наступні програми:

1. Пакет «OpenServer»;
2. Графічний інтерфейс «PhpMyAdmin» для управління СУБД MySQL;
3. Редактор програмного коду PhpStorm.

Кожне з цих додатків має низку унікальних, незамінних властивостей, які полегшують розробку і тестування web-додатки.

2.1.1 Пакет для WEB-розробки «OpenServer»

Open Server Panel – це портативна серверна платформа і програмне середовище, створена спеціально для веб-розробників з урахуванням їх рекомендацій і побажань.

Програмний комплекс має багатий набір серверного програмного забезпечення, зручний, багатофункціональний продуманий інтерфейс, має потужні можливості з адміністрування та налаштування компонентів. Платформа широко використовується з метою розробки, налагодження і тестування веб-проектів, а так само для надання веб-сервісів в локальних мережах.

Хоча спочатку програмні продукти, що входять до складу комплексу, не розроблялись спеціально для роботи один з одним, така зв'язка стала дуже популярною серед користувачів Windows, в першу чергу через те, що вони отримували безкоштовний комплекс програм з надійністю на рівні Linux серверів.

Зручність і простота управління безумовно не залишать вас байдужими, за час свого існування Open Server зарекомендував себе як першокласний і надійний інструмент необхідний кожному веб-майстру.

Основні компоненти:

- Apache 2.2.31/2.4.38/2.4.41/2.4.43
- Bind 9.16.1
- ConEmu 19.10.12
- FTP FileZilla 0.9.60
- Ghostscript 9.52
- HeidiSQL 11.0.0.5944
- Nginx 1.17.10
- NNCron Lite 1.17
- Opera 67.0.3575.137
- Sendmail 32
- Sublime 3.2.2.3211
- Wget 1.20.3

Системи управління базами даних:

- MariaDB 5.5.67 /10.1.44 / 10.2.31 / 10.3.22 / 10.3.22 / 10.4.12
- Memcached 1.2.6 / 1.4.5 / 1.5.10
- MongoDB 2.4.14 / 2.6.12 / 3.0.15 / 3.2.22 / 3.4.24/ 3.6.17 / 4.0.17 / 4.2.5
- MySQL 5.1.73 / 5.5.62 / 5.6.47 / 5.7.29 /8.0.19
- PostgreSQL 9.2.24 / 9.3.25 / 9.4.26 / 9.5.21 / 9.6.17 / 10.12/ 11.7 / 12.2
- Redis 2.8.2402 / 3.0.504 / 3.2.100 / 4.0.14.2 / 5.0.6-dev

PHP модулі:

- PHP 5.4.45
- PHP 5.5.38
- PHP 5.6.40
- PHP 7.0.33
- PHP 7.1.33
- PHP 7.2.29
- PHP 7.3.17

Компоненти збірки представлені в 64-бітної і частково 32-бітної версіях. Кожна збірка перед релізом перевіряється антивірусами Dr.Web і Kaspersky, ми гарантуємо відсутність вірусів.

Системні вимоги:

- Підтримувані версії ОС: 64-біт Windows 7 SP1 або новіше (32-бітові системи не підтримуються);
- Мінімальні апаратні вимоги: 500 МБ вільної RAM і 4 ГБ вільного місця на HDD;
- Потрібна наявність Microsoft Visual C ++ 2005-2008-2010-2012-2013-2015-2019 Redistributable Package;

Можливості керуючої програми:

1. Непомітна робота в треї Windows;
2. Швидкі старт і зупинка;
3. Автостарт сервера;
4. Кілька режимів управління доменами;
5. Монтування віртуального диска;
6. Підтримка управління через командний рядок;
7. Підтримка профілів налаштувань;
8. Зручний перегляд логів всіх компонентів;
9. Перемикання HTTP, MySQL і PHP модулів;
10. Детальна і зрозуміла документація;
11. Доступ до доменів в один клік;
12. Швидкий доступ до шаблонів конфігурації;
13. Багатомовний інтерфейс;
14. Автозапуск програм за списком.

Особливості комплексу:

- Не вимагає установки (портативність);
- Можливість роботи з USB накопичувача;

- Одночасна робота з Denwer, Xampp і т.д .;
- Робота на локальному / мережевому / зовнішньому IP адресу;
- Підтримка SSL без всякої дополн. настройки;
- Створення домену шляхом створення звичайної папки;
- Підтримка кириличних доменів;
- Підтримка алиасов (доменних покажчиків);
- Захист сервера від зовнішнього доступу;
- Punycode конвертер доменних імен;
- Набір популярних сторонніх розширень PHP;
- Планувальник завдань (cron);
- Створення локального поддомена без втрати видимості основного домену в мережі інтернет.

2.1.2 Система управління СУБД MySQL «phpMyAdmin»

PhpMyAdmin – веб-додаток з відкрітім кодом на мові PHP з графічнім веб-інтерфейсом для адміністрування бази даних MySQL або MariaDB. phpMyAdmin дозволяє через браузер здійснювати адміністрування сервера MySQL, запускати SQL, переглядати та редагувати вміст таблиць баз даних.

Ця програма користується великою популярністю у веб-розробників, оскільки дозволяє керувати базами даних MySQL без введення SQL команд через інтерфейс і з будь-якого комп'ютера під'єднаного до інтернету без необхідності встановлення додаткового програмного забезпечення.

На сьогоднішній день phpMyAdmin широко застосовується на практиці. Останнє пов'язано з тим, що розробник інтенсивно розвиває продукт, з огляду на всі нововведення СКБД MySQL:

- Підтримка безлічі одночасно відкритих підключень за допомогою TCP / IP, іменованих каналів або SSH-тунелювання, з можливістю збереження авторизаційних даних;
- Управління користувачами і їх правами на сервері в рамках бази даних або глобально;
- Підтримка управління серверними змінними;
- Перегляд серверної статистики і управління запущеними процесами з можливістю проаналізувати виконувані SOL-запити і перервати "погані";
- Підтримка експорту баз даних в SQL файл або на інший сервер, з можливістю подальшого імпорту;
- Перегляд і управління базами даних, таблицями, відображеннями, тригерами і збереженими процедурами.

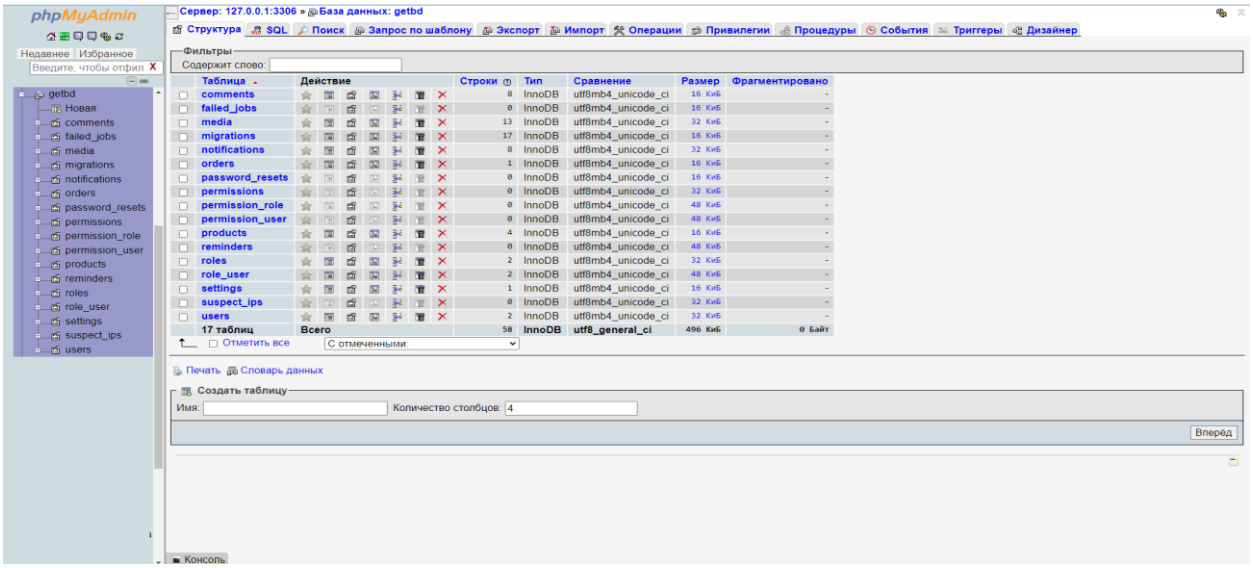

Рисунок 2.4 – Програма phpMyAdmin для роботи з базами даних

2.1.3 Редактор прогрманого коду PhpStorm

JetBrains PhpStorm – комерційне IDE для PHP розроблене на основі платформи IntelliJ IDEA (рис. 2.5). Надає «розумний» редактор для PHP,

HTML і JavaScript коду з підтримкою аналізу якості коду «на льоту» і просунутими можливостями автоматичного рефакторінга. Доповнення коду повністю підтримує всі можливості PHP 7.3, включаючи простір імен і замикання. Ключові особливості:

- Підтримка фреймворка для тестування PhpUnit;
- Підтримка налагодження PHP і JavaScript коду;
- Дистанційна вивантаження по FTP, SFTP, мережного диска з автоматичною синхронізацією;
- Підтримка специфікацій HTML5 і EcmaScript 5;
- Інтеграція з системами контролю версій: CVS, SVN, Git, Perforce, Mercurial;
- Інтеграція з баг-трекер.

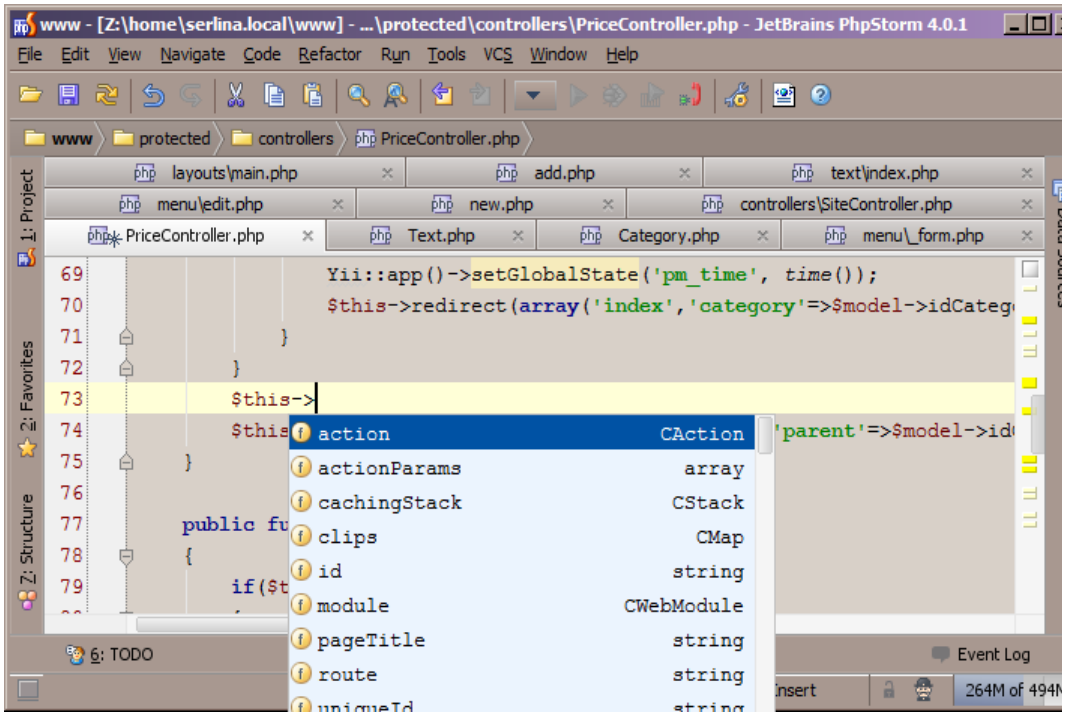

Рисунок 2.5 – IDE PhpStorm для редагування коду

2.2 Розробка дизайну проекту та верстки

Прототип дизайну створювався на основі базової структури інтерфейсів інтернет-магазинів метою якого було створення зручних для користувача веб-сторінок, що дозволять ефективно та приємно користуватися сервісом. Також зручний дизайн повинен забезпечити користувачу швидке отримання потрібного результату від сайту.

Взагалі UI/UX методи розробки передбачають цілісний і комплексний підхід до взаємодії з користувальницьким інтерфейсу. Розробляючи елементи веб-сайту з якими будуть взаємодіяти користувачі, намагався максимально врахувати всі дрібниці взаємодії, починаючи від дизайну для користувача та та закінчуючи зручним виглядом панелі керування.

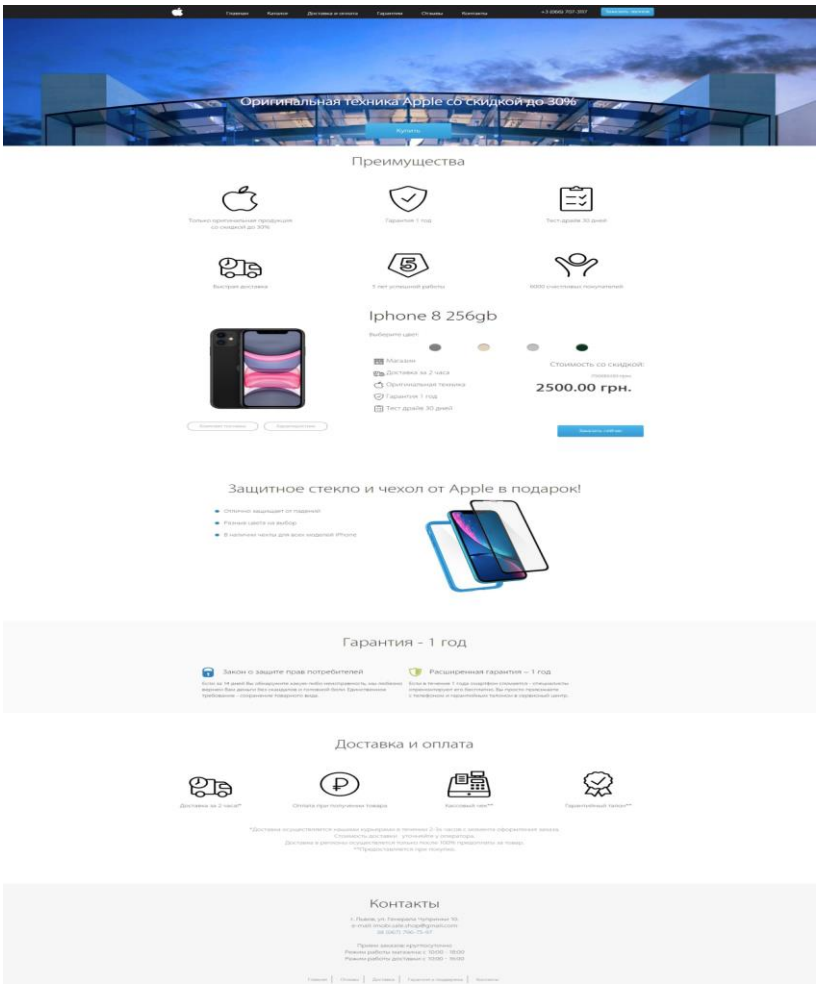

Рисунок 2.6 – Головна сторінка сайту

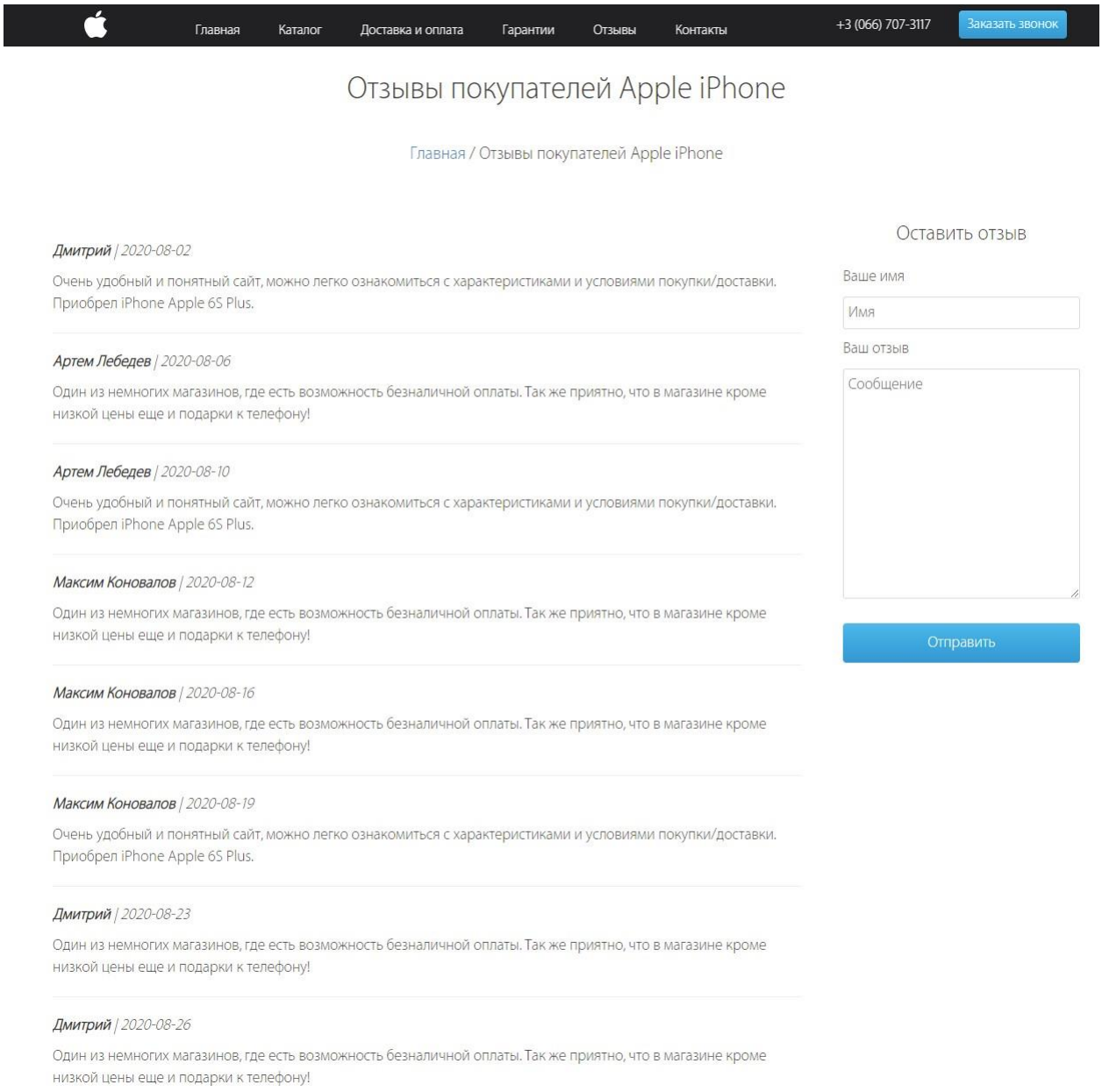

Рисунок 2.7 – Сторінка відгуків

У верхній частині сайту знаходиться навігація для зручного переміщення по сторінкам. Також список товарів зроблений у вигляді розгортаючого меню, по кліку користувача на модель яка його цікавить, відбудеться перехід но неї.

Також на сайті знаходиться окрема сторінка з правилами роботи магазину.

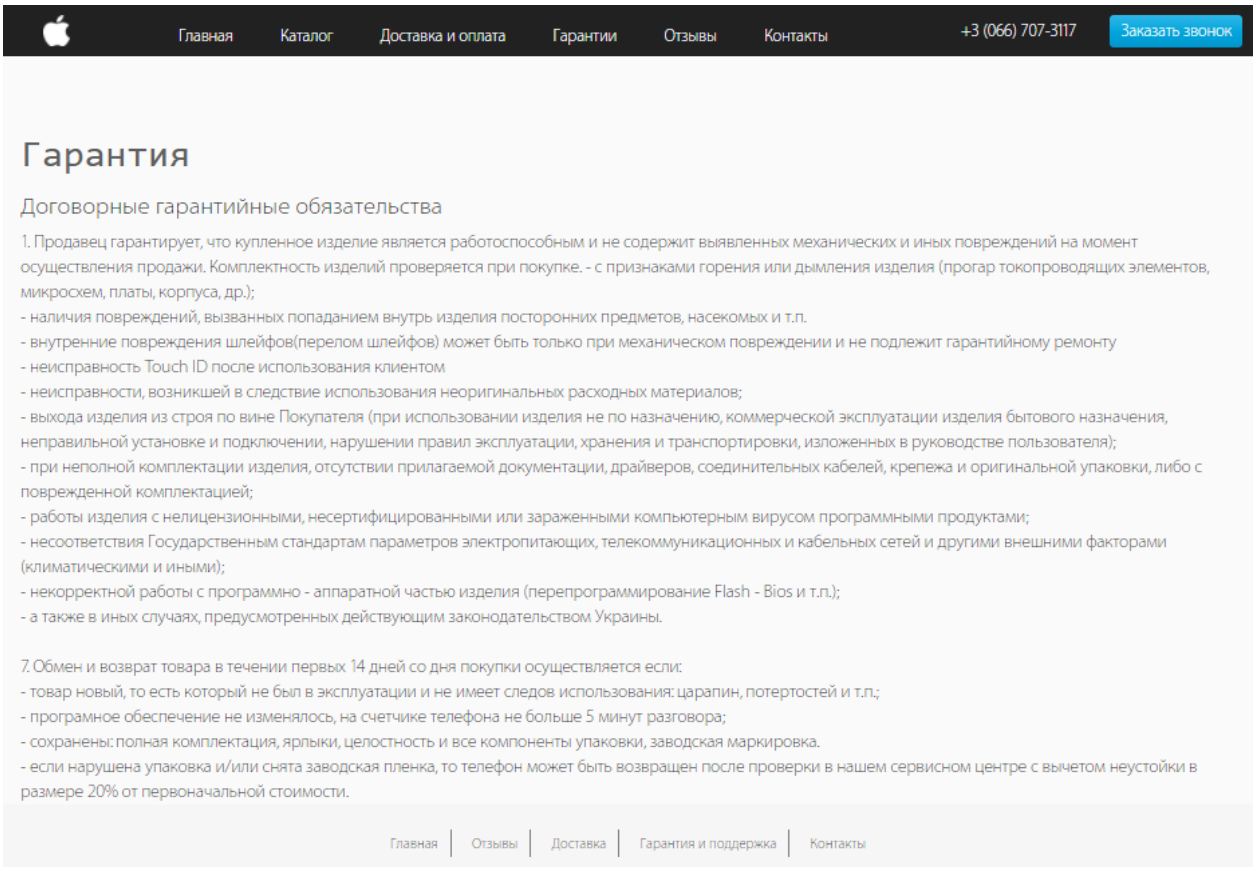

Рисунок 2.8 – Сторінка гарантій

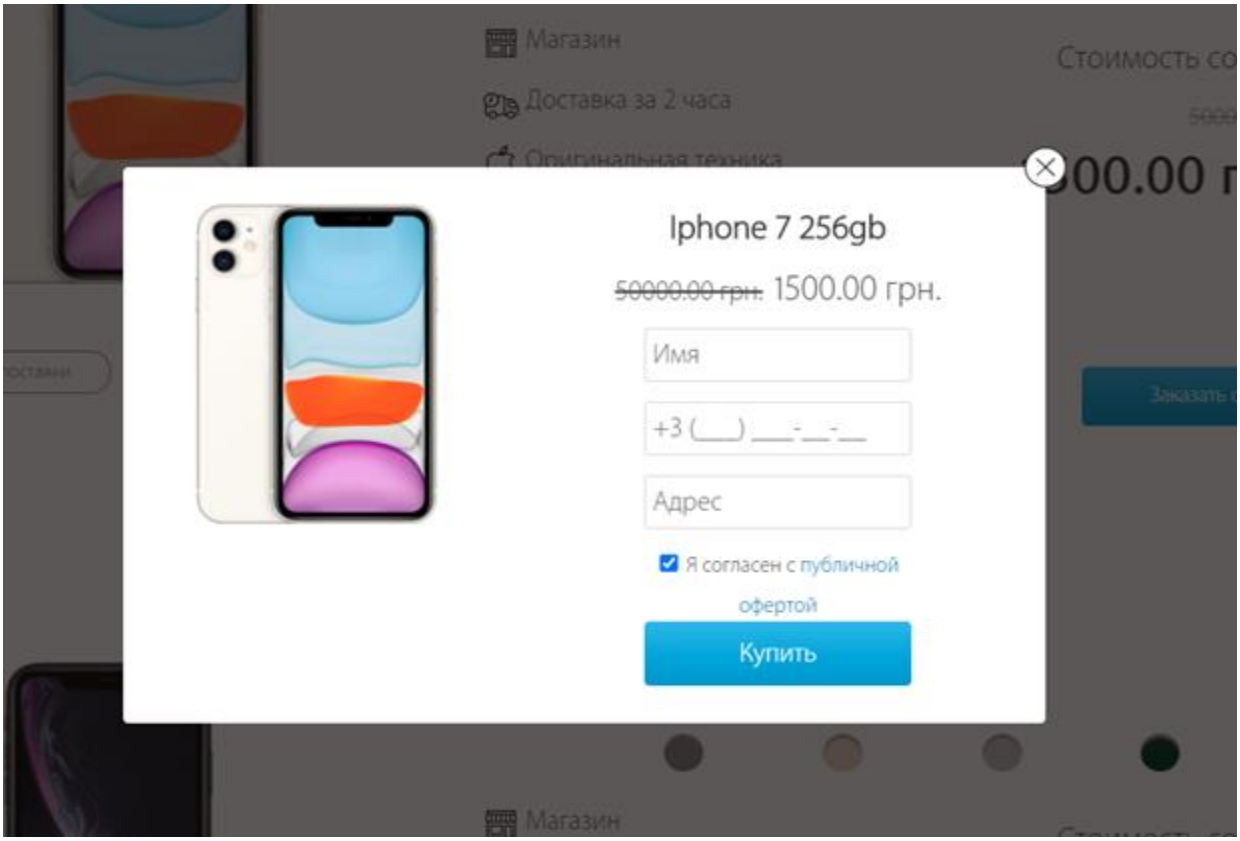

Рисунок 2.9 – Поп-ап замовлення товару

Під час замовлення користувач повинен ввести контактну інформацію про себе. Також він погоджується з правилами роботи магазину, для зручності напис є відразу посиланням на сторінку гарантій.

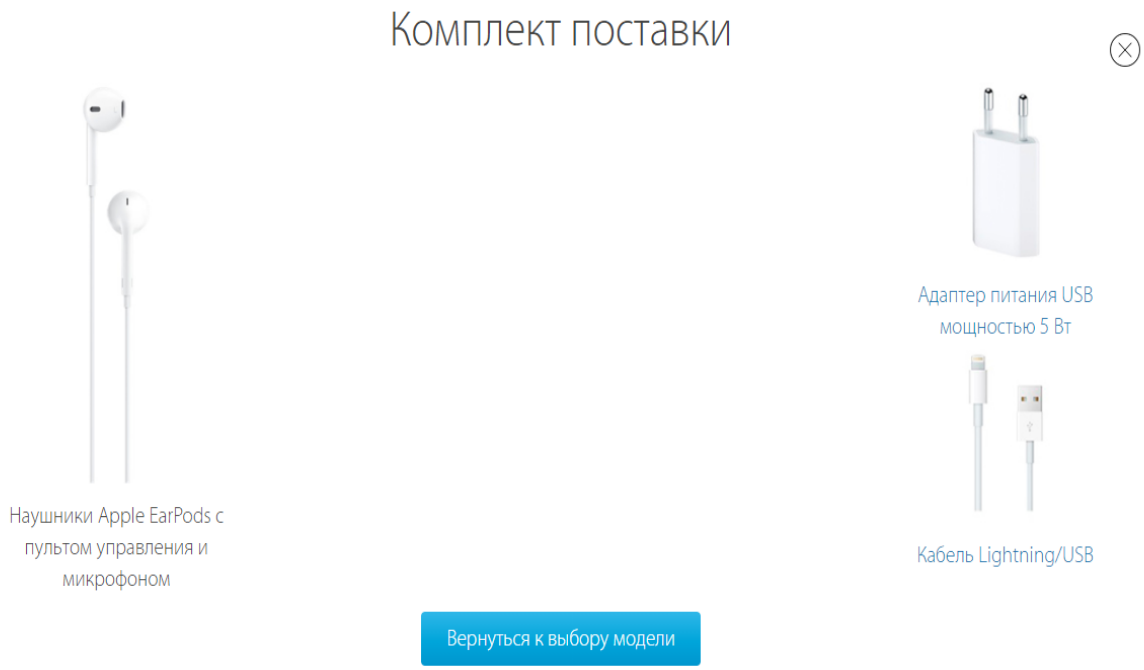

Рисунок 2.10 – Додаткова інформація про товар

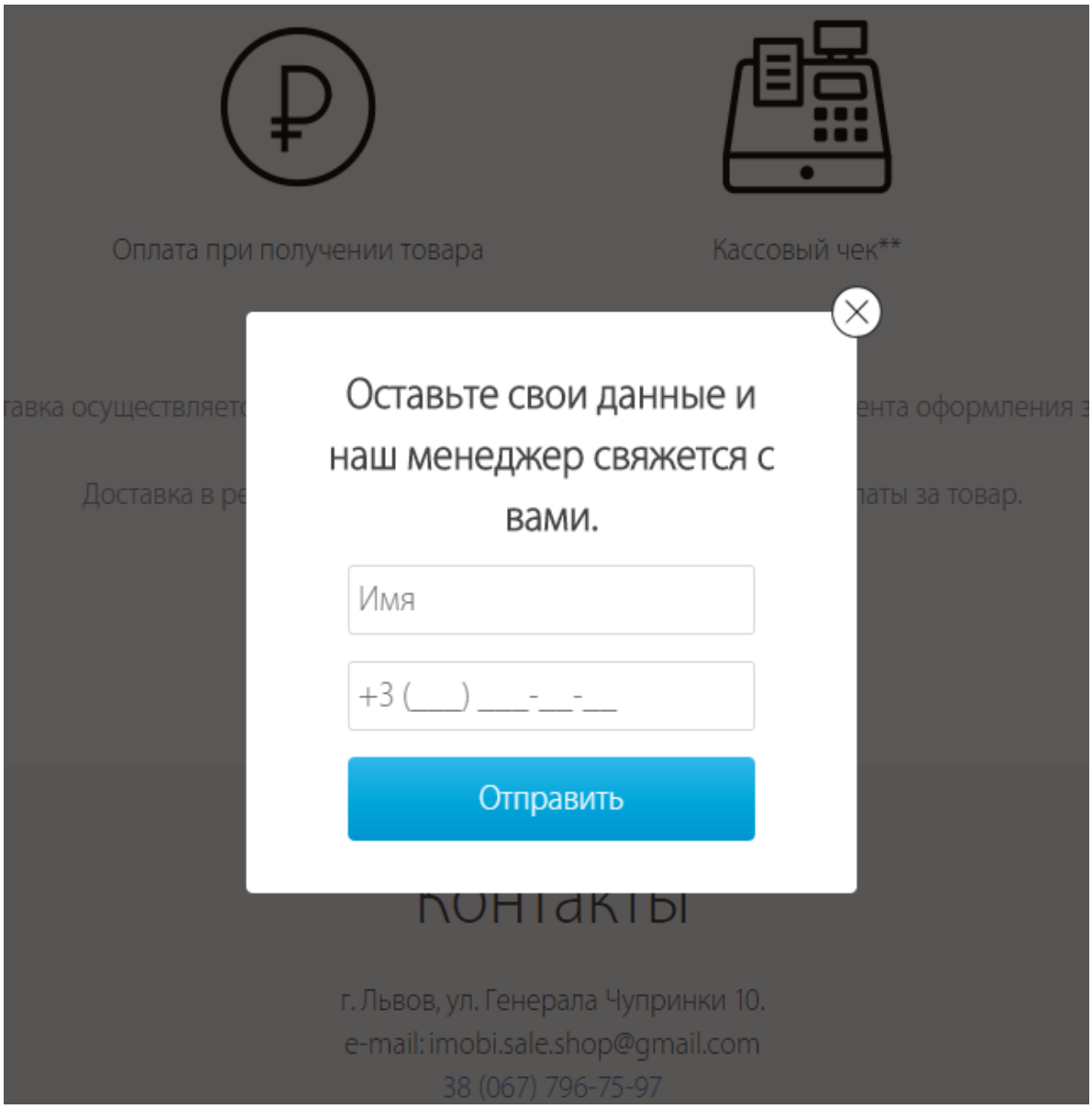

Рисунок 2.11 – Поп-ап замовлення дзвінку

Якщо користувача цікавлять запитання що до товару, він може замовити дзвінок з інтернет магазину через просту форму. Кнопка виклику форми знаходиться в головному меню.

2.3 Проектування і створення бази даних

Під час проектування база даних були виділені наступні таблиці.

Таблиця «products», яка зберігає в собі всю інформацію про товар. Його назву в меню, основну інформацію, також ціну товара, зображення і його варіанти кольору.

	#	Имя	Тип	Сравнение
☐	1	**id**	bigint(20)	
☐	2	**tag**	varchar(255)	utf8mb4_unicode_ci
☐	3	**title**	varchar(191)	utf8mb4_unicode_ci
☐	4	**general_info**	text	utf8mb4_unicode_ci
☐	5	**price**	double(8,2)	
☐	6	**old_price**	double(8,2)	
☐	7	**image**	varchar(191)	utf8mb4_unicode_ci
☐	8	**variations**	text	utf8mb4_unicode_ci
☐	9	**is_hidden**	tinyint(1)	
☐	10	**created_at**	timestamp	
☐	11	**updated_at**	timestamp	

Рисунок 2.12 – Таблиця products

Таблиця користувачів містить інформацію про дату створення, імя користувача якій залишив відгук а також текст самого коментаря.

	#	Имя	Тип	Сравнение
☐	1	**id**	bigint(20)	
☐	2	**name**	varchar(191)	utf8mb4_unicode_ci
☐	3	**email**	varchar(191)	utf8mb4_unicode_ci
☐	4	**text**	text	utf8mb4_unicode_ci
☐	5	**value**	tinyint(3)	
☐	6	**is_hidden**	tinyint(1)	
☐	7	**created_at**	date	
☐	8	**updated_at**	timestamp	

Рисунок 2.13 – Таблиця comments

3 РЕАЛІЗАЦІЯ

3.1 Створення основи програми та конфігурація

При створенні проекту використовувався віртуальний хост в OpenServer, розташований по шляху Z:\OpenServer\domains. Використання віртуального хоста полегшує розробку і тестування проекту.

Для створення програми на основі фреймворку YII, необхідно завантажити останню версію з офіційного сайту www.YIIframework.com, Розпакувати і скопіювати папку framework в Z:\OpenServer\domains\YII.loc.

Для зручності перенесення на інший web-сервер або хостинг, після створення, каркас додатка був винесений з папки magazin на один рівень вище, а в index.php і index-test.php виправлений шлях до фреймворку.

Файл конфігурації програми розташований за наступним шляхом: protected / config / main.php. Файл описує налаштування програми у вигляді асоціативного php-масиву. Для налаштування підключення до БД був налаштований компонент додатка db в секції components:

```
'Db' => array (
        // рядок у форматі dsn для підключення до БД
        'ConnectionString' => 'mysql: host = localhost; dbname = magazin',
        // Ім'я користувача
        'Username' => 'root',
        'Password' => '',
        'Charset' => 'utf8',
);
```

Для створення програми на основі фреймворку codeIgniter, необхідно завантажити останню версію з офіційного сайту https://codeigniter.com/, Розпакувати в Z:\OpenServer\domains\ codeigniter.loc.

Файл конфігурації програми розташований за наступним шляхом: \application\config. Тут знаходяться всі файли налаштувать.

Для налаштування роботи з базою даних, відредагуємо файл database.php:

```
 $active_group = 'default';
$query_builder = TRUE;
$db['default'] = array(
        'dsn'   => '',
        'hostname' => 'localhost',
        'username' => 'root',
        'password' => '',
        'database' => 'bd',
        'dbdriver' => 'mysqli',
        'dbprefix' => '',
        'pconnect' => FALSE,
        'db_debug' => (ENVIRONMENT !== 'production'),
        'cache_on' => FALSE,
        'cachedir' => '',
        'char_set' => 'utf8',
        'dbcollat' => 'utf8_general_ci',
        'swap_pre' => '',
        'encrypt' => FALSE,
        'compress' => FALSE,
        'stricton' => FALSE,
        'failover' => array(),
        'save_queries' => TRUE
);
```

Для встановлення потрібен Composer, тому що Laravel використовує Composer для управління залежностями. Openserver має необхідний функціонал, тому просто відкриємо термінал і напишемо команду:

composer global require "laravel/installer"

Обов’язково розмістимо каталог ~ / .composer / vendor / bin у своєму PATH, щоб виконуваний файл laravel міг знаходитись у системі.

Після встановлення проста нова команда laravel створить новий проект Laravel у вказаному каталозі. Наприклад, новий блог laravel створить каталог із назвою blog, що містить свіжу інсталяцію Laravel з усіма вже встановленими залежностями Laravel.

Команда для встановлення:

composer create-project laravel/laravel blog "6.1.*"

Файл з налаштуванням бази даних знаходиться в папці config.

3.2 Генерація каркаса коду за допомогою кодогенератору GII

В YII2 використаємо зручний генератор коду Gii.

Після того як основа додатка створена і налаштована, можна приступити до розробки контролерів і моделей. Починаючи з версії 1.1.2, фреймворк доступний разом з вбудованим генератором коду Gii. Gii дозволяє генерувати моделі на основі таблиць в базі даних, а так само CRUD-контролери для основних дій з управління записами, такими як додавання запису (Create), побачити список записів, перегляд запису (Read),

редагування записи (Update) і видалення (Delete). Для активації Gii в файл конфігурації програми було додано опис підключення модуля gii:

```
'Gii' => array (
        'Class' => 'system.gii.GiiModule',
        // Пароль для входу в кодогенератор
        'password '=>' generate ',
        // Дозволені ip-адреси
        'ipFilters '=> array (' 127.0.0.1 ',' :: 1 '),
);
```

Після цього, для того щоб зайти в Gii необхідно перейти за наступним посиланням в браузері: http://YII2.local/index.php?r=gii/. Після авторизації буде доступний простий, зручний інтерфейс для генерації коду (рис. 3.1).

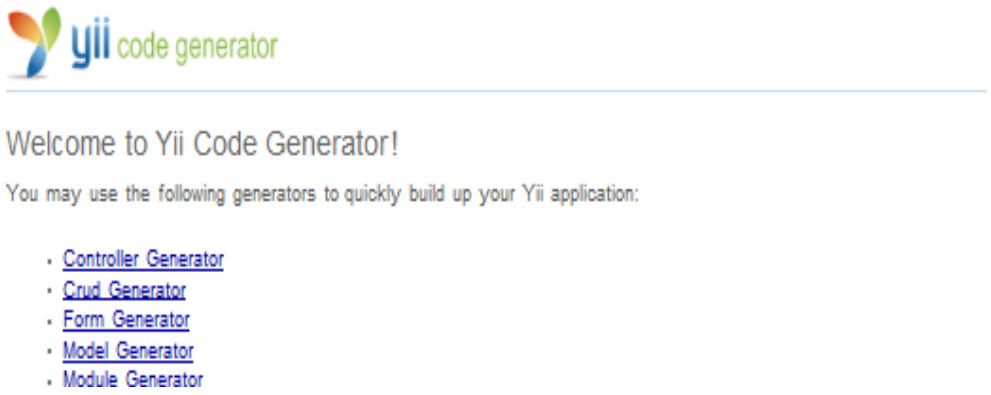

Рисунок 3.1 – Генератор коду GII

Цей генератор використовується для зручності і швидкості створення нових контролерів, але створювати нові файли можна і в ручну.

В інших фреймворках laravel і codeigneter дані будуть створені на основі вже наявних в них контролерів. Наприклад в codeigneter всі контролери знаходяться за маршрутом application\controllers.

Код контролеру головної сторінки:

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Welcome extends CI_Controller {
	public function __construct() {
		parent::__construct();
		$this->load->model('products_model');
	}
	public function index()
	{
		$data['goods'] = $this->products_model->getProducts();

		$this->load->view('templates/header', $data);
		$this->load->view('index', $data);
		$this->load->view('templates/footer');
	}
}
```

3.3 Встановлення верстки та доопрацювання каркаса

Подання в фреймворку YII – це спеціальний php-скрипт, що складається з простих виразів, що відповідає за відображення елементів призначеного для користувача інтерфейсу. У кожного уявлення є ім'я, яке використовується для пошуку скрипта уявлення. Якщо ім'я уявлення test, то ім'я скрипта уявлення test.php.

Для відображення уявлення використовується метод CController::render, якому передається ім'я уявлення і масив значень, який перетворюється в локальні змінні всередині уявлення. Наприклад, якщо всередині дії update контролера ItemController викликається:

```
$ This-> render ( 'update', array (
'Model' => $ model,
'Language' => YII :: app () -> language,
));
```

Скрипт уявлення буде розташований за наступним шляхом: protected/views/item/update.php, а всередині нього будуть доступні дві локальні змінні: $model і $language.

Всередині скрипта уявлення так само доступний екземпляр контролера, до якого можна звернутися за допомогою $this. Це дозволяє використовувати всередині уявлення властивості і методи контролера, наприклад, для відображення віджета, кешування, виведення заголовка сторінки і т. д.

Так само, при обробці скрипта уявлення неявно використовується спеціальне подання – макет, який декорує призначений для користувача інтерфейс і може містити такі елементи як шапка сайту і підвал. Усередині макета доступна локальна змінна $content, в якій зберігається висновок скрипта уявлення. Макет, по-замовчуванню, розташований за наступним шляхом: protected/views/layouts/main.php.

Віджет – це компонент, вбудований в уявлення, з метою відображення складної, самостійної частини інтерфейсу. Наприклад, віджет може використовуватися для виведення меню, таблиці і т. д.

Крім віджетів, в фреймворку присутній клас CHtml, статичні методи якого дозволяють виводити посилання на основі заданих маршрутів (route), форми і її елементів введення і т. д. Найбільш часто використовуваний метод

це Chtml :: link (<заголовок>, <маршрут в вигляді масиву або рядки>), який дозволяє вивести посилання на основі правил маршрутизації.

Всі дії по встановленню верстки та доопрацюванні каркаси можна розділити на наступні пункти:

1. налаштування макета;
2. налаштування уявлень видимих для відвідувачів;
3. доробка правил валідації моделей і провайдерів даних;
4. налаштування уявлень адміністратора.

При розробці проекту основною мовою був англійський, всі інші мови були додані за допомогою файлів перекладів і будуть описані пізніше в роботі.

Подання в фреймворку laravel зазвичай містять HTML-код додатків і представляють собою зручну можливість розділення бізнес-логіки та логіки відображення інформації. Шаблони зберігаються в папці resources/views.

Простий шаблон виглядає приблизно так:

```
<!-- resources/views/greeting.php -->
<html>
<body>
<h1> Привіт, <? php echo $ name; ?> </h1>
</body>
</html>
```

Можна сформувати шаблон у відповідь приблизно так:

```
Route::get('/', function()
{
        return view('greeting', ['name' => 'James']);
});
```

Як можна побачити, перший аргумент хелпера view – ім'я файлу шаблону в папках resources/views, а другий – дані, які будуть передані у відображенні.

Звичайно, ви можете робити піддиректорії в resources/views. Наприклад, якщо ваш шаблон знаходиться в файлі resources/views/admin/profile.php, то виклик хелпери буде виглядати так:

```
return view('admin.profile', $data);
```

Або так:

```
return view('admin/profile', $data);
```

Наступні приклади роблять доступним в шаблоні змінну $name і привласнюють їй значення 'Victoria':

```
// використовуючи with ()
$ View = view ( 'greeting') -> with ( 'name', 'Victoria');
// використовуючи "магічний" метод
$ View = view ( 'greeting') -> withName ( 'Victoria');
// за допомогою другого параметра хелпери
$ View = view ( 'greetings', [ 'name' => 'Victoria']);
```

Іноді потрібно передати дані в усі шаблони. Можна зробити це кількома шляхами: за допомогою хелпера view(), використовуючи contract або за допомогою загального шаблону (wildcard) композер.

За допомогою хелпери це можна зробити ось так:

```
view () -> share ( 'data', [1, 2, 3]);
```

Або, якщо ви хочете використовувати фасад, як в Laravel 4.x:

```
View :: share ( 'data', [1, 2, 3]);
```

Цей код можна покласти в метод boot() сервіс-провайдера – або загального сервіс-провайдера додатки AppServiceProvider або свого власного.

Коли хелпери view() викликається без аргументів, він повертає імплементацію контракту Illuminate\Contracts\View\Factory

Щоб визначити, чи є заданий файл шаблону на диску, використовуйте метод exist():

```
if (view () -> exists ( 'emails.customer'))
 {
 //
 }
```

Можна взяти файл шаблону по повному шляху до нього в файлової системі:

```
return view () -> file ($ pathToFile, $ data);
```

Відображення в Codeigneter це веб-сторінки, або фрагменти сторінок, такі як шапка, футер, колонка і т.д. Фактично відображення можуть бути гнучко об'єднані з іншими видами, якщо вам потрібен такий тип ієрархії.

Відображення ніколи не викликаються безпосередньо, вони повинні бути завантажені контролером.

Використовуючи текстовий редактор, створемо файл, з назвою view.php і запишемо в нього:

<Html>
<Head>
<Title> My store</ title>
</ Head>
<Body>
<H1> Welcome to my Store! </ H1>
</ Body>
</ Html>

Потім збережемо файл в директорію application/views/.

Щоб завантажити конкретний файл відображення, потрібно використати наступну функцію:

$ This-> load-> view ('name');

Де name це ім'я файлу відображення, розширення .php можна не вказувати, так як при бажанні ви можете використовувати інше.

Тепер відкрити ваш файл контролера і підключити вид:

<? Php
class Blog extends CI_Controller {

function index ()
{
$ This-> load-> view ('blogview'); }} ?>

3.3.1 Налаштування макета

Встановлення верстки була розпочата з макета. Для кожного фреймворка архітектура файлів своя, але певний принцип створення файлів макету зберігається для кожного з них.

Для Codeigneter файли макету були розміщені відразу в корені сайту в папці Assets.

YII2 містить в собі папку web а laravel папку public які також знаходяться відразу в корнеі сайту. В цих папках відповідно були розміщені файли шаблону.

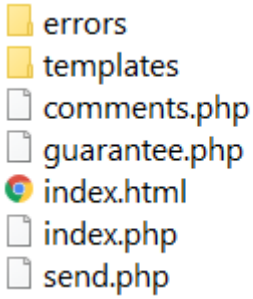

Рисунок 3.2 – Файлова структура шаблонів Codeigneter

Файлова структура шаблону схожа, вона має розділення на «футер», «хедер» і основну частину контенту для всіх трьох фреймворків. Основною відмінністю Codeigneter являється розміщення відповідних файлів сторінок не в відповідно названих папках. В YII2 наприклад вид comments буде розміщено в так само названій папці.

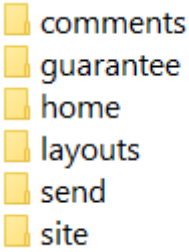

Рисунок 3.3 – Файлова структура шаблонів YII2

3.3.2 Налаштування уявлень для відвідувачів

Головна сторінка сайту на фреймворкі YII2 реалізована наступним методом. Ми звертаємось до моделі таблиці бази даних і використовуємо метод для отримання всіх наявних записів. Коли дані були отримані ми передамо їх у вид де вони будуть відображені відповідно до нашої верстки:

```
<?php

namespace app\controllers;
use YII\web\Controller;
use app\models\Products;

class HomeController extends Controller
{

   public function actionIndex()
   {
            $goods = Products::find()->all();
            return $this->render('index', compact('goods'));
   }
}
```

Принцип роботи у всіх трьох фреймворках однаковий, але код відрізняється. Наприклад в Codeigneter завантаження моделі відбувається в магічному методі construct:

```
<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Welcome extends CI_Controller {

public function __construct() {
        parent::__construct();
        $this->load->model('products_model');
}

public function index()
{
        $data['goods'] = $this->products_model->getProducts();

        $this->load->view('templates/header', $data);
        $this->load->view('index', $data);
        $this->load->view('templates/footer');
}}
```

3.3.3 Налаштування уявлень адміністратора

Для виведення посилань для адміністраторських дій необхідно перевірка чи є користувач адміністратором. Так як адміністратор всього один в системі і поділ прав не використовується, то достатньо лише перевірки на авторизовані користувача.

Laravel оснащений двома контролерами аутентифікації «з коробки». Вони знаходяться в просторі імен App\Http\Controllers\Auth. AuthController

обробляє реєстрацію та аутентифікацію користувачів, а PasswordController містить логіку для скидання паролів існуючих користувачів. Кожен з цих контролерів використовує трейти для включення необхідних методів. Для багатьох додатків вам не знадобиться редагувати ці контролери.

За замовчуванням ніякі роути не ведуть до контролерів аутентифікації. Потрібно самостійно додати їх у файлі app/Http/routes.php:

```
// Authentication routes ...
Route :: get ( 'auth / login', 'Auth \ AuthController @ getLogin');
Route :: post ( 'auth / login', 'Auth \ AuthController @ postLogin');
Route :: get ( 'auth / logout', 'Auth \ AuthController @ getLogout');
// Registration routes ...
Route :: get ( 'auth / register', 'Auth \ AuthController @ getRegister');
Route :: post ( 'auth / register', 'Auth \ AuthController @ postRegister');
```

Хоча контролери аутентифікації включені в фреймворк, всеодно доведеться створити шаблони, які ці контролери будуть відображати. Вони повинні знаходитися в каталозі resources/views/auth. Шаблон сторінки входу розташований в resources/views/auth/login.blade.php

Для YII2 всі методи реєстрації та аутентифікації адміністратора наявні з коробки у контролері по замовчуванню. Для роботи потрібно лише перевіряти чи виконав користувач авторизацію на сайті, і після цього давати доступ до функціоналу адміністративної частини.

```
<?php

namespace app\controllers;

use YII;
```

```
use YII\filters\AccessControl;
use YII\web\Controller;
use YII\web\Response;
use YII\filters\VerbFilter;
use app\models\LoginForm;
use app\models\ContactForm;

class SiteController extends Controller
{

    public function actionIndex()
    {
        return $this->render('index');
    }

    /**
     * Login action.
     *
     * @return Response|string
     */
    public function actionLogin()
    {
        if (!YII::$app->user->isGuest) {
            return $this->goHome();
        }

        $model = new LoginForm();
        if ($model->load(YII::$app->request->post()) && $model->login()) {
```

```
            return $this->goBack();
        }

        $model->password = '';
        return $this->render('login', [
            'model' => $model,
        ]);
    }

    /**
     * Logout action.
     *
     * @return Response
     */
    public function actionLogout()
    {
        YII::$app->user->logout();

        return $this->goHome();
    }
```

3.4 Налаштування кешування, багатомовності та «красивих» посилань

Після того як всі основні дії з доопрацювання каркаса виконані, залишилося додати підтримку великих навантажень (кешування), налаштувати багатомовність і красиві посилання.

Для активації кешування в файлі конфігурації було додано опис компонента додатка cache з класом CFileCache, що відповідає за зберігання кешу в локальних файлах.

Для налаштування кешування в контролерах використовувався компонент COutputCache, описаний в методі filters контролера:

```
array (
// Компонент відключений для адміністраторських дій
'COutputCache -create, update, upload',
// Термін придатності
'Duration' => 24 * 3600 * 365,
// залежності
'Dependency' => array (
'Class' => 'CChainedCacheDependency',
'Dependencies' => array (
// по глобальній зміні pm_time
new CGlobalStateCacheDependency ( 'pm_time'),
// по глобальній зміні cu_time
new CGlobalStateCacheDependency ( 'cu_time'),
),
),
// дані в кеші відрізняються по категорії, сортування, сторінок, мови
'VaryByParam' => array ( 'category', 'sort', 'page', 'lang'),
// і по статусу авторизованого користувача
'VaryByExpression' => 'YII :: app () -> user-> isGuest',
// кеш використовується тільки для GET-запитів
'RequestTypes' => array ( 'GET'),
),
```

де cu_time – поновлення категорій, а pu_time – товарів і змінюються за подією збереження моделі.

Так як, по-замовчуванню, додаток вже підтримує багатомовність за допомогою класу CPhpMessageSource, який зберігає перекази в php-файлах перекладу, то в проекті використовувався саме цей спосіб зберігання.

Кожне повідомлення перекладу відноситься до будь-якої категорії. Повідомлення перекладів зберігаються за наступним шляхом: protected / messages / <код мови> / <ім'я категорії> .php. Файли переклади містять в собі асоціативний масив, де ключем є фраза на вихідному (англійська) мовою, а значення – перекладене для даного коду (російська, наприклад).

Для перекладу повідомлення, фреймфорк YII надає статичний метод YII :: t («ім'я категорії», «повідомлення»), а для зберігання поточної мови на засланні в файл конфігурації було підключено стороннє розширення як компонент додатка urlManager: LangUrlManager.

Для активації «красивих», человекопонятний посилань компонент додатка urlManager був налаштований таким чином:

```
'UrlManager' => array (
'Class' => 'application.extensions.urlManager.CLangUrlManager',
// формувати посилання в форматі / route / p1 / v1 / p2 / v2 / ...
'UrlFormat' => 'path',
'Rules' => array (
// код мови завжди розташований попереду посилання
'<Lang: (en | ru | ro)>' => '/ price',
'<Lang: (en | ru | ro)> / <_ c>' => '<_ c>',
'<Lang: (en | ru | ro)> / <_ c> / <_ a>' => '<_ c> / <_ a>',
'<Lang: (en | ru | ro)> / text / index / <page:. *>' => 'Text / index',
),
```

```
// не виводити index.php на засланні
'ShowScriptName' => false,
)
```

В кореневій папці сайту (www) був доданий файл .htaccess з активацією і налаштуванням mod_rewrite в web-сервері apache:

```
Options + FollowSymLinks
IndexIgnore * / *
RewriteEngine on
RewriteCond% {REQUEST_FILENAME}! -F
RewriteCond% {REQUEST_FILENAME}! -D
RewriteRule. index.php
```

Laravel надає виразний, універсальний API для різних систем кешування. Налаштування кешу знаходяться в файлі config/cache.php. Тут можна вказати драйвер, який використовується за умовчанням в додатку. Laravel спочатку підтримує багато популярних системи, такі як Memcached і Redis.

Цей файл також містить безліч інших параметрів, які в ньому ж документовані. За замовчуванням, Laravel налаштований для використання драйвера file, який зберігає серіалізовані об'єкти кеша в файлової системі.

Контракти Factory і Repository надають доступ до служб кеша Laravel. Контракт Factory надає доступ до всіх драйверів кешу, певним для вашого застосування. А контракт Repository зазвичай є реалізацією драйвера кешу за замовчуванням для вашої програми, який заданий в файлі налаштувань cache.

В нашому випадку використаємо фасад Cache, який будемо використовувати для роботи з кешем. Фасад Cache забезпечує зручний і

лаконічний спосіб доступу до лежачих в його основі реалізацій контрактів кеша Laravel:

```
<? Php
  namespace App \ Http \ Controllers;

  use Illuminate \ Support \ Facades \ Cache;
  // use Cache;

  class UserController extends Controller
  {
   / **
    * Виведення списку всіх користувачів програми.
    *
    * @Return Response
    * /
   public function index ()
   {
    $ Value = Cache :: get ( 'key');

     //
   }
}
```

CodeIgniter має обгортки навколо найпопулярніших форм динамічного кешування і принцип їх використання дуже схожий з Laravel та YII2:

```
$this->load->driver('cache', array('adapter' => 'apc', 'backup' => 'file'));
```

```
if ( ! $foo = $this->cache->get('foo'))
{
        echo 'Saving to the cache!<br />';
        $foo = 'foobarbaz!';
        // Save into the cache for 5 minutes
        $this->cache->save('foo', $foo, 300);
} echo $foo;
```

ВИСНОВКИ

При розробці веб-сайту акцентували увагу на нові та актуальні технології, які служили б гарною опорою при проектуванні та при запуску сайту. Завдяки адаптивним технологіям користувачі мають змогу працювати з веб-сайтом, не залежно від пристрою, результат роботи та швидкодія при цьому не втрачаються.

Під час розробки акцентували увагу:

- Формулювання основних завдань, вимог та суворе дотримування їх;
- Аналіз та вибір програмних інструментів, методів розробки;
- Створення та наповнення бази даних;
- Розробка програмного забезпечення;
- Забезпечення адаптивного дизайну;

В використаних фреймворках різний підхід до реалізації певних функцій, але їх об’єднує схожа архітектурна побудова, тому для всіх них можна виокремити переваги якими вони володіють:

1. Відмінна підтримка та гарна документація фреймворків. Для російськомовних користувачів не складно знайти документацію та навіть форуми присвячені даним фреймворкам;
2. Допомога від розробників фреймворків. Великий плюс їх підтримки це можливість задати питання розробникам на форумі;
3. Повна підтримка ООП. Фреймворки повністю заточений під п'яту версію php що дозволяє підтримувати весь функціонал при об'єктно орієнтованому програмуванні. Розробники не пішли на підтримку php4 на шкоду гнучкості і зручності ООП. У зв'язку з цим фреймворк не працюватиме на php4, але зате відмінно покаже себе на php5;
4. Захист. Всі стандартні класи заточені під високий рівень безпеки що при вмілому зверненні дозволяє повністю убезпечити свій сайт від Sql-Inj, XSS, CSRF і інших атак.

ПЕРЕЛІК ПОСИЛАНЬ

1. Аткінсон Л. PHP 5. Бібліотека професіонала / Леон Аткінсон, Зеєв Сураські. – 2006.

2. Бенкен О. PHP, MySQL, XML: програмування для Інтернету / Олена Бенкен. – С.-П. : БХВ-Петербург, 2015. – 200 с.

3. Веллінг, Люк. PHP and MySQL Web Development / Люк Веллінг, Лаура Томсон. – М. : Альфа-книга, 2017. – 768 с.

4. Дакетт, Джон. HTML і CSS. Розробка і дизайн веб-сайтів / Джон Дакетт. – М. : Ексмо, 2017. – 280 с.

5. Дронов В. PHP, MySQL і Dreamweaver. Розробка інтерактив-них Web-сайтів / Володимир Дронов. – М. : Вільямс, 2017. – 750 с.

6. Зандстра, Мет. PHP. Об'єкти, шаблони і методики програмування / Мет Зандстра. – М. : Вільямс, 2015. – 576 с.

7. Івашкевич В. Інтегрований облік в системі управління підприємства / Віталій Івашкевич. – М. : Вільямс, 2018. – 124 с.

8. Коггзолл Д. PHP 5. Повне керівництво / Джон Коггзол. – 2006.

9. Колісниченко Д. PHP 5/6 і MySQL 6. Розробка Web-додатків / Денис Колісниченко. – С.-П. : БХВ-Петербург, 2015. – 280 с.

10. Колісниченко Д. PHP і MySQL. Розробка Web-додатків / Денис Колісниченко. – С.-П. : БХВ-Петербург, 2017. – 640 с.

11. Котеров Д. PHP 7 / Дмитро Котеров, Ігор Сімдянов. – С.-П. : БХВ-Петербург, 2016. – 311 с.

12. Кузнєцов М. PHP 5 на прикладах / Максим Кузнєцов, Ігор Сімдянов, Сергій Голишев. – С.-П. : БХВ-Петербург, 2005.

13. Кузнєцов М. Самовчитель MySQL 5 / Максим Кузнєцов, Ігор Сімдянов. – М. : Аванта, 2017. – 750 с.

14. Кухарчік А. PHP: навчання на прикладах / А. Кухарчік. – Нове

знання, 2004. – 240 с.

15. Мазуркевич О. PHP: настільна книга програміста / Олександр Мазуркевич, Дмитро Єловий.

16. Макграт, Майк. PHP7 для початківців з покроковою інструкцією / Майк Макграт. – М. : Ексмо-Пресс, 2018. – 256 с.

17. Маклафлін, Бретт. PHP і MySQL. Вичерпне керівництво / Бретт Маклафлін. – С.-П. : Пітер, 2016. – 544 с.

18. Ніксон, Робін. Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS і HTML5 / Робін Ніксон. – М. : Аванта, 2017. – 768 с.

19. Оліщук А. В..Н. Розробка WEB додатків на PHP 5. Професійна робота / А. В. Оліщук, А. Н. Чаплигіна. – Вільямс, 2006. – 352с.

20. Орлов А. PHP: Корисні прийоми / Антон Орлов. – 2005.

21. Сідерхолм, Ден. Куленепробивний веб-дизайн. Бібліотека спеціаліста / Ден Сідерхолм. – М. : Вільямс, 2015. – 304 с.

22. Сімдянов І. MySQL 5 / Ігор Сімдянов, Максим Кузнєцов. – С.-П. : БХВ-Петербург, 2018. – 423 с.

23. Скляр, Девід. Вивчаємо PHP 7. Керівництво по створенню інтерактивних веб-сайтів / Девід Скляр. – М. : Вільямс, 2017. – 464 с.

24. Суерінг С. PHP і MySQL Біблія програміста. Діалектика / С. Суерінг, Т. Конверс, Д. Парк. – 2010. – 912 с.

25. Шлосснейгл Д. Професійне програмування на PHP / Джордж Шлосснейгл. – 2006.

26. Уїтні, Девід. Програмування. Вчимося створювати сайти, програми та ігри. HTML, CSS і JavaScript / Девід Уїтні. – С.-П. : Пітер, 2018. – 208 с.

27. Buraga S. Proiectarea siturilor Web. Design şi funcţionalitate / Sabin Buraga. – [ediţia a II-a]. – Bucureşti: Ed. Teora, 2005. – 344 p.

28. Verificarea stării şi a parametrilor de funcţionare MySQL // Lamp.

2011. Disponibil pe internet: http://www.lamp.ro/articole-tutoriale-optimizare-mysql/verificarea-starii-si-a-parametrilor-de-functionare-mysql/.

ДОДАТОК А

Лістинг контролеру HomeController (Laravel)

```
<?php

declare(strict_types = 1);

namespace App\Http\Controllers;

use App\Category;
use App\Comment;
use App\Http\Requests\Admin\Order\StoreRequest;
use App\Http\Requests\Client\Order\StoreRequest as ClientOrderStoreRequest;
use App\Notifications\CommentNotification;
use App\Order;
use App\Product;
use App\Setting;
use App\User;
use Butschster\Head\Packages\Entities\OpenGraphPackage;
use Illuminate\Contracts\View\View as ViewContract;
use Illuminate\Http\JsonResponse;
use Illuminate\Support\Arr;
use Illuminate\Support\Facades\Notification;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\View;
use Butschster\Head\Facades\Meta;

/**
 * Class HomeController
 *
 * @package App\Http\Controllers
 */
class HomeController extends Controller
{
    /**
     * @return \Illuminate\Contracts\View\View
```

```
     *
     * @throws \Illuminate\Contracts\Filesystem\FileNotFoundException
     */
    public function index(): ViewContract
    {
        $settings = Setting::latest('updated_at')->first() ?? null;
        $productByCategory = [];
        $products = Product::query()->where('is_hidden', false)->get() ?? [];
        $categories = Category::query()->where('is_hidden', false)->get() ?? [];

        foreach ($products as $product) {
            $productCategory = $product->category()->first();
            if ($productCategory->getAttribute('name') !== Category::ACCESSORIES) {
                $productByCategory[$productCategory->getAttribute('name')][] = [
                    'id' => $product->getKey(),
                    'title' => $product->getAttribute('title'),
                    'image' => $product->getAttribute('image'),
                    'price' => $product->getAttribute('price'),
                    'old_price' => $product->getAttribute('old_price'),
                    'general_info' => $product->getAttribute('general_info'),
                    'variations' => $product->getAttribute('variations'),
                ];
            }
        }

        $seoTitle = isset($settings) && isset($settings->getAttribute('general_settings')['seo_title'])
            ? $settings->getAttribute('general_settings')['seo_title']
            : '';
        $seoImage = isset($settings) && isset($settings->getAttribute('general_settings')['seo_image'])
            ? $settings->getAttribute('general_settings')['seo_image']
            : '';

        $og = new OpenGraphPackage('home_og');

        $og->setType('OG META TAGS')
            ->setSiteName($seoTitle)
            ->setTitle($seoTitle)
```

```
            ->addImage($seoImage);

        $og->toHtml();

        Meta::registerPackage($og);

        Meta::prependTitle($seoTitle)
            ->setKeywords(isset($settings) ? $settings->getAttribute('general_settings')['seo_keywords'] : '')
            ->setDescription($seoTitle);

        return View::make('home', [
            'settings' => $settings ?? [],
            'products' => $productByCategory,
            'categories' => $categories,
            'content' => json_decode(Storage::disk('file')->get('content.json'), true)
        ]);
    }

    /**
     * @return \Illuminate\Contracts\View\View
     */
    public function comments(): ViewContract
    {
        return View::make('comments', [
            'categories' => Category::query()->where('is_hidden', false)->get() ?? [],
            'settings' => Setting::latest('updated_at')->first() ?? null,
            'comments' => Comment::query()->where('is_hidden', false)->paginate(5) ?? [],
        ]);
    }
    /**
     * @return \Illuminate\Contracts\View\View
     */
    public function guarantee(): ViewContract
    {
        return View::make('guarantee', [
            'categories' => Category::query()->where('is_hidden', false)->get() ?? [],
            'settings' => Setting::latest('updated_at')->first() ?? null,
        ]);
```

```
}

/**
 * @param \App\Http\Requests\Admin\Order\StoreRequest $request
 *
 * @return \Illuminate\Http\JsonResponse
 */
public function makeOrder(StoreRequest $request): JsonResponse
{
    $orderData = array_merge($request->all(), ['ip_address' => $request->ip()]);

    Order::create($orderData);

    return $this->json()->noContent();
}
/**
 * @param \App\Http\Requests\Client\Order\StoreRequest $request
 *
 * @return \Illuminate\Http\JsonResponse
 */
public function callMe(ClientOrderStoreRequest $request): JsonResponse
{
    $orderData = $request->except(
            [
                'ordered_product',
            ]
        ) + [
            'ordered_product' => $request->get('ordered_product') ?? [],
        ];

    Order::create(array_merge($orderData, ['ip_address' => $request->ip()]));

    return $this->json()->noContent();
}

/**
 * @param \App\Http\Requests\Admin\Comment\StoreRequest $request
 *
 * @return \Illuminate\Http\JsonResponse
 */
```

```
   public function addComment(\App\Http\Requests\Admin\Comment\StoreRequest $request): JsonResponse
   {
      $comment = Comment::create(array_merge($request->all(), ['is_hidden' => true]));

      Notification::send(
         User::query()->scopes(['notifiableUsers'])->get(),
         new CommentNotification($comment->getKey())
      );

      return $this->json()->noContent();
   }
   /**
    * @param \App\Product $product
    *
    * @return \Illuminate\Contracts\View\View
    */
   public function product(Product $product): ViewContract
   {
      $og = new OpenGraphPackage('product_og');
      $og->setType('OG META TAGS')
         ->setTitle($product->getAttribute('seo')['title'] ?? '')
         ->addImage($product->getAttribute('seo')['image'] ?? '');
      $og->toHtml();

      Meta::registerPackage($og);

      Meta::prependTitle($product->getAttribute('seo')['title'] ?? '')
         ->setKeywords($product->getAttribute('seo')['keywords'] ?? '')
         ->setDescription($product->getAttribute('seo')['meta'] ?? '');

      $ids = Arr::pluck($product->getAttribute('recommended_products'), 'id');

      return View::make(
         'product',
         [
            'product' => $product,
            'categories' => Category::query()->where('is_hidden', false)->get() ?? [],
```

```
            'settings' => Setting::latest('updated_at')->first() ?? null,
            'recommended' => Product::query()->whereIn('id', $ids)->get(),
        ]
    );
  }
  /**
   * @return \Illuminate\Contracts\View\View
   */
  public function accessories(): ViewContract
  {
    $productByCategory = [];
    $accessories = Product::query()->where('is_hidden', false)->get() ?? [];

    foreach ($accessories as $product) {
      $productCategory = $product->category()->first();
      if ($productCategory->getAttribute('name') === Category::ACCESSORIES) {
        $productByCategory[$productCategory->getAttribute('name')][] = [
          'id' => $product->getKey(),
          'title' => $product->getAttribute('title'),
          'image' => $product->getAttribute('image'),
          'price' => $product->getAttribute('price'),
          'old_price' => $product->getAttribute('old_price'),
          'general_info' => $product->getAttribute('general_info'),
          'variations' => $product->getAttribute('variations'),
        ];
      }
    }

    return View::make(
      'accessories',
      [
        'products' => $productByCategory,
        'categories' => Category::query()->where('is_hidden', false)->get() ?? [],
        'settings' => Setting::latest('updated_at')->first() ?? null,
        'content' => json_decode(Storage::disk('file')->get('content.json'), true)
      ]
    );
  }
}
```

ДОДАТОК Б

Лістинг контролеру Welcome (Codeigneter)

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Welcome extends CI_Controller {


	public function __construct() {
		parent::__construct();
		$this->load->model('products_model');
	}

	public function index()
	{
		$data['goods'] = $this->products_model->getProducts();

		$this->load->view('templates/header', $data);
		$this->load->view('index', $data);
		$this->load->view('templates/footer');
	}
}
```

ДОДАТОК В

Лістинг виду Index контролеру Welcome (Codeigneter)

```
<section class="header-baner">
        <div class="header-baner-title">
         <p class="header-baner-title-head hidden-opacity">Оригинальная техника Apple со скидкой до 32%</p>
         <p class="header-baner-title-desc hidden-opacity animate-infScale">Гарантия лучшей цены. Оплата после получения товара. Доставка в день заказа. </p>
         <button class="know-more hidden-opacity">Товары со скидкой</button>
        </div>
        <div class="carousel slide" data-ride="carousel" id="carousel-example-generic">
         <!-- <ol class="carousel-indicators">
                                <li class="active" data-slide-to="0" data-target="#carousel-example-generic"></li>
                                <li data-slide-to="1" data-target="#carousel-example-generic"></li>
                                <li data-slide-to="2" data-target="#carousel-example-generic"></li>
                                 <li data-slide-to="3" data-target="#carousel-example-generic"></li>
                            </ol> -->
        <div class="carousel-inner">
              <div class="item active">
               <img width="100%" src="assets/img/fon.jpg">
               <div class="carousel-caption">
                    <p class="site-title">Оригинальная техника Apple со скидкой до 30%</p>
                    <p>
                     <button class="know-more hidden-opacity">Купить</button>
                    </p>
               </div>
              </div>
        </div>
```

```
        </div>
</section>

<section class="advantages container-fluid" id="press-about">
        <div class="container" id="advantages">
          <div class="row">
                <div class="col-xs-12">
                  <p class="press-about-title hidden-opacity">Преимущества</p>
                  <!--<p class="press-about-text">покупки в официальном дисконт-центре со статусом premium reseller</p>-->
                </div>
          </div>
          <div class="row advantages-row">
                <div class="col-md-4 col-xs-12 hidden-opacity"><img src="assets/img/advantages-1.png">
                  <p>Только оригинальная продукция<br>
                        со скидкой до 30%</p>
                </div>
                <div class="col-md-4 col-xs-12 hidden-opacity"><img src="assets/img/advantages-2.png">
                  <p>Гарантия 1 год</p>
                </div>
                <div class="col-md-4 col-xs-12 hidden-opacity"><img src="assets/img/advantages-3.png">
                  <p>Тест-драйв 30 дней</p>
                </div>
          </div>
          <div class="row advantages-row">
                <div class="col-md-4 col-xs-12 hidden-opacity"><img src="assets/img/advantages-4.png">
                  <p>Быстрая доставка</p>
                </div>
                <div class="col-md-4 col-xs-12 hidden-opacity"><img width="100px" src="assets/img/5i.jpg">
                  <p>5 лет успешной работы</p>
                </div>
                <div class="col-md-4 col-xs-12 hidden-opacity"><img width="100px" src="assets/img/pe.png">
                  <p>6000 счастливых покупателей</p>
                </div>
          </div>
```

```
        </div>
</section>


<div id="catalogue">
<?php foreach ($goods as $k => $v): ?>
        <section class="phone-container container-fliud product-<?=$v['tag'];?>" id="product-<?=$v['tag'];?>">
                <div class="container">
                  <div class="row">
                        <div class="col-md-5 hidden-opacity"></div>
                        <div class="col-md-7 hidden-opacity">
                          <div class="phone-container-title hidden-opacity" id="<?=$v['tag'];?>">
                                <span>
                                  <a> <?=$v['title'];?> </a>
                                </span>
                          </div>
                        </div>
                  </div>
                </div>

                <?php $arr = json_decode($v['variations']);?>

                <div class="carousel slide" data-interval="false" data-ride="carousel" id="">
                  <!-- Wrapper for slides -->
                  <div class="carousel-inner" role="listbox">
                        <div class="item active">
                          <div class="container">
                                <div class="row">
                                  <div class="col-md-5 phone-preview">

                                        <img class="hidden-opacity" src="<?=$v['image'];?>">
                                        <?php foreach ($arr->images as $img): ?>
                                                <img class="hidden-opacity visible full-visible" src="<?=$img;?>" style="display: none;">
                                        <?php endforeach ?>

                                        <div class="clear"></div>
```

```
                                <div class="clear"></div>

                                <div class="preview-buttons">
                                 <button class="delivery-kit fancybox-2 link_modal" href="#package-popup">Комплект поставки</button>
                                 <span style="display: none">assets/iphone-11promax/3.jpg</span>
                                 <button class="characteristics fancybox-2 hidden-xs" href="#characteristics-popup">Характеристики</button>
                                 <div style="display: none">
                                     <p class="text-center">
                                         <?=$v['general_info'];?>
                                     </p>
                                 </div>
                                </div>
                            </div>

                            <div class="col-md-7 hidden-opacity">
                                <form class="variations_form cart" data-product_id="95" data-product_variations="" enctype="multipart/form-data" method="post">
                                <div class="check-color">Выберите цвет:</div>
                                <div class="row col colorflex">
                                    <?php foreach ($arr->colors as $key => $color): ?>
                                        <div class="col-xs-2 col-md-2 col col-md-2 col-md-offset-0 col">
                                          <div class="color-div" data-number="<?=$key + 1?>" data-product="<?=$v['tag'];?>">
                                              <div class="color-img color-img-white" style="background: <?=$color;?>; "></div>
                                          </div>
                                        </div>
                                    <?php endforeach ?>
                                </div>

                                <div class="row price price_30">
                                    <span>Стоимость со скидкой:</span>
                                    <span class="price-phone">
                                      <div class="woocommerce-variation single_variation" style="">
```

```
                                        <div class="woocommerce-variation-
description"></div>
                                        <div class="woocommerce-variation-
price">
                                          <span class="price">
                                              <del>
                                                <!-- цена х -->
                                                <span class="amount
main_p">
                                                    <?=$v['old_price'];?>
грн.</span>
                                              </del>
                                              <ins>
                                                <span class="amount
sale_p"><?=$v['price'];?> грн.</span>
                                              </ins>
                                          </span>
                                        </div>
                                        <div class="woocommerce-variation-
availability"></div>
                                    </div>
                                </span>
                            </div>

                            <div class="row hr">
                                <p><img class="hr"
src="/assets/img/uploads/punkt.png">Магазин</p>
                                <p><img class="hr"
src="/assets/img/uploads/dost.png">Доставка за 2 часа</p>
                                <p><img class="hr"
src="/assets/img/uploads/orig.png">Оригинальная техника</p>
                                <p><img class="hr"
src="/assets/img/uploads/gar.png">Гарантия 1 год</p>
                                <p><img class="hr"
src="/assets/img/uploads/test.png">Тест драйв 30 дней</p>
                            </div>

                            <div class="single_variation_wrap">
                                <div class="woocommerce-variation-add-to-
cart variations_button buy-buttons hidden-opacity">
                                    <button class="buy-in-click buy-in-
```

```
click_act fancybox" href="#buy-in-click-popup">Заказать сейчас</button>
                                        </div>
                                    </div>

                                        </form>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </section>
<?php endforeach ?>
</div>


<section class="container-fluid delivery" id="akcii">
        <div class="container">
          <div class="row">
                <div id="present" class="module test2 presents-box" style="margin-top:  20px; background-color: # ; padding-bottom: 40px;">
                  <div class="container-12 relative">
                        <div id="header3" class="module-header">Защитное стекло и чехол от Apple в подарок!</div>
                        <div style=""></div>
                        <div style="float: left; width: 450px; font-size: 18px; font-weight: 400; padding-top: 40px; padding-left: 40px; visibility: visible; animation-duration: 1s; animation-name: slideInLeft;" class="wow slideInLeft moduleee" data-wow-duration="1s">
                          <div>
                                <div class="point1"></div> Отлично защищает от падений
                          </div>
                          <div style="margin-top: 20px;">
                                <div class="point1"></div> Разные цвета на выбор
                          </div>
                          <div style="margin-top: 20px;">
                                <div class="point1"></div> В наличии чехлы для всех моделей iPhone
                          </div>
                        </div>
```

```
                    <div style="float: left; text-align: center; width: 450px; margin-top: 0px; margin-bottom: 0px; visibility: visible; animation-duration: 1s; animation-name: slideInRight;" class="wow slideInRight moduleeeim" data-wow-duration="1s">
                      <img class="primggg" src="/assets/img/present.jpg">
                    </div>
                  </div>
                </div>
              </div>
            </div>
</section>


<div id="warranty" class="module test2 warranty-box" style="margin-top: 20px; background-color: #fafafa; padding-bottom: 0px;">
        <div class="container-12 relative">
          <div id="header3" class="module-header">Гарантия - 1 год</div>
          <div style="float: left; width: 960px; padding-top: 40px;" class="newsec2">
                <div style="float: left; width: 480px; visibility: visible; animation-name: slideInLeft;" class="wow slideInLeft newsec2-1">
                  <div style="margin-top: 20px; margin-left: 10px; font-size: 23px;" id="test2-text2" class="test2-text2">Закон о защите прав потребителей</div>
                  <div id="test2-text4" class="test2-text4" contenteditable="false" style="line-height: 23px; font-size: 16px; margin-left: 10px;">
                    Если за 14 дней Вы обнаружите какую-либо неисправность, мы любезно вернем Вам деньги без скандалов и головной боли. Единственное требование - сохранение товарного вида.
                  </div>
                </div>
                <div style="float: left; width: 480px; visibility: visible; animation-name: slideInRight;" class="wow slideInRight newsec2-1">
                  <div id="test2-text3" style="margin-top: 20px; margin-left: 10px; font-size: 23px;" class="test2-text3">Расширенная гарантия – 1 год</div>
                  <div id="test2-text5" class="test2-text4" contenteditable="false" style="line-height: 23px; font-size: 16px; margin-left: 10px; padding-bottom: 40px">
                    <div>Если в течение 1 года смартфон сломается - специалисты
                      <div> отремонтируют его бесплатно. Вы просто приезжаете</div> с телефоном и гарантийным талоном в сервисный центр.
                    </div>
```

```
            </div>
          </div>
        </div>
        <div class="clear"></div>
      </div>
</div>


<section class="container-fluid delivery" id="delivery">
      <div class="container">
        <div class="row">
          <div class="col-md-8 col-md-offset-2 col-xs-12">
            <p class="press-about-title hidden-opacity">Доставка и оплата</p>
          </div>
          <div class="row delivery-icons advantages-row">
            <div class="col-md-3 col-xs-12 hidden-opacity"><img
src="/assets/img/dost.png">
              <p>Доставка за 2 часа!*
              </p>
            </div>
            <div class="col-md-3 col-xs-12 hidden-opacity"><img
src="/assets/img/rub.png">
              <p>Оплата при получении товара</p>
            </div>
            <div class="col-md-3 col-xs-12 hidden-opacity"><img
src="/assets/img/kass.png">
              <p>Кассовый чек**</p>
            </div>
            <div class="col-md-3 col-xs-12 hidden-opacity"><img
src="/assets/img/garant.png">
              <p>Гарантийный талон**</p>
            </div>
          </div>
          <div class="row delivery-desc hidden-opacity">*Доставка
осуществляется нашими курьерами в течении 2-3х часов с момента
оформления заказа.<br>
            Стоимость доставки   уточняйте у оператора. <br>
            Доставка в регионы осуществлется только после 100%
предоплаты за товар. <br>
            **Предоставляется при покупке.</div>
        </div>
```

```
        </div>
</section>


<section class="container-fluid contacts" id="contacts">
        <address>
                <p class="press-about-title hidden-opacity">Контакты</p>

                <p class="press-about-text-2 hidden-opacity">г. Львов, ул. Генерала Чупринки 10.<br>
                  e-mail: imobi.sale.shop@gmail.com <br>
                  <a href="tel:38(067) 796-75-97">
                        <span class="lptracker_phone">38 (067) 796-75-97</span>
                  </a>
                </p>
                <p        class="press-about-text-2        pickup">Прием        заказов: круглосуточно <br>
                  Режим работы магазина: с 10:00 - 18:00 <br>
                  Режим работы доставки: с 10:00 - 16:00 <br>
                </p>
        </address>
</section>
```