МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ «ОРЕНДА АВТОМОБІЛІВ» НА БАЗІ PHP ФРЕЙМВОРКУ LARAVEL»**

Виконав: студент 2 курсу, групи 8.1218
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
С. В. Поляков
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, к.ф.-м.н. Кудін О. В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Матвійшина Н. В.
(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 програмна інженерія

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, доцент, к.ф.-м.н.

_____ Лісняк А.О.
(підпис)

« 29 » травня 2019 р

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Полякову Сергію Володимировичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Розробка інформаційної системи «Оренда автомобілів» на базі PHP фреймворку Laravel

керівник роботи Кудін Олексій Володимирович, к.ф.-м.н.
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 29 » травня 2019 року № 811-с

2. Строк подання студентом роботи 26 грудня 2019 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналітичний огляд технологій
2. Проектування системи
3. Приклад роботи. Тестування

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 30.05.2019

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	10.09.2019	
2.	Збір вихідних даних.	18.09.2019	
3.	Обробка методичних та теоретичних	25.09.2019	
	джерел.		
4.	Розробка першого та другого розділу.	17.10.2019	
5.	Розробка третього розділу.	26.11.2019	
6.	Оформлення та нормоконтроль	27.12.2019	
	кваліфікаційної роботи.		
7.	Захист кваліфікаційної роботи.	15.01.2020	

Студент _____ С.В.Поляков
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В.Кудін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

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

РЕФЕРАТ

Кваліфікаційна робота магістра: «Розробка інформаційної системи «Оренда автомобілів» на базі PHP фреймворку Laravel» 47 с., 13 рис., 2 табл., 11 джерел.

LARAVEL, FRAMEWORK, PHP, MVC, CMS, ВЕБ-САЙТ, ІНФОРМАЦІЙНА СИСТЕМА.

Об’єкт дослідження – створення веб-сайту «Оренда автомобілів».

Мета роботи: розробити інформаційну систему «Оренда автомобілів» для можливості огляду та замовлення товарів.

Метод дослідження – порівняння, аналіз емпіричний, аналітичний.

У кваліфікаційній роботі було проведено дослідження актуальних технології створення веб-додатків, здійснено порівняння найбільш популярних PHP фреймворків.

На основі розглянутого матеріалу було реалізовано веб-сайт. Результатом виконання кваліфікаційної роботи є готова система. Можливості системи можуть бути доопрацьовані та використовуватися в електронному бізнесі.

SUMMARY

Master's qualification paper: «Development of Information System «Rent a Car» on the Basis of PHP Framework Laravel» 47 pages., 13 figures., 2 table, 11 referenses.

LARAVEL, FRAMEWORK, PHP, MVC, CMS, WEB SITE, INFORMATION SYSTEM.

The object of the study is to create a Car Rental website.

The aim of the study is develop the information system "Rent a car" for inspection and ordering of goods.

Method of research - comparison, analysis empirical, analytical.

In the qualification work was conducted research of current technologies of creation of web applications, comparisons of the most popular PHP frameworks were made.

Based on the material reviewed, a website was implemented. The result of qualification work is a ready-made system. System capabilities can be further refined and used in e-business.

ЗМІСТ

ВСТУП

Швидкість, з якою поширюється інформація сьогодні, є вражаючою. Інтернет став каталізатором до поширення інформації. В Інтернеті знаходяться мільйони веб-сайтів різного спрямування. Створення та підтримка веб-сайту є важкою працею.

Розробка веб-сайтів є перспективним напрямком в розвитку інформаційних технологій. Глобальна мережа Інтернет охоплює з кожним роком все більше і більше користувачів. Так, в 2015 році було проведено дослідження Міжнародним союзом електрозв'язку, яким було встановлено, що розмір інтернет- аудиторії на кінець 2015 року складе 3.2 млрд чоловік, що становить 43 % населення світу

Розвиток Інтернету нерозривно пов'язано з проектуванням сайтів. Масова поява сайтів спровокувала проблем у їх якості. Популярність створення веб-ресурсів сприяла розробці різних систем і програм, які спрощують процес написання сайту. Також вони допомагають підвищити ефективність роботи, а також дозволяють розробнику сфокусуватися над основною логікою програми. Такі технології, як PHP, Java, MySQL, Oracle та розроблені на їх основі фреймворки - це каркаси системи або підсистеми, що можуть включати допоміжні програми, мови сценаріїв - і все, що полегшує розробку й об'єднання різних компонентів.

Фреймворки за останній час набрали популярність, і стали базовою платформою для розробки web-додатків. Іншими словами можна сказати, що вони забезпечують основну структуру програми. Використання фреймворків, дозволяє економити велику кількість часу, зменшити навантаження на процес розробки, позбавляючи від проблеми повторюваного коду, і швидко створювати додатки.

Метою кваліфікаційної роботи є: розробка Веб-додатка для можливості перегляду та оренди автомобілів, попередньо ознайомившись з основними принципами розробки на Laravel.

Виходячи з поставленої мети, сформульовано наступні задачі:

– проаналізувати існуючі фреймворки;
– спроектувати майбутню систему;
– реалізувати та протестувати проект.

Структурно дана робота складається за трьох розділів.

Переший розділ присвячено огляду технологій та можливостей фреймворка Laravel, а також його прівняння з іншими існуючими фреймворками.

У другому розділі увагу акцентовано на проектуванні системи що розробляється. Для цього було використано переважно діаграмми та схеми.

Третій розділ є завершальним в роботі та дозволяє побачити як працює, та поводить себе при тестуванні розроблена система.

1 АНАЛІТИЧНИЙ ОГЛЯД ТЕХНОЛОГІЙ

1.1 Загальне поняття фреймворка

Фреймворк – це структура програмної системи, що полегшує розробку і об'єднання різних компонентів великого програмного проекту. На відміну від бібліотек, які об'єднують набір підпрограм близької функціональності, фреймворк містить у собі велику кількість різних за призначенням бібліотек. Широкого вжитку також набуло слово «каркас», а деякі автори використовують його в якості основного. Можна також говорити про каркасний підхід як про підхід до побудови програм, де будь – яка конфігурація програми будується з двох частин: перша (постійна частина) – каркас, в якому конфігурації не пов’язані між собою і містяться гнізда, в яких розміщується друга частина (змінна) – змінні модулі або точки розширення.

Фреймворк, як правило, містить тільки базові програмні модулі, а все специфічні для проекту компоненти реалізуються розробником на їх основі. Тим самим досягається не тільки висока швидкість розробки, а й велика продуктивність і надійність рішень.

Фреймворк служить основою для сайту, але зазвичай не містить в собі готових програмних модулів для реалізації конкретних бізнес-процесів. Висловлюючись технічною мовою, фреймворк – це більш низькорівневе рішення, ніж CMS. Розробники, при створенні сайту на фреймворку, створюють не тільки публічну частину сайту, але і проектують базу даних, розробляють алгоритми для модулів системи, а також створюють адміністративний інтерфейс для управління проектом. Необхідність серйозних витрат на програмування робить розробку більш дорогою, але і результат виходить більш індивідуальним.

Цей вид платформ використовують майже всі великі веб-проекти, на CMS побудована лише дуже мала частина дійсно серйозних проектів, а також

переважна більшість веб-додатків і веб-сервісів (коробкових рішень для унікальних бізнес-процесів просто не існує, а використання не дуже відповідних CMS в якості основи для кастомізації дуже ускладнює розробку).

Для того щоб відповісти на питання, що має входити до фреймворка , щоб останній розглядався як WEB – технологія, необхідно розглянути сучасні види фреймворків:

– фреймворки програмної системи;
– фреймворки додатків;
– фреймворки концептуальної моделі.

Фреймворк програмної системи – це каркас системи або підсистеми, що може включати допоміжні програми, мови сценаріїв – все, що полегшує розробку й об'єднання різних компонентів. Фреймворк відрізняється від поняття бібліотеки тим, що бібліотека може бути використана в програмному продукті просто як набір підпрограм з близькою функціональністю, не впливаючи на архітектуру програмного продукту і не накладаючи на неї ніяких обмежень. У той час як фреймворк диктує правила побудови архітектури додатку, задаючи на початковому етапі розробки поведінку за замовчуванням, каркас, який потрібно буде розширювати і змінювати відповідно до зазначених вимог. До цього виду фреймворків відносяться і фреймворки для WEB.

Фреймворк додатку має стандартну структуру. Зі зростанням необхідності у графічних інтерфейсах користувача з'явилася і необхідність у фреймворках додатків. З їх допомогою простіше розробляти засоби для автоматичного створення графічних інтерфейсів. Для створення фреймворку додатків використовують об'єктно – орієнтоване програмування. Перший подібний фреймворк написала компанія Apple для Macintosh. Спочатку він був створений за допомогою Паскаль, а з часом його було переформатовано у C++ [1].

Одною з головних переваг у використанні фреймворків є те, що фреймворк визначає уніфіковану структуру для побудованих на його базі

додатків. Тому додатки на фреймворках значно простіше супроводжувати і допрацьовувати, так як стандартизована структура організації компонентів зрозуміла всім розробникам на цій платформі і не потрібно довго розбиратися в архітектурі, щоб зрозуміти принцип роботи програми або знайти місце реалізації того чи іншого функціоналу.

1.2 Поняття CMS

CMS – це система управління контентом, набір скриптів для створення, редагування і управління контентом сайту. Прикладами CMS є WordPress, Joomla, PrestaShop та інші.

Якщо раніше більшість сайтів були статичними; і вимагали внесення правок в їх вміст вручну, то зараз динаміка розвитку проектів вимагає готовності швидко реагувати на зміни і впроваджувати їх з максимальною оперативністю. При цьому не всі користувачі хочуть або можуть собі дозволити звертатися до розробників, особливо якщо сайт вимагає постійної роботи над ним.

У свою чергу, системи управління контентом дозволяють користувачам, які не володіють навичками розробки сайтів і знаннями мов програмування, самостійно працювати над створенням і зміною сайту.

Суть роботи CMS укладена в схемі поділу вмісту сайту і його дизайну. Користувачеві надається можливість вибрати шаблон, або заготовку, в якій заздалегідь визначено оформлення сторінки, і залишається тільки заповнити її потрібною інформацією. Більшість систем управління вмістом грунтується на використанні візуального редактора (WYSIWYG – від англ. What You See Is What You Get – «що бачиш, то і отримаєш») – програми, що дозволяє за допомогою інтуїтивно зрозумілого інтерфейсу додавати або змінювати інформацію на сайті. Варто відзначити, що сайт не складається з сукупності сторінок як такої, а формується динамічно. Доданий контент зберігається в

базі даних, наприклад, MySQL, і використовується при генерації сторінки після отримання відповідного запиту з боку клієнта.

Як правило, CMS використовуються для таких сайтів:

– блог, форум (WordPress, phpBB, vBulletin);
– інтернет-магазин (Magento, OpenCart, osCommerce);
– соціальні мережі (InstantCMS, Social Engine);
– персональні сайти (WordPress, Monstra);
– корпоративні сайти (Joomla, Drupal);
– портали (DLE, Drupal).

Тим не менше, більшість CMS гнучко настроюється і можуть бути використані для розробки сайтів різної спрямованості. Наприклад, найбільш популярним і універсальним варіантом є WordPress, на якому можливо створити практично будь-який проект: від особистого сайту до інтернет-магазину [2].

Переваги CMS:

– просто і зручно використовувати;
– доступний широкий функціонал за рахунок доповнень, тем та розширень;
– сайт можна створити за короткий проміжок часу;
– наявність документації.

Недоліки:

– не підходять для нетипових завдань;
– популярні CMS уразливі;
– необхідно стежити за оновленням CMS і сумісністю версій доповнень;
– підвищене споживання ресурсів, особливо при використанні плагінів.

1.3 Аналіз сучасних фреймворків

Сьогодні існують сотні фреймворків, що реалізовані на різних мовах програмування, таких як: Java, Rubi, PHP, Phyton, та інші.

PHP фреймворки дозволяють:

– прискорити процес розробки веб-додатків
– допомагають писати простий і якісний код
– повторно використовувати код в проектах
– легко масштабувати проекти
– використовувати сучасні практики програмування
– легше тестувати програмний код
– забезпечити безпеку проекту

Далі буде представлено список найпопулярніших PHP фреймворків з їх перевагами та недоліками.

Yii – це безкоштовний об'єктно-орієнтований компонентний full-stack PHP фреймворк. В основі Yii лежить інший фреймворк – PRADO, написаний на ASP.NET і згодом перенесений на PHP.

Yii можна використовувати для розробки будь-якого виду веб-додатків. Завдяки своїй основі компонентів, архітектурі і складній підтримці кешування, фреймворк підходить для розробки об'ємних проектів, таких як портали, форуми, системи управління контентом (CMS), систем електронної комерції, RESTful веб-сервісів, тощо.

Yii реалізує для використання MVC (Model-View-Controller) архітектурний шаблон і сприяє організації коду на основі цього шаблону.

Yii є full-stack фреймворком, надаючи безліч перевірених і готових до використання функцій: конструктор запитів і ActiveRecord для реляційних і NoSQL баз даних, RESTful API, підтримку багаторівневого кешування і т.п.

Yii надзвичайно розширюваний фреймворк в якому можна замінити майже кожну частину коду і розробляти потрібні розширення [6].

Переваги:

– для російськомовних розробників великим плюсом фреймворка, є хороша документація, безліч статей з прикладами коду і співтовариство;

– Yii фреймворк підкріплений сильною командою розробників, а також великим співтовариством професіоналів які постійно сприяють розвитку Yii;

– вбудований механізм створення віджетів уявлення, наприклад, для розміщення на сайті різних блоків: останні пости, категорії, навігація, блоки реклами тощо;

– вбудована підтримка автоматичної валідації форм і виведення повідомлень про помилки на основі даних з моделей веб-додатків;

– механізм Active Record для побудови реляційної обробки запитів бази даних;

– безліч готових розширень на Github і їх установка через Composer;

– вбудовані віджети для відображення даних: DetailView, ListView, GridView ;

– вбудовані механізми для аутентифікації, авторизації, реєстрації користувачів;

– містить вбудовану і зручну debug панель.

Недоліки:

– слабка екосистема навколо фреймворка, кілька форумів з невеликою активністю і т.п;

– хоч фреймворк і дозволяє робити код простим, але далеко не елегантним. Якщо його синтаксис порівнювати з фреймворком Laravel, то він поступається;

– Yii відстає від мови, стандартів і інших фреймворків. Нові оновлення з дійсно корисними функціями виходять не дуже часто;

– не дуже гнучка система маршрутизації.

Symfony – це PHP фреймворк для швидкої розробки веб-додатків і виконання рутинних завдань веб-програмістів. Розробка і підтримка фреймворку спонсорується французькою компанією Sensio.

Symfony складається з набору не пов'язаних між собою компонентів, які можна використовувати повторно в проектах.

За допомогою Symfony було розроблено безліч великих проектів:

– систем управління контентом: Magento, Drupal, Opencart;

– сервіс соціальних закладок Delicious;

– французький відеохостинг Dailymotion.

В тому числі, Symfony вплинув на розробку фреймворка Laravel, де були задіяні його компоненти.

Symfony дозволяє встановлювати сторонні пакети, бібліотеки, компоненти і налаштовувати їх за допомогою конфігурації в форматах YAML, XML, PHP. Symfony не забезпечує компонент для роботи з базою даних, але забезпечує тісну інтеграцію з бібліотекою Doctrine.

Symfony надає функцію поштової програми на основі популярної бібліотеки Swift Mailer. Ця поштова програма підтримує відправку повідомлень з ваших власних поштових серверів, а також з використанням популярних поштових провайдерів, таких як Mandrill, SendGrid і Amazon SES.

Механізм інтернаціоналізації дозволяє встановити і здійснити переказ повідомлень веб-додатку на основі вибраної мови або країни.

Symfony пропонує систему логування помилок додатку, а також підключити бібліотеку логування Monolog [7].

Переваги:

– потужна екосистема навколо фреймворка, з хорошим співтовариством і безліччю розробників;

– хороша документація яка постійно оновлюється для всіх версій фреймворка;

– безліч різних компонентів для повторного використання;

– пропонує механізм функціональних і модульних тестів для знаходження помилок в веб-додатку;

– підходить для складних веб-проектов електронной комерції.

Недоліки:

– незважаючи на хорошу документацію, фреймворк є складним для вивчення.

Zend Framework – це об'єктно-орієнтований PHP фреймворк для розробки веб-додатків, розроблений компанією Zend. Даний фреймворк як правило найбільше використовують при розробці великих комерційних проектів.

Такі компанії, як Google, Microsoft, співпрацюють з Zend для надання інтерфейсів веб-сервісам.

Компанія Zend бере участь в розробці ядра мови програмування PHP.

В якості менеджера залежностей пакетів Zend Framework використовує Composer. Для тестування веб-додатків застосовується PHPUnit, а для безперервної інтеграції служба Travis CI.

Підтримка безлічі баз даних: MariaDB, MySQL, Oracle Database, IBM DB2, Microsoft SQL Server, PostgreSQL, SQLite і Informix [8].

Переваги:

– дуже добре підходить для розробки комерційних веб-додатків;

– об'єктно-орієнтований підхід до розробки;

– непов'язані компоненти для повторного використання в проектах;

Недоліки:

– не підходить для швидкої розробки проектів;

– для російськомовного сегмента розробників дуже мало корисної інформації.

CodeIgniter – це популярний PHP мікро-фреймворк з відкритим вихідним кодом, для розробки веб-систем і додатків.

У CodeIgniter компоненти завантажуються і процедури виконуються тільки за запитом, а не глобально. Система не робить ніяких припущень

щодо того, що може знадобитися крім мінімальних основних ресурсів, тому система за замовчуванням дуже легка.

Компоненти фреймворка слабо пов'язані між собою і не залежать один від одного. Чим менше компонентів залежить один від одного, тим більш гнучкою і багаторазової стає система.

Хоча CodeIgniter працює досить швидко, обсяг динамічної інформації, яка відображається на сторінках, буде прямо залежати від використовуваних ресурсів сервера, пам'яті і циклів обробки, які впливають на швидкість завантаження сторінок.

Тому CodeIgniter дозволяє кешувати сторінки для досягнення максимальної продуктивності, за допомогою вбудованого компонента кешування [9].

Перваги:

– відмінна документація і англомовне співтовариство;
– висока продуктивність фреймворка;
– невеликий розмір фреймворку;
– надає легкі і прості рішення для розробки;
– підходить для швидкої розробки невеликих сайтів і веб-додатків;
– структура фреймворка не вимагає строгих правил кодування;
– не вимагає складної настройки, майже нульова конфігурація;
– MVC-архітектура веб-додатку.

Недоліки:

– довгий застій у розвитку Codeigniter.

Laravel – це безкоштовний PHP фреймворк з відкритим вихідним кодом, для розробки веб-додатків за архітектурним шаблоном MVC.

Він був створений як альтернатива фреймворку Codeigniter, в якому було недостатньо корисних функцій для розробки. В якості основи Laravel виступають компоненти іншого фреймворка – Symfony.

Фреймворк Laravel дуже популярний серед західних розробників веб-додатків.

За допомогою менеджера пакетів Composer, фреймворк Laravel дозволяє легко встановлювати і підключати різні компоненти для використання в веб-додатку.

Реалізація шаблону ActiveRecord – Eloquent ORM, дозволяє встановити відносини між об'єктами бази даних веб-додатку і вибудовувати зручні запити для маніпуляції даними.

Механізм автозавантаження класів дозволяє не підключати вручну файли через include і запобігає завантаженню компонентів що не використовуються.

Зручна система міграцій допомагає спростити розгортання і оновлення веб-додатку.

При створенні програми можна використовувати Artisan – інтерфейс командного рядка для введення вбудованих команд, а також створення своїх власних.

У Laravel є багато корисних функцій, що дозволяють зробити процес розробки веб-додатків швидким, простим і якісним.

Переваги:

– велика кількість інформації та зрозуміла документація;

– навколо фреймворка створена потужна екосистема. Різні курси, конференції, навчальні матеріали дозволяють зібрати навколо фреймворка велику кількість розробників і спонсорів, які зацікавлені в розвитку інструменту і приймають в цьому участь;

– синтаксис API фреймворка досить простий і зрозумілий. Тут немає довгих і складних конструкцій, а тільки короткі і продумані назви функцій;

– Laravel містить зручний механізм обробки помилок і виключень;

– фреймворк включає в себе вбудовані механізми аутентифікації і авторизації користувачів, яку можна переналаштувати під свої потреби;

– Laravel надає чистий і простий API поверх популярної бібліотеки SwiftMailer з драйверами для SMTP, Mailgun, Amazon SES і sendmail, щоб

зробити відправку пошти через локальну або хмарну службу за вибором. У тому числі є механізм для побудови черг відправки пошти;

– використання Composer. Даний менеджер залежностей дозволяє простим і зручним способом довантажувати бібліотеки необхідні для роботи кожного конкретного проекту.

Його відмінною рисою є те, що він встановлює їх під кожен проект локально, що дуже зручно.

Недоліки:

– для розробників без знання англійської мови або його слабким знанням, до мінусів фреймворка можна віднести досить невелику кількість статей, прикладів коду, перекладів офіційної документації. Для тих, хто знає англійську на рівні читання технічної документації, цей мінус можна опустити.

Каталоги що забезпечуюють роботу Laravel:

– app тут, як ви здогадуєтеся, розташовується власне вашу програму. Нижче ми розглянемо вміст цього каталогу докладніше;

– bootstrap містить файли, які здійснюють початкову завантаження (bootstraping) фреймворка і налаштовують автозагрузку класів. В папці cache зберігаються різні згенеровані файли, необхідні для роботи фреймворка;

– config тут знаходяться конфігураційні файли програми;

– database каталог для файлів міграцій БД і "посіву" даних. Тут же можна зберігати файл бази даних SQLite;

– public є DocumentRoot домену вашої програми і містить статичні файли; css, js, зображення і т.п;

– resources тут знаходяться шаблони (Views), файли локалізації та, якщо такі є, робочі файли LESS, SASS і js-додатки на фреймворками типу ReactJS, AngularJS або Ember, які потім збираються зовнішнім інструментом в папку public;

– storage цей каталог повинен мати права для запису в нього ззовні і в ньому Laravel зберігає скомпільовані Blade-шаблони, файли сесії,

файловий кеш, інші згенеровані файли, потрібні для роботи, а так само логи роботи програми. Ці файли розташовуються в підпапках app, framework long;

– vendor в цей каталог Composer встановлює пакети, зазначені в composer.json.

В каталозі app знаходяться класи вашого Laravel-додатки. За замовчуванням, цей каталог має неймспейс Appі класи в ньому автозавантажувані згідно стандарту PRS-4 .

Усередині знаходяться декілька підкаталогів, таких як Console, Http Providers. Перші два каталогу, як випливає з назви, містять класи, надають API до вашого додатком по протоколам CLI (командний рядок) і HTTP (робота через браузер).

В Console знаходяться класи Artisan-команд, а в Http контролери, посередники (middlewares) і реквести (класи валідації призначеного для користувача введення).

Такий підхід повинен підштовхнути новачків до відходу від загальноприйнятого, але шкідливого підходу писати весь код в контролерах, і абстрагувати логіку додатки від методу звернення до нього.

Каталог Events містить класи подій . Події служать для оповіщення інших частин вашого застосування про те, що відбувається в конкретному екшені, забезпечуючи більшу гнучкість і слабку зв'язаність коду.

Каталог Exceptions містить обробники винятків, а так само тут слід розміщувати самі класи виключень, які використовуються в вашому додатку.

Каталог Jobs містить завдання, що виконуються в чергах.

Каталог Listener смістить класи оброблювачів подій . У них міститься логіка – що саме потрібно робити у відповідь на виникле подія. Так, подія User Registered з папки Events може викликати оброблювач SendWelcomeEmail, який знаходиться в папці Listeners

Каталог Policies містить класи політик, які займаються розмежуванням доступу додатки.

Service providers (сервіс-провайдери, дослівно – «постачальники послуг») займають центральне місце в архітектурі Laravel. Вони призначені для первинного завантаження (bootstraping) додатки.

Програма, а також сервіси самого фреймворка завантажуються через сервіс-провайдери. Головним чином це реєстрація деяких речей – таких як Біндінг в IoC-контейнер (фасадів і т.д.), слухачів подій (event listeners), фільтрів Payca (route filters) і самих Payca (routes).

Сервіс-провайдери – центральне місце для конфігурації вашого застосування. Якщо ви відкриєте файл config/app.php, ви побачите масив providers.

У ньому перераховані всі класи сервіс-провайдерів, які завантажуються при старті вашої програми (звичайно, крім тих, які є «відкладеними» (deferred), тобто завантажуються на вимогу іншого сервіс-провайдера). Можна і потрібно створювати свої власні сервіс-провайдери для завантаження і налаштування різних частин свого застосування.

Сервіс-провайдери повинні розширювати (extends) клас Illuminate\Support\ServiceProvider. Це абстрактний клас, який вимагає, щоб в успадковане класі був метод register().

У методі register() можна тільки реєструвати свої класи (bindings) в сервіс-контейнері. Слухачів подій (event listeners), Payca і фільтри Payca там реєструвати не можна.

За допомогою Artisan можна легко створити нового провайдера, використовуючи команду make:provider [10].

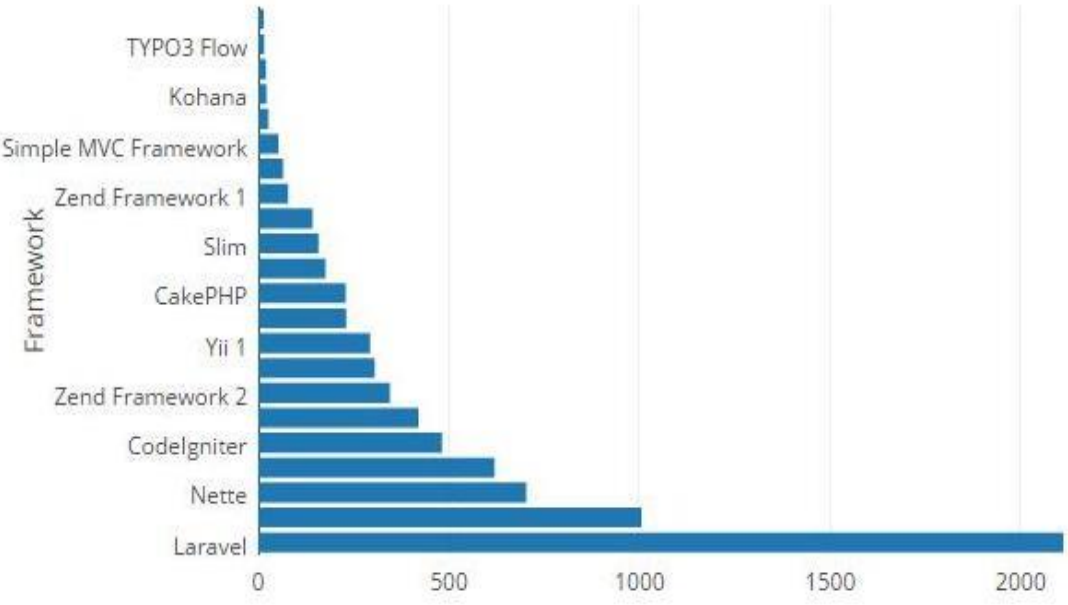

Рисунок 1.1 – Діаграма популярності фреймворків

Популярність Laravel порівняно з іншими фреймворками наочно зображена на діаграмі.

1.4 Критерії вибору фреймворку

Під час вибору фреймворку можуть виникнути певні труднощі, пов’язані з визначенням завдань, які він може виконувати, та його призначення. Не кожен фреймворк підтримує ORM для роботи з базами даних, має якісну інтеграцію і хорошу документацію. Якщо для створення сайту потрібно знайти зручний і простий в освоєнні фреймворк, то необхідно ретельно підійти до питання його вибору та зважити всі «за» і «проти».

Розглянемо загальні переваги використання фреймворку:

– гнучкість розробки та розвитку проекту;
– ефективне використання ресурсів сервера;
– відкритий код;

– легкість і надійність web-розробок. фреймворк складається з базових, перевірених, налагоджених функцій і операцій. Він побудований на базі об'єктно-орієнтованого програмування;

– постійний розвиток і вдосконалення фреймворка;

– великий обсяг супровідної документації, прикладів з розробки на різних мовах;

– світова популярність, велика кількість розробників;

– легкість супроводу проекту надалі, так як розробка із застосуванням фреймворка заснована на певних угодах;

– фреймворк дозволяє сконцентруватися на вирішенні архітектурних завдань, а не базових, як при розробці без його застосування;

– фреймворк дозволяє вузько вирішувати поставлену задачу.

Одна з головних переваг фреймворка – зручна розробка нестандартних проектів. Усі оригінальні проекти розробляють на фреймворках.

Web-проект, розроблений за допомогою фреймворку, розвивається динамічно. При зміні вимог змінюється і сайт, достатньо лише замінити окремий модуль, створити новий розділ або внести новизну в дизайні [5].

Недоліки застосування фреймворку досить умовні і незначні порівняно з перевагами.

1.5 Критерії раціональності використання фреймворку

Фреймворки використовуються в різних областях залежно від їх основних характеристик. Виділимо основні критерії раціональності використання фреймворку в таблиці 1.1.

Таблиця 1.1 – Критерії раціональності використання фреймворку

Назва	Характеристика
Підтримка баз даних	Питання підтримки баз даних в PHP фреймворк дуже важливе. Наприклад, Laravel підтримує MySQL, Postgres і SQLite, а фреймворк Kohana не підтримує Oracle і SQLite. Частина фреймворків мають вбудований ORM – шар, частина – ні.
Архітектура	Фреймворк також повинен використовувати MVC архітектуру. Більшість хороших PHP фреймворків мають бібліотеки, плагіни, модулі та розширення. Це дуже добре для того щоб реалізувати велике коло функціоналу, вдосконалити і прискорити процес розробки.
Швидкість розвитку	Цей пункт так само дуже важливий, оскільки деякі фреймворки оновлюються раз в кілька років (Codeigniter), а деякі раз на пару місяців. Це дозволяє уникнути використання старого, недопрацьованого коду при розробці.
Створення і перевірка форм	Наявність даного функціоналу в фреймворку полегшує створення полів для вводу тексту, логіну чи паролю до мінімума. А також додає потрібну майже в усіх проектах функцію валідації форм

Продовження таблиці

Назва	Характеристика
Підтримка баз даних	Питання підтримки баз даних в PHP фреймворк дуже важливе. Наприклад, Laravel підтримує MySQL, Postgres і SQLite, а фреймворк Kohana не підтримує Oracle і SQLite. Частина фреймворків мають вбудований ORM – шар, частина – ні.
Документація	Частина фреймворків мають слаборозвинуту або 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 для розуміння.
Поріг входження	Не всі фреймворки прості в освоєнні, це дуже важливо враховувати при виборі, так як на освоєння одного фреймворка може не вистачити року, а на освоєння іншого вистачить всього тижня.

Багато фреймворків відповідають зазначеним у таблиці характеристикам, але після аналізу стає зрозуміло що саме Laravel є найбільш раціональним рішенням для реалізації поставленої задачі.

1.6 Висновки до першого розділу

Фреймворки активно використовуються розробниками при створенні web-додатків із різним функціоналом і рівнем складності. Аналіз основних характеристик і можливостей сучасних фреймворків дозволяє вибрати оптимальний варіант для конкретних web-додатків з урахуванням

поставлених завдань. Вибір та використання фреймворків відіграє важливу роль при проектуванні, реалізації та супроводі як простих web-додатків, так і складних програмних комплексів.

Laravel на даний момент є найбільш перспективним PHP фреймворком і підходить для створення як невеликих, так і великих веб-проектів.

PHP фреймворки є незамінним інструментом для швидкого зведення фундаменту будь-якого сайту. Вони допоможуть прискорити процес розробки і зробити якісний проект, уникнувши програмування всіх частин з нуля.

2 ПРОЕКТУВАННЯ

2.1 Вимоги до сайту

При розробці системи було з’ясовано, що вона має відповідати наступним характеристикам:

– інтуїтивно зрозумілий інтерфейс адміністрування дозволяє керувати автомобілями для оренди, типами автомобілів, та наявністю додаткових послуг;

– можливість задати різні додаткові послуги для кожного автомобіля, наприклад: дитяче крісло, GPS навігація;

– легкий та інтуітивно зрозумілий інтерфейс користувача;

– керування резервуванням дозволяє адміністратору додавати, редагувати і видаляти замовлення, керувати даними клієнтів;

– сайт повинен бути адаптивним та кроссбраузерним;

– зворотній зв’язок.

Вимоги до реалізації сайту:

– мова програмування PHP;

– використання бази даних MySQL для зберігання інформації;

– сервер Apache;

– сайт має мати високу бистродію;

– архітектура MVC.

Система матиме такі ролі користувачив:

– адміністратор;

– менеджер;

– замовник.

2.2 Архітектура системи

Починаючи з проектування Веб-додаток використовує наступні елементи: веб-браузер, веб-сервер та базу даних. На цій основі було створено наступну діаграму компонентів:

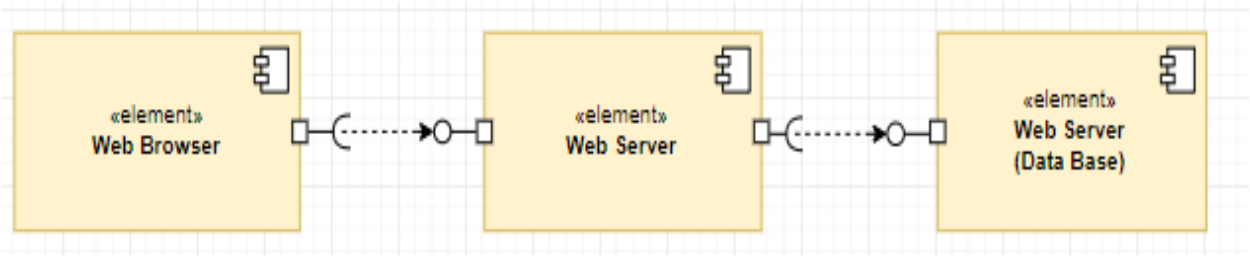

Рисунок 2.1 – Діаграма компонентів

Проектування архітектури при розробці на фреймворку дуже спрощується оскільки в методології фреймворків зазвичай закладені кращі практики програмної інженерії та просто дотримуючись цих правил можна уникнути багатьох проблем і помилок в проектуванні.

У сутності, фреймворк – це безліч конкретних і абстрактних класів, пов'язаних між собою і впорядкованих згідно з методологією фреймвока.

Конкретні класи зазвичай реалізують взаємні відносини між класами, а абстрактні класи являють собою точки розширення, в яких закладений базовий функціонал, що може бути використаний як є або адаптований під завдання конкретного додатка.

Для забезпечення розширення можливостей в більшості фреймворків використовуються техніки об'єктно-орієнтованого програмування наприклад, частини програми можуть успадковуватися від базових класів фреймворка або окремі модулі можуть бути підключені як доданки [3].

На початку роботи структура додатку з використанням Laravel має наступний вигляд:

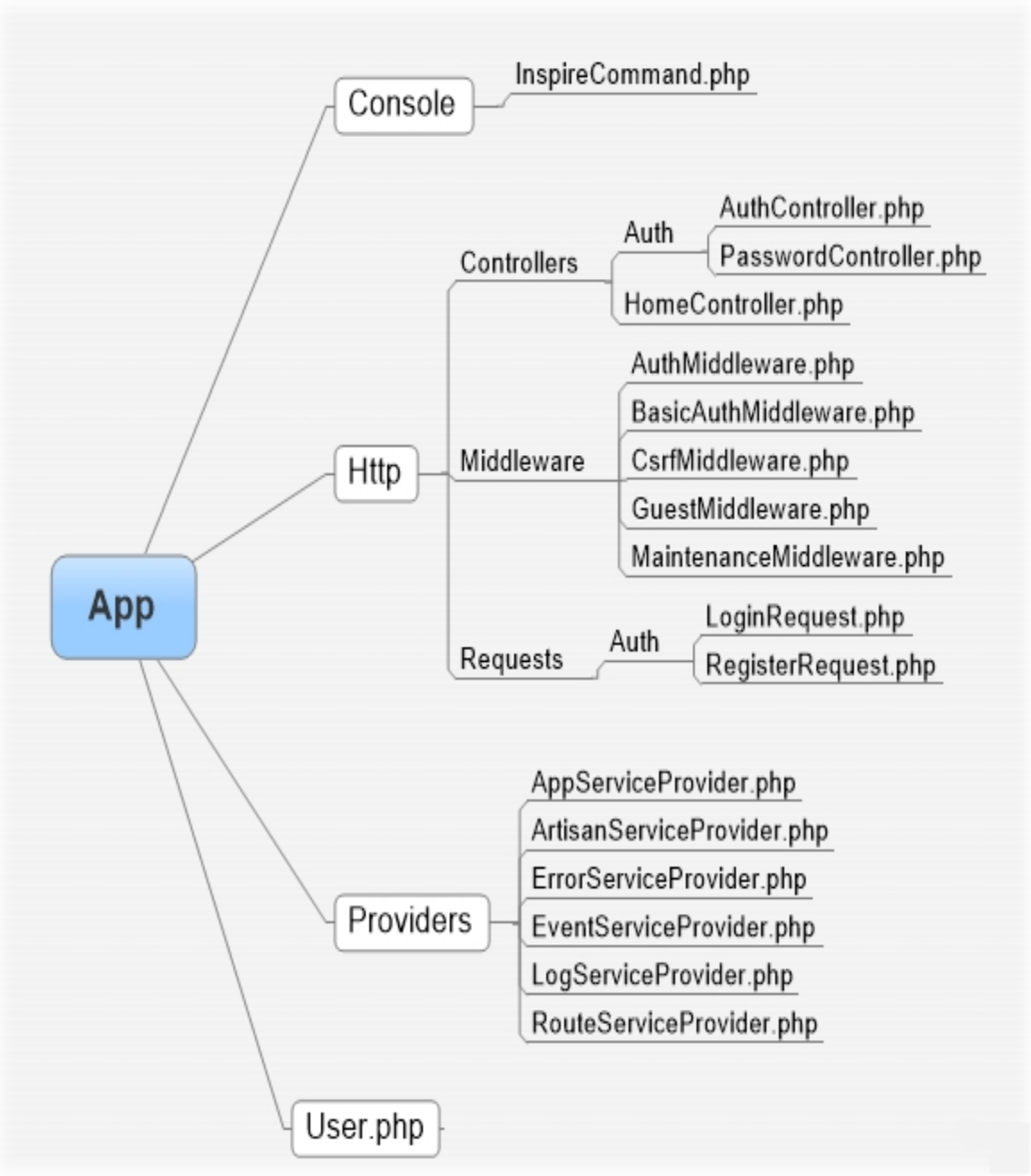

Рисунок 2.2 – Структура додатку

Розглянувши структуру додатків у Laravel (див. рис. 2.2), перейдемо до специфіки взаємодії кожного компонента із системою для виконання базового запиту.

Наступна діаграма описує типову послідовність процесу обробки користувальницького запиту додатком:

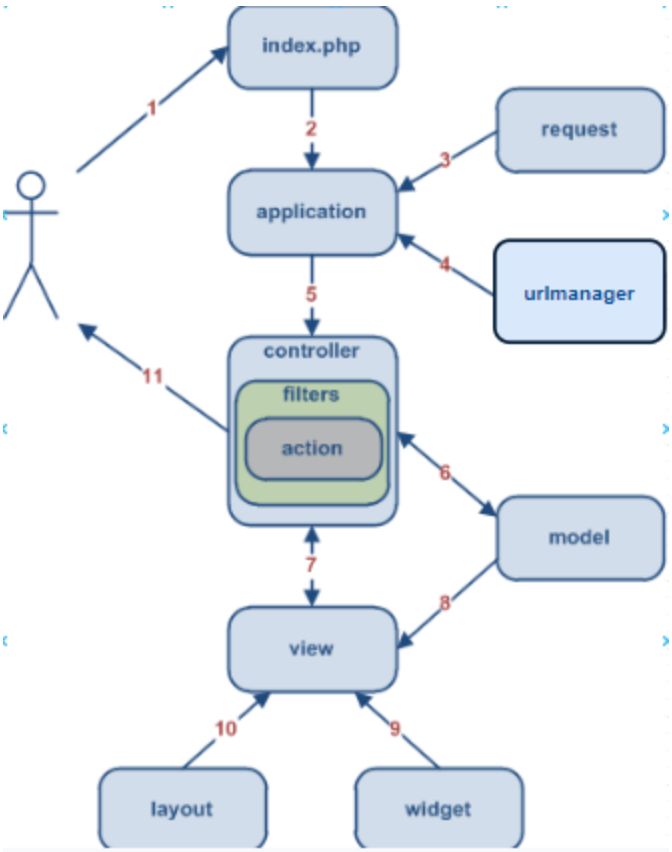

Рисунок 2.3 – Послідовність роботи додатку

Детальний опис послідовності процесу обробки користувальницького запиту додатком :

1) Користувач здійснює запит за допомогою URL, і веб-сервер обробляє його, запускаючи скрипт ініціалізації index.php;
2) Скрипт ініціалізації створює екземпляр додатку і запускає його на виконання;
3) Додаток отримує детальну інформацію про запит користувача від компонента додатка request;
4) Додаток визначає запитані контролер і дію;
5) Додаток створює екземпляр запитуваного контролера для подальшої обробки запиту користувача;
6) Дія зчитує з бази даних модель;
7) Дія підключає уявлення;
8) Уявлення отримує і відображає атрибути моделі;

9) Уявлення підключає деякі віджети;

10) Сформоване уявлення вставляється в макет сторінки;

11) Дія завершує формування уявлення і виводить результат користувачеві.

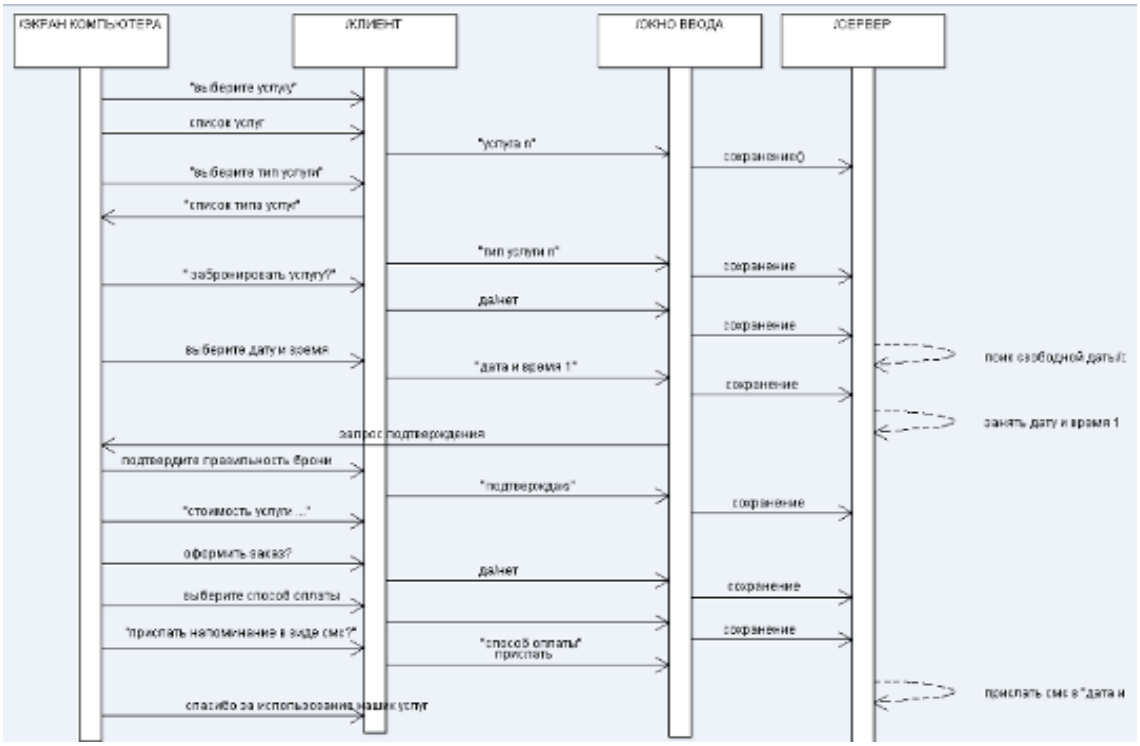

Рисунок 2.5 – Діаграма послідовності

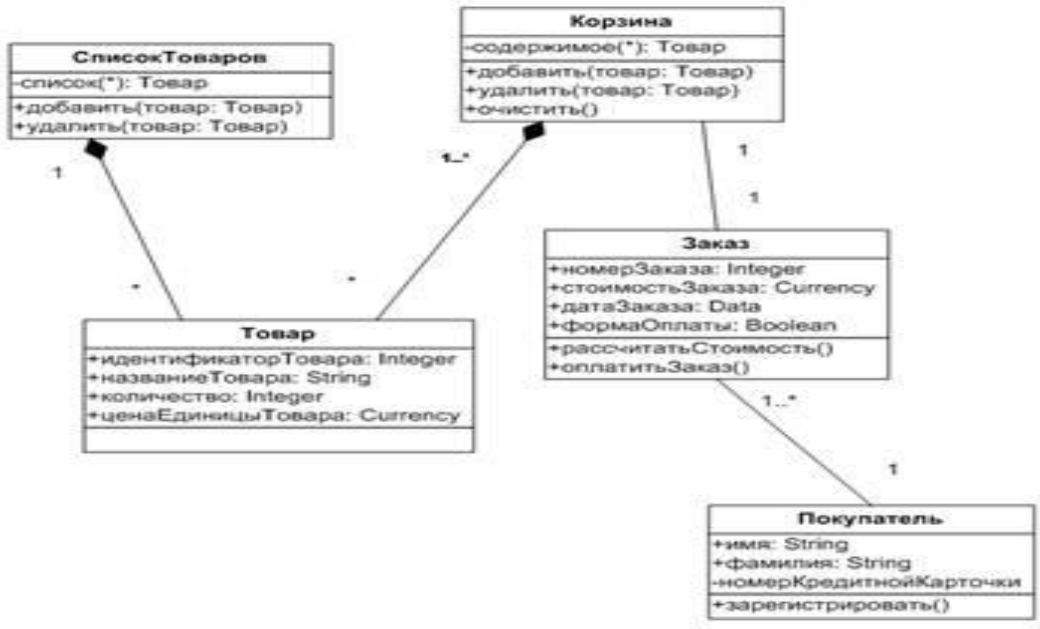

Рисунок 2.6 – Діаграма класів

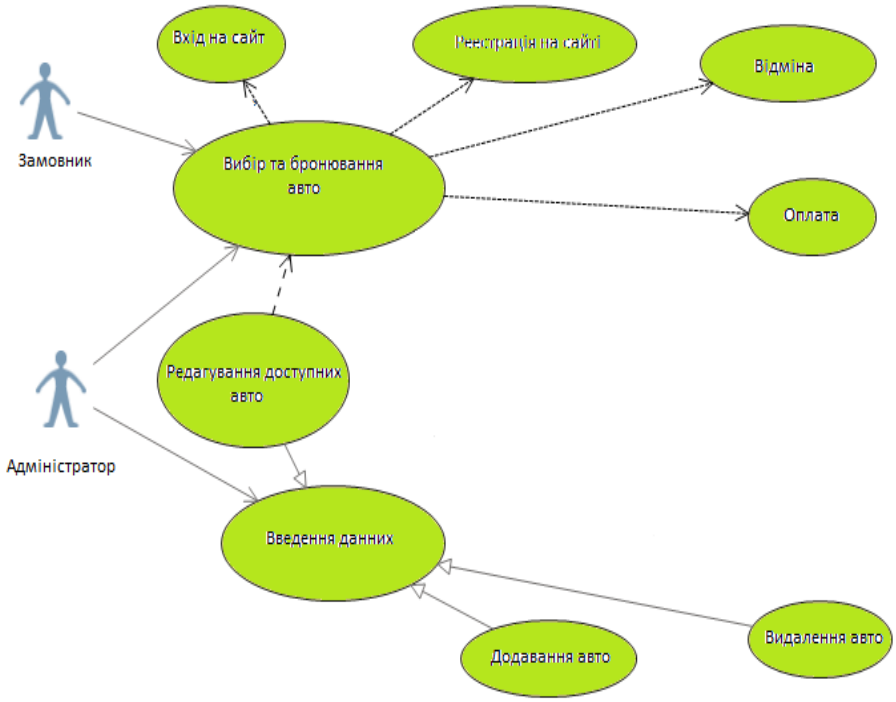

Рисунок 2.4 – Діаграма використання

Варіант використання «Бронювання авто»:

Система запитує ім'я користувача та пароль.

Якщо користувач не зареєстрований в системі, то «Перший альтернативний потік».

Користувач вводить ім'я і пароль.

Якщо логін або пароль введені невірно, то «Другий альтернативний потік».

Система перевіряє ім'я і пароль, після чого відкривається доступ в систему.

Користувач вибирає з каталогу необхідне йому авто:

Вибір по категорії.

Якщо користувач вибрав не той товар, то «Третій альтернативний потік»

Користувач переходе до бронювання.

Користувач оформляє заявку.

2.3 Життєвий цикл додатку Laravel

Точкою входу для всіх запитів до додатка Laravel є файл. Всі запити направляються в цей файл конфігурацією веб-сервера. Файл не містить багато коду. Швидше, його можна вважати відправною точкою для завантаження решти фреймворка .public /index.php index.php

Файл завантажує Composer генерується визначення Автозавантажувача, а потім повертає екземпляр додатку Laravel зі сценарію. Перша дія, що виконується самим Laravel, - створення екземпляру контейнера додатка / servece .index.php bootstrap / app.php

HTTP / Консольні Ядра

Потім вхідний запит відправляється або ядру HTTP, або ядру консолі, в залежності від типу запиту, що надходить в додаток. Ці два ядра служать центральним місцем, через яке проходять усі запити. Детальныше про ядро HTTP, яке знаходиться в .app / Http / Kernel.php

Ядро HTTP розширює клас, який визначає масив, який буде запущений до виконання запиту. Ці завантажувачі налаштовують обробку помилок, налаштовують ведення журналу, виявляють середу додатка і виконують інші завдання, які необхідно виконати до фактичної обробки запросу.Illuminate \ Foundation \ Http \ Kernel bootstrappers

Ядро HTTP також визначає список проміжного програмного забезпечення HTTP, через яке всі запити повинні пройти, перш ніж вони будуть оброблені додатком. Це проміжне програмне забезпечення обробляє читання і запис сеансу HTTP, визначає, чи знаходиться додаток в режимі обслуговування, перевіряє токен CSRF і багато іншого.

Підпис методу для методу ядра HTTP handle доволі проста: отримати Request і повернути Response.

Постачальники послуг

Одним з найбільш важливих дій початкового завантаження ядра є завантаження постачальників послуг. Усі постачальники послуг для додатка настроюються в масиві файлу конфігурації.

Спочатку метод буде викликатися для всіх постачальників, потім, після реєстрації всіх постачальників, буде викликаний метод.config / app.php providers register boot.

Постачальники послуг відповідають за завантаження всіх різних компонентів інфраструктури, таких як база даних, черга, перевірка та компоненти маршрутизації. Оскільки вони завантажують і налаштовують кожну функцію, запропоновану платформою, постачальники послуг є найбільш важливим аспектом всього процесу початкового завантаження Laravel.

Запит на відправку

Після того, як додаток було завантажено і всі постачальники послуг зареєстровані, Request будет передано маршрутизатору для відправки. Маршрутизатор відправить запит на маршрут або контролер, а також запустить будь проміжне програмне забезпечення для конкретного маршруту [11].

2.4 Висновки до другого розділу

Після аналізу вимог сайту , фреймворк Laravel було обрано для створення проекту. Вибір був здійснений згідно з критеріями які були оглянуті в розділі 1. Laravel має багато можливостей які виділяють цей фреймворк на тлі інших і роблять розробку з його допомогою дуже швидкою і комфортною.

3 ПРИКЛАД РОБОТИ. ТЕСТУВАННЯ

3.1 Початок роботи

На основі проведеного аналізу методів та засобів розробки Веб-додатків було створено безпосередньо Веб-сайт, його структуру та дизайн.

Мовою програмування було обрано PHP, а саме фреймворк Laravel, оскільки після порівняння у першому розділі виявилось що його можливостей буде достатньо для реалізації поставленої задачі.

Інтерфейс було спроектовано таким чином, щоб користувачу було зрозуміле користування на інтуітивному рівні, без довідок та інструкцій.

Розробка інтерфейсу велась за наступними етапами (див. рис. 3.1):

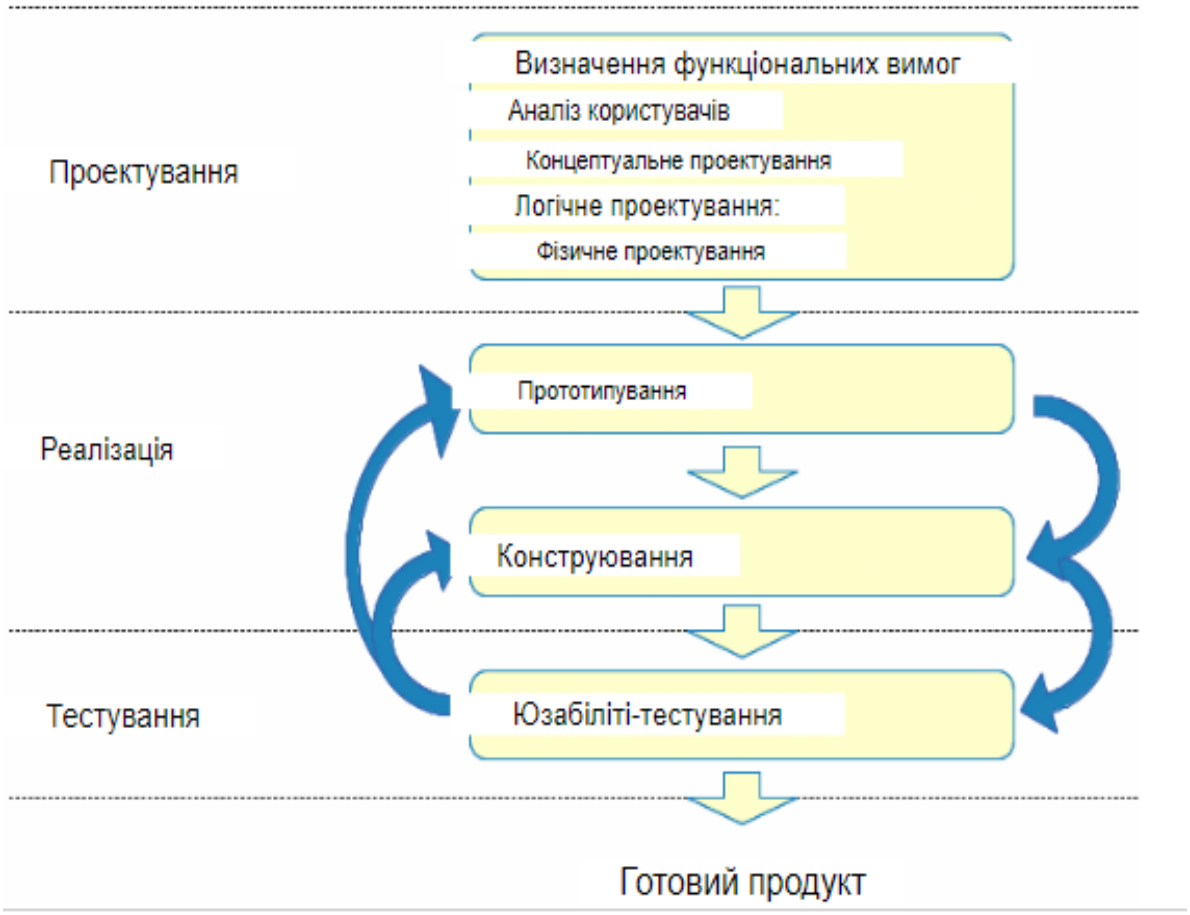

Рисунок 3.1 – Етапи розробки інтерфейсу

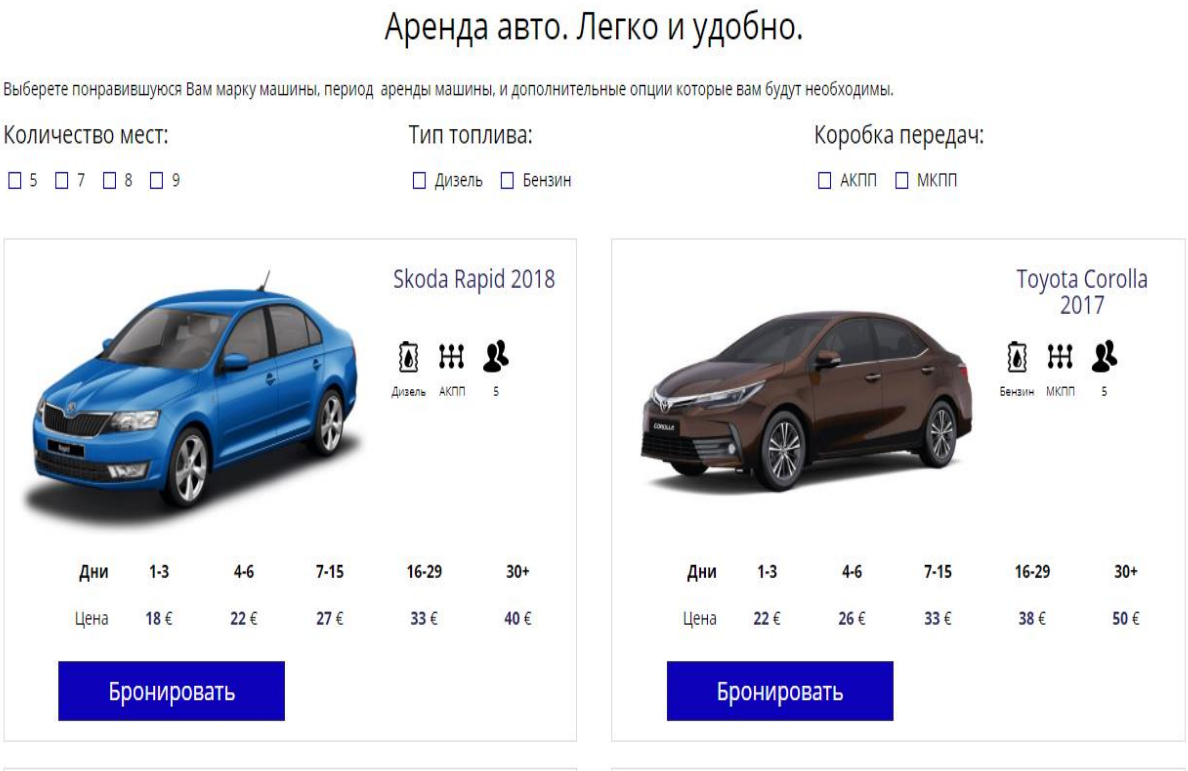

Рисунок 3.2 – Головна стрінка

Починаючи з головної сторінки сайту користувач одразу може переглянути усі доступні для оренди автомобілі, обрати та ознайомитись з деякими характиками машини, наприклад такими як:

– тип палива (дизельне паливо, або бензин);
– тип коробки передач(автоматична, або механічна);
– кількість місць.

Ціни варіюються залежно від класу обраного авто, та від кількості днів оренди. Деякі додаткові послуги що можна обрати беспосередньо при бронюванні такі як дитяче крісло або GPS-навігатор є безкоштовними тому не входять в початкову вартість.

А за умови якщо користувачеві підходить обране авто, він може перейти до його бронювання (див. рис. 3.1)

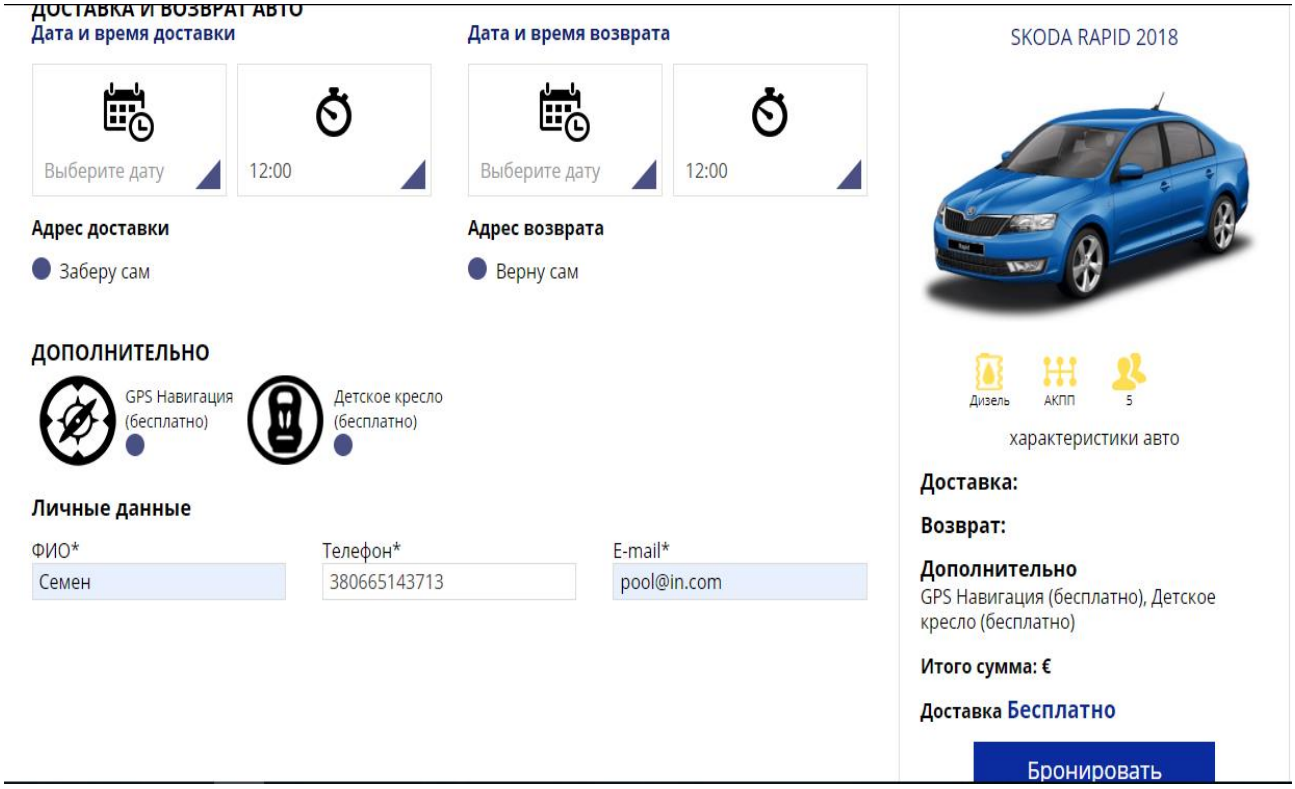

Рисунок 3.3 –Бронювання авто

На данному етапі користувач обирає дату і час замовлення, та дату повернення авто. А також за необхідності вибирає додаткові послуги які були описані вище.

Для створення заявки користувачеві неодхідно внести деякі особисті данні, а саме ПІБ, контактний номер телефону та E-mail.

Після заповнення усіх необхідних форм користувач побаче підсумкову вартість свого замовлення.

3.2 Адмінпанель

Керування сайтом виконується засобами адмінпанелі. З її допомогою адміністратор може стежити за замовленнями, додавати нові позиції для перегляду, та додаткові послуги.

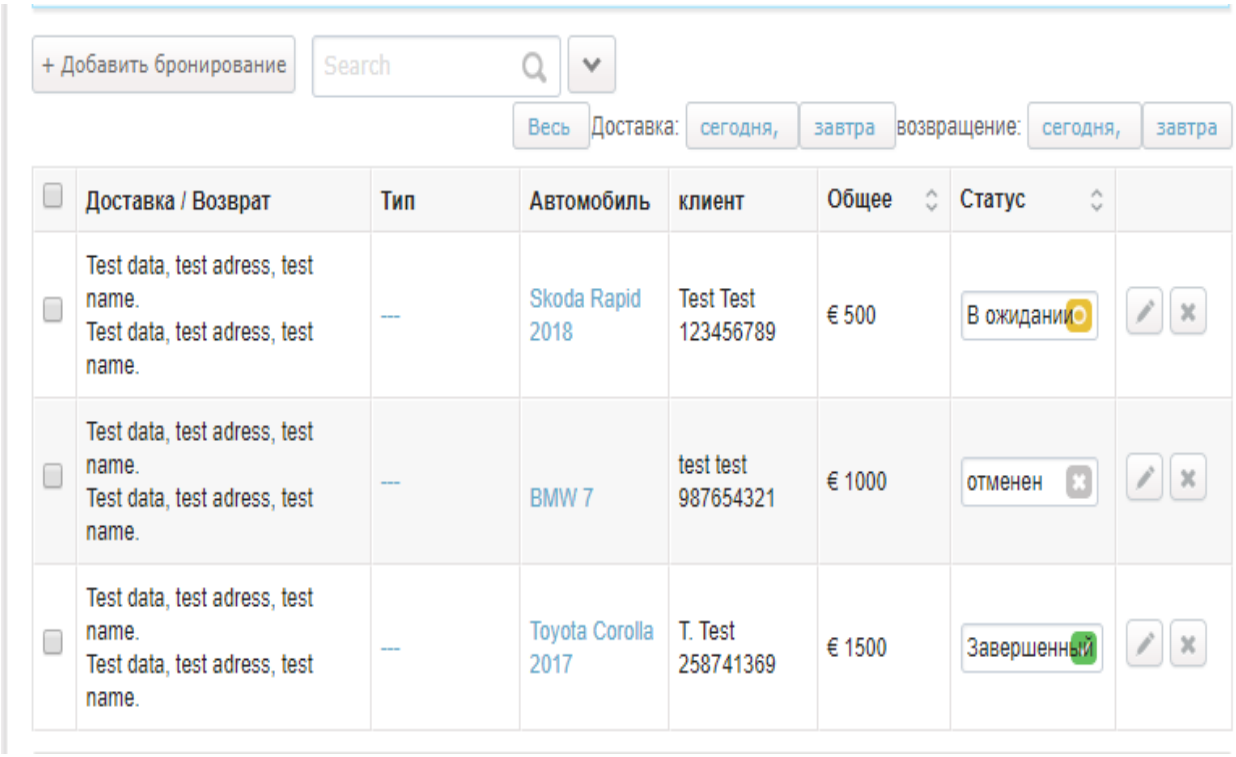

Рисунок 3.4 – Управління замовленнями

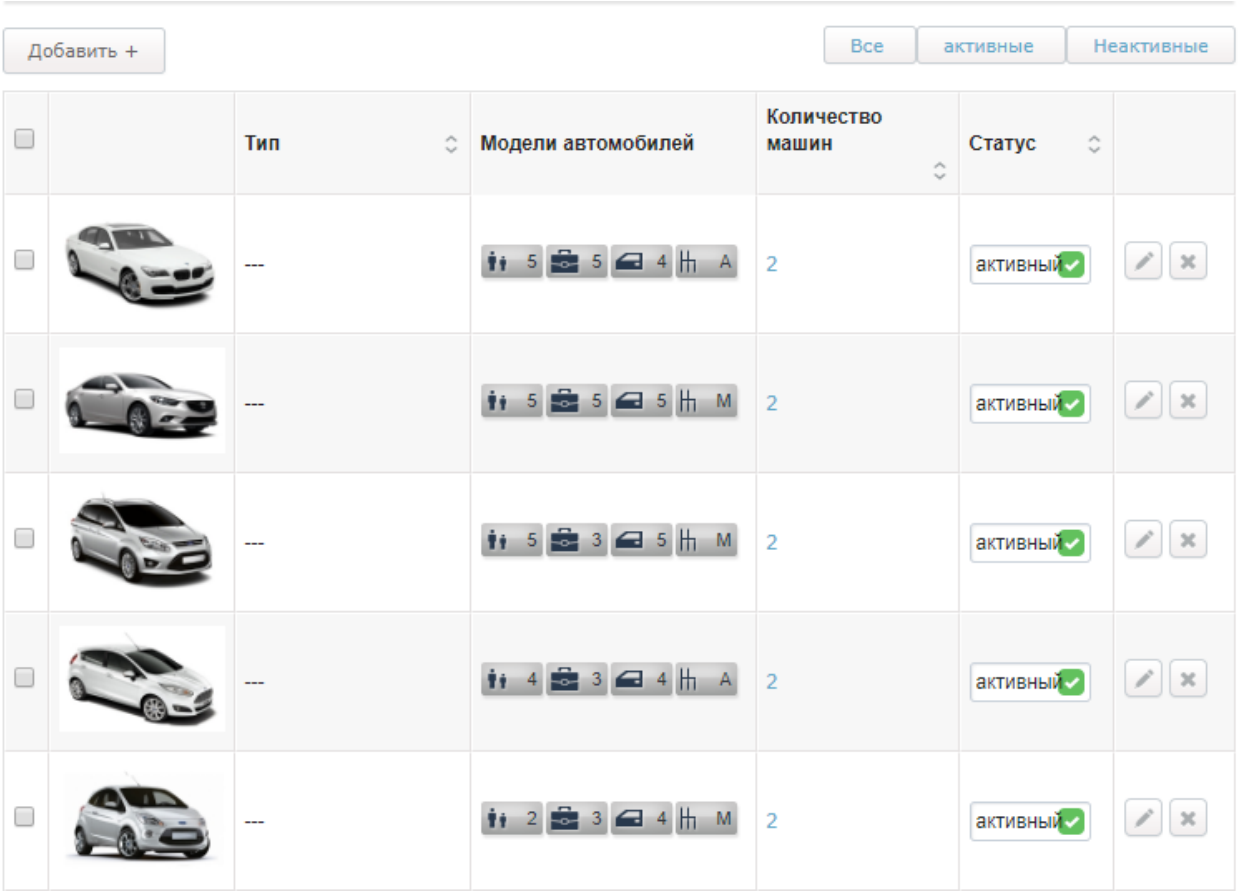

Рисунок 3.5 – Додавання нових авто

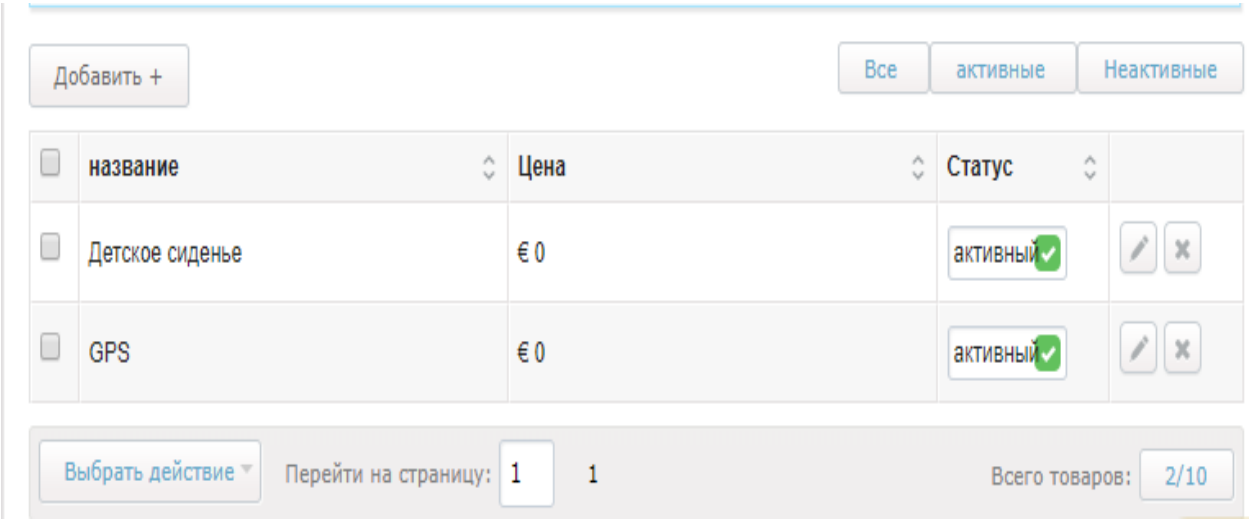

Рисунок 3.6 – Додавання супутніх послуг

Завдяки адмінпанелі адміністратор оримує можливість підтримувати сайт в належному стані, слідкувати за його функціонуванням та постійно доповнювати.

3.3 Тестування

Після розробки Веб-додатка до нього було застосовано ручне тестування. Для цього розроблено тест план, відповідно до тест плану були написані тест-кейси.

3.3.1 Тест план

Зміст документу

Опис процесу тестування, стратегії та підходів до сайту.

Цілі документу:

– спланувати управління тестуванням та технічну підтримку тестування в ході життєвого циклу розробки системи;

– забезпечення повноти тестування відповідно вимог до продукту,

охоплення всіх типів тестування, опис необхідної проектної документації;

– організація контролю процесу тестування, дефектів, необхідного програмного забезпечення;

– визначити графік робіт, описати використовувану методологію і стратегію тестування;

– визначення критеріїв якості;

– визначення ресурсів, необхідних для реалізації проекту;

– опис документації, необхідної для забезпечення тестування проекту.

Цільова аудиторія.

Документ призначений тестувальнику та служить для розуміння цілей і змісту робіт з тестування програмного продукту, визначає і описує перелік робіт і етапи тестування програми.

Цілі тестування.

Основними цілями тестування є:

– виявлення проблем, пов’язаних з невідповідністю вимогам до програмного продукту що розробляється;

– відстеження статусу проблем;

– надання сумарною оцінки якості продукту;

– зниження ризиків проекту, пов’язаних з якістю продукту, що розробляється.

Об’єм тестування.

Тестуванню на першій ітерації підлягають всі функції системи як ізольовано, тестування роботи функцій системи всередині модулів, так і в цілому, тестування взаємодії модулів системи, виконання бізнес-процесів в системі, перевірка повних циклів життя документів.

Тестування інсталяції та роботи системи в різних конфігураціях.

У процесі проведення робіт з тестування на другий ітерації передбачається проведення регресійного тестування – перевірка виправлення

дефектів, зареєстрованих на першій ітерації, тест нової функціональності, виконання бізнес-процесів в системі, перевірка повних циклів життя документів.

Нетестуємі компоненти.

На етапі складання плану тестування таких функціональних компонентів визначено не було.

Стратегія тестування.

Основним методом перевірки даного програмного продукту буде ручне функціональне тестування з використанням методу «чорного ящика», який базується на використанні вимог і специфікацій, і не передбачає наявності будь-яких спеціальних знань про конфігурацію і внутрішню структуру об'єкта випробувань.

У процесі тестування додатка було застосовано такі види тестування:

Функціональне тестування – це процес тестування програмного продукту на предмет відповідності вимогам до програмного продукту і правильності реалізації всіх характеристик, що були передбачені в продукті здійснюється вручну тестувальником.

Функціональний тест – основний вид тестування, під час якого перевіряються основна функціональність програмного продукту при стандартному його використанні, а також перевіряється нестандартне використання програмного продукту, кордони переповнення масивів даних, введення спеціальних символів і т.д. Для проведення функціонального тесту використовується весь обсяг створеної тестової документації. Особлива увага приділяється тестуванню бізнес-процесів системи, логічних ланцюжків послідовностей дій користувачів, взаємодія функцій системи між собою, коректне оновлення і відображення даних, логічне завершення операцій.

Основна мета функціонального тестування – забезпечити максимально можливе покриття тестами програмного продукту.

Технічне тестування.

Технічне тестування має свою основну мету перевірки нефункціональних вимог до програмного продукту.

Даний вид тестування включає в себе кілька типів тестів:

– навантажувальне тестування – перевірка працездатності системи при роботі з великою кількістю даних;

– тест продуктивності – перевірка швидкості роботи системи час відгуку при виклику різних функцій;

– стрес тести – перевірка працездатності системи в критичних умовах (завантаження процесора виконанням додаткових задач, і т.п.);

– відновлення працездатності системи при збоях.

Тести показали що система справляється з різними за обсягом навантаженнями на достатньому рівні, поводить себе стабільно та має зрозумілий для користувача інтерфейс. Але, деякі аспекти не відповідають очікуванням тому потребується ще багато часу, на доробку та опрацювання відхилень в роботі системи [5].

3.3.2 Тест-кейси

Тест-кейс № 1. Некоректні особисті данні користувача.

Кроки:

Перейти до бронювання авто.

Вписати в поле «Телефон»букви замість цифр.

Натиснути кнопку «Бронювати».

Очікуваний результат

З’являється повідомлення про помилку «Не верный формат номера» , бронювання не відбувається.

Тест-кейс № 2. Не коректна дата повернення авто.

Кроки:

Перейти до бронювання авто.

В поле «Дата время возврата» вибрати не коректну дату.

Натиснути кнопку «Бронювати».

Очікуваний результат

З’являється попередження «Не верно выбрана дата возврата» , бронювання не відбувається.

Опис тест-кейсів зображено нижче, у таблиці 3.1.

Таблиця 3.1 – Таблиця тест-кейсів

	Тест 1	Тест 2
Умови		
Умови виконані	Ні	Так
Данні коректні	Так	Ні
Дія		
Виконати бронювання	Ні	Ні

Огляд процесу тестування

Підхід до процесу тестування можна відобразити у вигляді наступного рисунку 3.10. На початку надходять вимоги, далі вимоги аналізуються. З цього аналізу складається тестова документація, формується календарний план та план тестування. Після проходження тестування складається звіт про якість продукту та дифекти.

Для забезпечення процесу тестування потрібна розробка наступної проектної тестової документації:

– план тестування ;
– сценарій тестування;
– технічне тестування;
– журнал тестування.

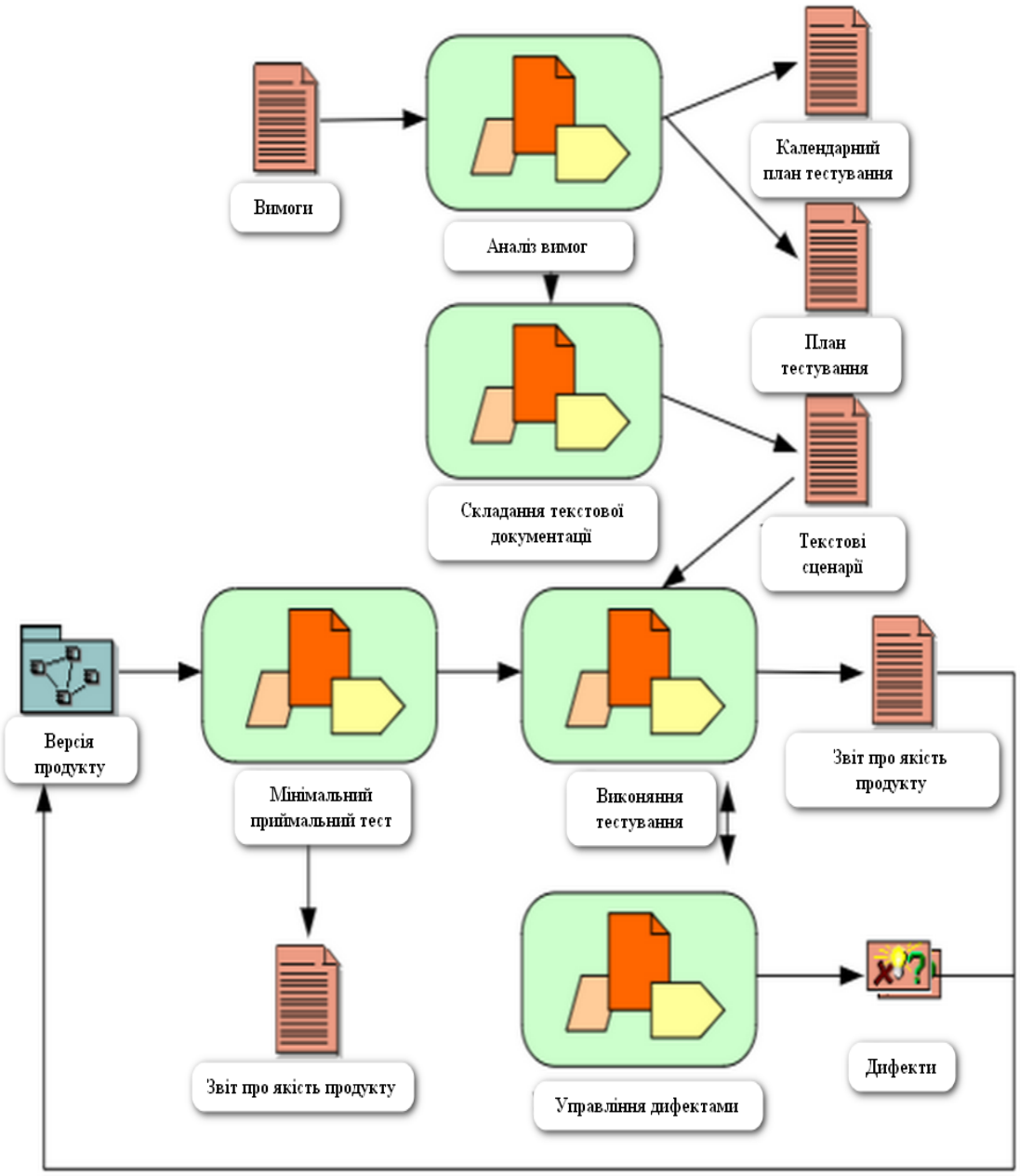

Рисунок 3.7 – Схема процесу тестування

Протестована версія має пройти мінімальний приймальний тест, після чого знов складається звіт про якість. Якщо приймальний тест не пройдено, то тестування повторюється.

3.4 Висновки до третього розділу

Розроблений Веб-сайт є дуже простим у використанні, користувач може на інтуітивному рівні здогадатися як саме прачювати з сайтом.

На рисунках 3.2 та 3.3 можна побачити основні положення в роботі з Веб-додатком.

За допомогою тестування було виявлено, що розроблена інформаційна система система поводить себе стабільно та має зрозумілий для користувача інтерфейс. Але, деякі аспекти не відповідають очікуванням тому потребується ще багато часу, на доробку та опрацювання відхилень в роботі сайта.

ВИСНОВКИ

У кваліфікаційній роботі було розглянуто область розробки Веб-сайтів, об’єктом дослідження є розробка веб-сайту «Оренди автомобілів» а також вивчення методів та засоби для розробки Веб-додатка.

Частину роботи становить дослідження та прівняння популярних фрейморків та інших засобів розробки.

На основі розглянутого матеріалу було реалізовано систему для перегляду та аренди авто.

За результатами тестування було виявлено що система справляється з різними навантаженнями на достатньому рівні, поводить себе стабільно, та має зрозумілий для користувача інтерфейс.

У переспективі можливості додатка надалі можуть бути використані для отримання прибутку в елетронному бізнесі.

ПЕРЕЛІК ПОСИЛАНЬ

1. W. Jason Gilmore and Eric L. Barnes, Easy E-Commerce Using Laravel and Strip, Nashville: Springer-Verlag, 2015. 110 p

2. Dayle Rees, Laravel: Code Bright, NewYork: Dayle Rees, 2013. 166 p.

3. Kelt Dockins, Kelt Dockins – Design Patterns in PHP and Laravel, London: Apress, 2017. 238 p.

4. Matt Stauffer, Laravel: Up and Running: A Framework for Building Modern PHP Apps, Sebastopol: O'Reilly Media Inc, 2016. 464 p.

5. Навчальний посібник «Методи тестування та оцінки якості програмного забезпечення із застосуванням Pairwise тестування» для студентів денної та заочної форми навчання. Полтава : ПолтНТУ 2016. 391 с.

6. Веб-фреймворки и с чем их едят URL: http://iwsm.ru/blog/show (дата звернення 16.10.2019).

7. Результаты тестирования шести ведущих фреймворков на производительность URL: http://www.alrond.com/ru (дата звернення 20.10.2019).

8. Використання PHP фреймворків в розробці сайту URL: http://ukrbukva.net (дата звернення 6.11.2019).

9. Обзоры Web-фреймворков URL: https://praktikatech.wordpress.com (дата звернення 15.11.2019).

10. Тот самый PHP-фреймворк для веб-ремесленников URL: https://laravel.ru (дата звернення 17.11.2019).

11. Laravel - Жизненный цикл URL: http://unetway.com (дата звернення 20.11.2019).