Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО «Предприятие ВТИ-Сервис»

наименование предприятия, организации, учреждения

Студента 4 курса, группы ПО-02б

курса, группы

Головиной Ирины Романовны

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

Оценка _____

_____

должность, звание, степень

_____ _____

фамилия и. о. подпись, дата

Руководитель практики от университета

Оценка _____

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А. _____

фамилия и. о. подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2024 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных.

ИС – информационная система.

ИТ – информационные технологии.

ПО – программное обеспечение.

РП – рабочий проект.

СУБД – система управления базами данных.

ТЗ – техническое задание.

ТП – технический проект.

UML (Unified Modelling Language) – язык графического описания для объектного моделирования в области разработки программного обеспечения.

1 Анализ предметной области

1.1 История интернет-магазинов

Прообраз современных интернет-магазинов появился задолго до создания мировой паутины. В 1979 году Михаэль Олдрич, занимавшийся поставками коммуникационных сетей в Великобританию, изобрёл телемагазин «videotex». Эта технология позволяла транслировать каталоги с товарами и контактными данными для связи с продавцом. Первый интернет-магазин, похожий на современные, появился в 1992 году. Чарльз Стэк создал интернет-магазин книг. В то время книги покупали по бумажным каталогам с ценами и описаниями. Чарльз оцифровал эти каталоги и перенёс их в формат, понятный для интернета. Книги не имели срока годности и не меняли спрос из-за модных тенденций. Многие интернет-магазины, появившиеся в начале 90-х, начинали свою деятельность с продажи печатной продукции. В 1994 году Джефф Безос придумал концепцию проекта «Amazon». Изначально в нём можно было купить только книги, но со временем ассортимент сильно расширился. Через интернет стали продавать больше вещей, которые раньше можно было купить только в обычном магазине. Позже «Amazon» стал предлагать постельное бельё, одежду и бытовую технику. Хоть код интернет-ресурса имел изъяны (например, можно было заказать отрицательное количество товара), этот магазин получил всеобщее признание. Первый российский интернет-магазин появился в 1997 году в Москве, он был полностью книжным, как и его предшественники. Через год в стране появились первые интернет-проекты по интернет-банкингу и автоматические интернет-шлюзы для обработки заказов за секунды. Российский интернет-трейдинг сделал резкий рывок вперёд, который определил его дальнейшее развитие. Появление интернет-магазинов стало настоящей революцией в сфере розничной торговли. Они дали покупателям возможность выбирать из огромного ассортимента товаров, сравнивать цены и делать покупки в любое удобное время. Кроме того, благодаря интернет-магазинам удалось сократить расходы на аренду торговых площадей и оплату труда персонала. Это позволило снизить

цены на товары, делая их более доступными для широкого круга потребителей.

1.2 Особенности продажи растений

Продавцы несут ответственность перед покупателями и должны соблюдать определённые правила. Они обязаны предоставить информацию о своём фирменном наименовании, месте нахождения организации и режиме работы. Эта информация должна быть размещена на вывеске магазина. Если продавец является индивидуальным предпринимателем, он также должен сообщить покупателю о своей государственной регистрации и наименовании органа, который её провёл. У продавцов должна быть книга отзывов и предложений, которую они обязаны предоставить покупателю по первому требованию. Это помогает улучшить качество обслуживания и создать положительную обратную связь. Правила продажи различных видов товаров должны быть наглядно и доступно представлены. Это важно для обеспечения прозрачности и доверия со стороны покупателей. Продавцы обязаны своевременно предоставлять покупателям достоверную информацию о товарах. Эта информация должна содержать наименование товара, сведения о стране происхождения (для импортных товаров), подтверждении соответствия, основных потребительских свойствах товара. Также необходимо указать цену и условия приобретения товара, информацию о разрешении на ввоз определённых видов дикорастущих растений в Российскую Федерацию. Если кассовый чек не содержит видового названия и количества растений, то покупателю должен быть передан товарный чек с этой информацией. Продавцы обязаны предупредить покупателей о недостатках товаров не только устно, но и письменно. Растения являются непродовольственными товарами, которые не подлежат возврату или обмену на аналогичный товар другого размера, формы, цвета и т. д., если они не имеют недостатков. Перед покупкой растения необходимо тщательно проверить его и убедиться в соответствии вашим требованиям. Некоторые растения могут быть опасными и ядовитыми, особенно для детей и домашних животных. При продаже таких растений продав-

цы также должны предоставить информацию о возможной опасности, чтобы покупатели были внимательны и предприняли необходимые меры предосторожности.

1.3 Особенности заказа растений через Интернет

Продажа семян и саженцев по почте — это удобный и популярный способ покупки в России. Вам больше не нужно обходить множество магазинов в поисках нужных сортов и гибридов. Вы можете спокойно и внимательно изучить каталог или сайт интернет-магазина, не выходя из дома. Во многих регионах страны выбор семян и посадочного материала ограничен, а качество и ассортимент уступают тем, что представлены в крупных интернет-магазинах и каталогах компаний. Поэтому жители регионов часто заказывают товары по почте, так как не могут найти нужный сорт или вид растения в обычных магазинах. Дистанционная торговля семенами и посадочным материалом может осуществляться двумя способами: через каталоги и через интернет-магазины. До недавнего времени основным видом торговли была продажа через каталоги. С развитием интернета в нашей стране, на смену этому виду торговли пришла торговля через интернет-магазины. Самый популярный и простой товар с точки зрения хранения и доставки — это семена [8]. Их можно заказывать круглый год, многие огородники планируют свои посевы ещё с осени и делают заказы заранее, в октябре-декабре. Такие посылки приходят адресату до Нового года. Однако пик заказов обычно приходится на январь-февраль. Заказывая семена, всегда нужно учитывать сроки посева каждой культуры. Затем отнимайте 3-5 дней на комплектацию заказа и 5-20 дней на доставку почтой, в зависимости от региона. В итоге получается, что заказ семян следует делать минимум за месяц до посева. Второй по популярности товар — это луковичные и многолетние растения. Этот вид товара имеет два сезона продаж — весна и осень. Весной предлагается огромный ассортимент многолетних растений, таких как астильбы, хосты, пионы травянистые, гейхеры, а также луковичных растений, таких как гладиолусы, лилии, амариллисы. Срок приёмки заказов с ноября по апрель, отправка зака-

зов начинается в марте. Осенний сезон более короткий и предлагает тюльпаны, нарциссы, крокусы, гиацинты и различные мелколуковичные растения. Заказы принимаются с июля по август, отправка осуществляется с августа по сентябрь. Саженцы плодовых и декоративных кустарников и деревьев — следующий по популярности товар. Приём заказов начинается с ноября, а рассылка стартует после 8 марта. Саженцы кустарников и деревьев, а также луковичные и многолетние растения — это более специфический продукт, чем семена. Для сохранения качества им нужны определённые температурные условия. При выборе оплаты наложенным платежом покупатель оплачивает посылку при получении на почте. Стоимость посылки состоит из нескольких компонентов: стоимости товара, стоимости доставки, которая зависит от веса посылки, расстояния и выбранного способа доставки, а также комиссии за обработку наложенного платежа, которую взимает почтовая служба или курьерская компания. При выборе предоплаты покупатель заполняет квитанцию в каталоге или получает её на электронную почту. Отправка заказа осуществляется после поступления денег на расчётный счёт продавца. При получении товара на почте покупатель оплачивает только расходы за почтовую пересылку.

2 Техническое задание

2.1 Основание для разработки

Полное наименование системы: «Интернет-магазин декоративных растений».

Основанием для разработки программы является тема выпускной квалификационной работы.

2.2 Цель и назначение разработки

Программно-информационная система предназначена для продажи растений через Интернет. С системой должны работать следующие группы пользователей:

- администратор интернет-магазина;
- посетитель интернет-магазина.

Администратор должен иметь возможность добавлять, удалять и редактировать товары и свои категории. Например, «акции» или «новинки». Посетитель интернет-магазина должен иметь возможность просматривать товары, добавлять их в корзину и оформлять заказ.

Задачами данной разработки являются:

- проектирование базы данных;
- проектирование пользовательского интерфейса;
- разработка методов отображения данных из базы данных;
- разработка методов управления базой данных администратором.

2.3 Требования к программной системе

2.3.1 Требования к данным

Входными данными для системы являются:

- информация о пользователе, предоставляемая им в процессе регистрации в системе;

- информация о пользователе, предоставляемая им в процессе оформления заказа;
- информация о товарных позициях (отдельных товарах) заказа;
- информация о товарах, вводимая администратором;
- информация о категориях, вводимая администратором.

Выходными данными для системы являются:

- стоимость всех товаров в корзине;
- список заказов;
- уведомление о добавлении товара в корзину;
- уведомление об удалении товара из корзины;
- уведомление об оформленном заказе;
- сообщения об ошибках.

2.3.2 Функциональные требования

В процессе разработки сайта была создана модель, наглядно демонстрирующая, как пользователи будут взаимодействовать с сайтом. Для создания диаграммы вариантов использования был использован унифицированный язык визуального моделирования UML. Диаграмма вариантов использования описывает функциональное назначение системы. Это исходное представление системы, которое используется при её проектировании и разработке.

Система представлена в виде набора прецедентов — наборов действий, которые система выполняет для актёров, взаимодействующих с ней. Актёром может быть человек или техническое устройство, которые взаимодействуют с системой извне.

На рисунке 2.1 изображены прецеденты для администратора сайта.

На рисунке 2.2 изображены прецеденты для покупателя.

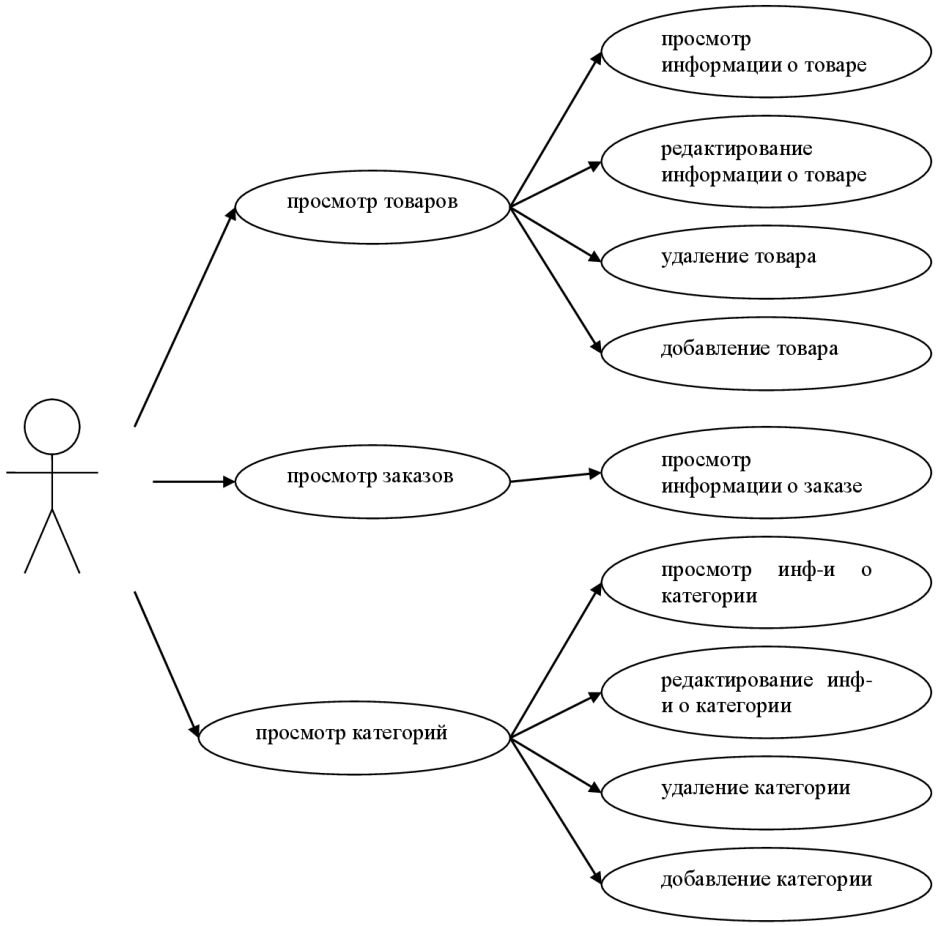

Рисунок 2.1 – Прецеденты для администратора

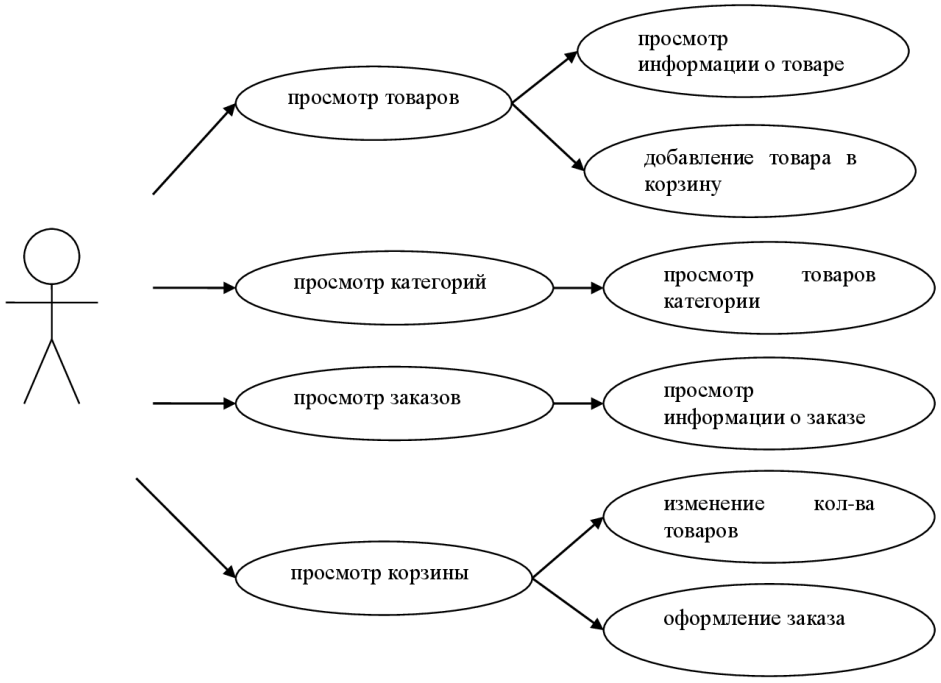

Рисунок 2.2 – Прецеденты для покупателя

2.3.2.1 Прецедент администратора «Просмотр информации о товаре»

Основной успешный сценарий для прецедента «Просмотр информации о товаре»:

1. Открыть страницу товаров.
2. Нажать на кнопку «Открыть» рядом с товаром, о котором нужно узнать информацию.
3. Ознакомиться с представленной информацией.

2.3.2.2 Прецедент администратора «Редактирование информации о товаре»

Основной успешный сценарий для прецедента «Редактирование информации о товаре»:

1. Открыть страницу товаров.
2. Нажать на кнопку «Изменить» рядом с товаром, информацию о котором нужно отредактировать.
3. Переписать информацию в некоторых полях.
4. Нажать на кнопку «Редактировать».

2.3.2.3 Прецедент администратора «Удаление товара»

Основной успешный сценарий для прецедента «Удаление товара»:

1. Открыть страницу товаров.
2. Нажать на кнопку Удалить» рядом с нужным товаром.

2.3.2.4 Прецедент администратора «Добавление товара»

Основной успешный сценарий для прецедента «Добавление товара»:

1. Открыть страницу товаров.
2. Нажать на кнопку «Добавить» в нижней части страницы.
3. Написать информацию о товаре.
4. Нажать на кнопку «Готово».

2.3.2.5 Прецедент администратора «Просмотр информации о заказе»

Основной успешный сценарий для прецедента «Просмотр информации о заказе»:

1. Открыть страницу заказов.
2. Нажать на кнопку «Открыть» рядом с нужным заказом.
3. Ознакомиться с представленной информацией.

2.3.2.6 Прецедент администратора «Просмотр информации о категории»

Основной успешный сценарий для прецедента «Просмотр информации о категории»:

1. Открыть страницу категорий.
2. Нажать на кнопку «Открыть» рядом с категорией, о которой нужно узнать информацию.
3. Ознакомиться с представленной информацией.

2.3.2.7 Прецедент администратора «Редактирование информации о категории»

Основной успешный сценарий для прецедента «Редактирование информации о категории»:

1. Открыть страницу категорий.
2. Нажать на кнопку «Изменить» рядом с категорией, информацию о которой нужно Отредактировать.
3. Переписать информацию в некоторых полях.
4. Нажать на кнопку «Редактировать».

2.3.2.8 Прецедент администратора «Удаление категории»

Основной успешный сценарий для прецедента «Удаление категории»:

1. Открыть страницу категорий.

2. Нажать на кнопку Удалить» рядом с нужной категорией.

2.3.2.9 Прецедент администратора «Добавление категории»

Основной успешный сценарий для прецедента «Добавление категории»:

1. Открыть страницу категорий.
2. Нажать на кнопку «Добавить» в нижней части страницы.
3. Написать информацию о категории.
4. Нажать на кнопку «Готово».

2.3.2.10 Прецедент покупателя «Просмотр информации о товаре»

Предусловие: открыта страница всех товаров или товаров категории.

Основной успешный сценарий для прецедента «Просмотр информации о товаре»:

1. Нажать на кнопку «Подробнее» в карточке интересующего товара.
2. Ознакомиться с представленной информацией.

2.3.2.11 Прецедент покупателя «Добавление товара в корзину»

Предусловие: открыта страница всех товаров или товаров категории.

Основной успешный сценарий для прецедента «Добавление товара в корзину»:

1. Нажать на кнопку «Подробнее» в карточке интересующего товара.
2. Ознакомиться с представленной информацией.
3. Нажать на кнопку «Добавить в корзину».

2.3.2.12 Прецедент покупателя «Просмотр товаров категории»

Основной успешный сценарий для прецедента «Просмотр товаров категории»:

1. Открыть страницу «Категории».
2. Нажать на интересующую категорию.
3. Ознакомиться с представленными товарами.

2.3.2.13 Прецедент покупателя «Просмотр информации о заказе»

Предусловие: наличие заказов.

Основной успешный сценарий для прецедента «Просмотр информации о заказе»:

1. Открыть страницу заказов.
2. Нажать на кнопку «Открыть» рядом с нужным заказом.
3. Ознакомиться с представленной информацией.

2.3.2.14 Прецедент покупателя «Изменение количества товара»

Предусловие: наличие товаров в корзине.

Основной успешный сценарий для прецедента «Изменение количества товара»:

1. Открыть корзину.
2. Нажимать на кнопку увеличения или уменьшения количества около нужного товара, пока не получится требуемое число.

2.3.2.15 Прецедент покупателя «Оформление заказа»

Предусловие: наличие товаров в корзине.

Основной успешный сценарий для прецедента «Оформление заказа»:

1. Открыть корзину.
2. Нажать на кнопку «Оформить заказ».
3. Ввести необходимую информацию.
4. Нажать на кнопку «Оформить заказ».

2.3.3 Требования пользователя к интерфейсу web-сайта

Сайт должен иметь следующие страницы:

- главная страница со всеми товарами;
- страница категорий товаров;
- страница товаров, принадлежащих выбранной категории;
- корзина;

- список заказов.

Администратору также необходимы страницы для редактирования, удаления и добавления товаров и категорий.

Товары на главной странице должны быть представлены в виде карточек одинакового размера, содержащих фото, название и цену товара.

Макет страницы с товарами приведён на рисунке 2.3.

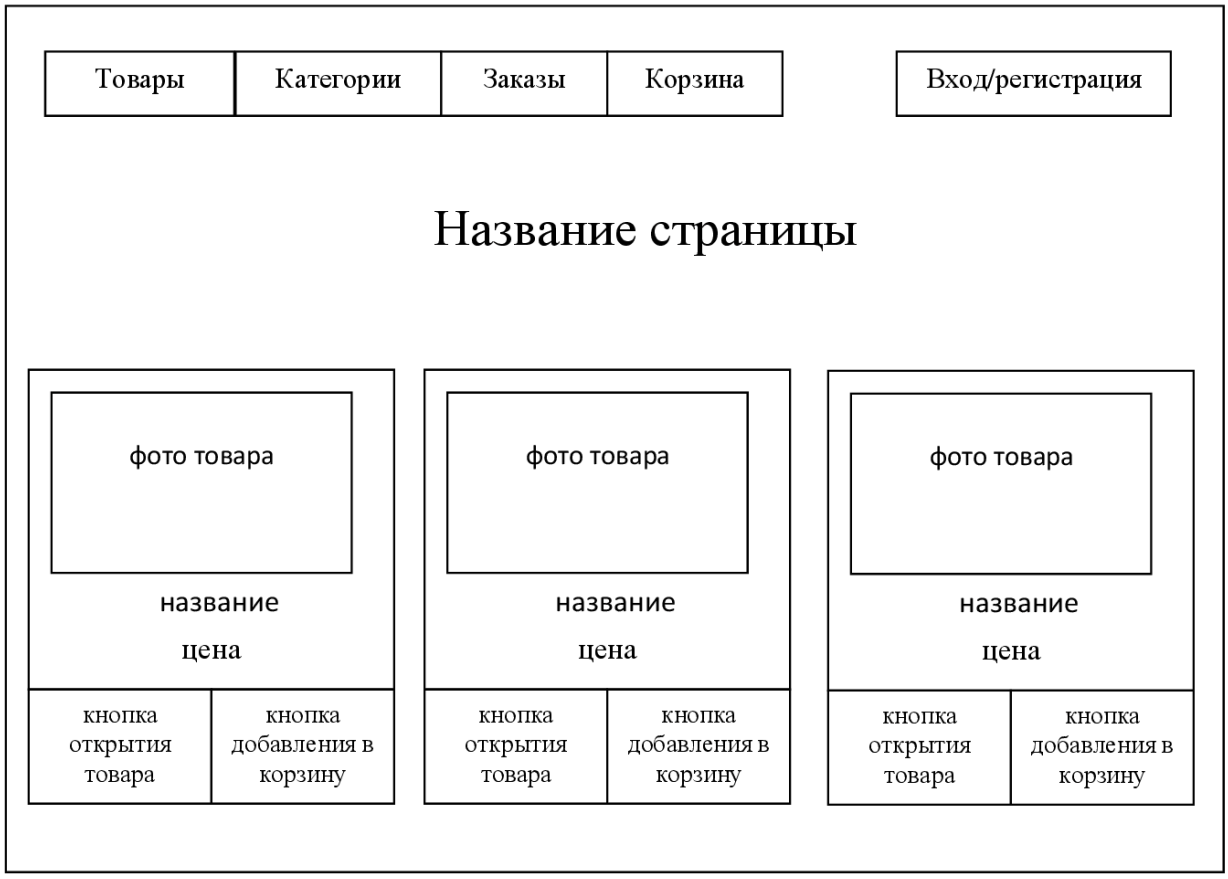

Рисунок 2.3 – Макет страницы с товарами

2.3.4 Нефункциональные требования

2.3.4.1 Требования к безопасности

Необходимо устранить уязвимости, возможные для веб-приложений:

- Межсайтовые скрипты (XSS) — это уязвимость системы безопасности, которая позволяет злоумышленнику размещать клиентские скрипты (обычно JavaScript) на веб-страницах. Когда другие пользователи загружают затронутые страницы, будут выполняться скрипты злоумышленника, что

позволяет злоумышленнику украсть cookie-маркеры и маркеры сеанса, изменить содержимое веб-страницы с помощью DOM или перенаправить браузер на другую страницу.

- SQL-инъекция (Внедрение SQL-кода) — это попытка изменить запрос к базе данных. Ввести ее можно через форму или ссылку, которая передает параметры методом GET[15].

- Доступ обычного пользователя к возможностям администратора.

- Межсайтовая подделка запроса (CSRF) — это вид атак на сайт, при котором злоумышленник с помощью мошеннического сайта или скрипта заставляет браузер пользователя выполнять на доверенном сайте действия от его имени: отправлять сообщения, менять пароли, переводить деньги и пр. В атаке используются недостатки протокола HTTP. Чтобы стать жертвой, пользователю достаточно перейти по вредоносной ссылке.

2.3.4.2 Требования к ПО

Для реализации программной системы необходимо подготовить следующие компоненты:

- пакетный менеджер Composer для управления зависимостями;
- фреймворк Laravel;
- интерпретатор языка программирования PHP;
- СУБД MySQL;
- веб-сервер, поддерживающий PHP.

Laravel совместим со всеми современными операционными системами, включая Windows, macOS и Linux.

2.3.4.3 Требования к аппаратному обеспечению

Для работы приложения требуется дисковое пространство не менее 2,5 Гб, минимум 2 Гб оперативной памяти и подключение к Интернету. Рекомендуется использовать процессор с 2 или более ядрами и частотой 2 ГГц или выше.

2.4 Требования к оформлению документации

Требования к стадиям разработки программ и программной документации для вычислительных машин, комплексов и систем независимо от их назначения и области применения, этапам и содержанию работ устанавливаются ГОСТ 19.102–77. Программная документация должна включать в себя:

1. Анализ предметной области;
2. Техническое задание;
3. Технический проект.

3 Технический проект

3.1 Компоненты приложения

Компоненты приложения и взамодействия между ними можно увидеть на диаграмме компонентов, приведённой на рисунке 3.1.

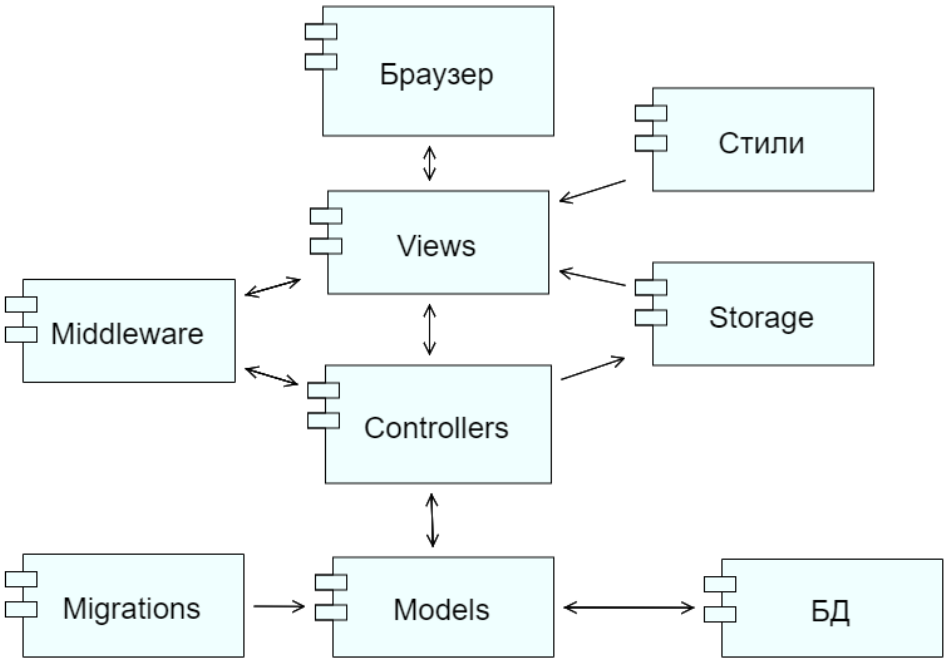

Рисунок 3.1 – Диаграмма компонентов приложения

3.2 Обоснование выбора технологии проектирования

3.2.1 PHP

PHP (Preprocessor Hypertext) — это язык программирования, который широко применяется для разработки веб-приложений и сайтов [3]. Он обладает рядом преимуществ, делающих его популярным среди разработчиков:

- Простота изучения: PHP относительно легко понять и освоить, особенно если у пользователя уже есть базовые знания HTML. Это делает его привлекательным выбором для начинающих программистов.

- Широкая поддержка: PHP имеет большое сообщество разработчиков, которое обеспечивает широкую поддержку и доступ к множеству ресурсов для обучения и решения проблем.

- Гибкость: PHP позволяет легко интегрировать различные технологии, такие как базы данных, XML, JSON и другие веб-технологии.

- Оптимизация производительности: PHP предлагает множество инструментов для оптимизации производительности, включая кэширование, минификацию кода и многопоточность.

- Поддержка MVC (Model-View-Controller): PHP поддерживает архитектуру MVC, которая помогает организовать код и упрощает разработку и поддержку больших приложений.

- Интеграция с другими языками программирования: PHP может быть использован вместе с другими языками программирования, такими как JavaScript, Python и Ruby. Это позволяет разработчикам использовать лучшие практики и инструменты из разных языков.

- Поддержка фреймворков: Существует множество популярных PHP-фреймворков, таких как Laravel, Symfony, CodeIgniter и Yii, которые предоставляют дополнительные инструменты и структуры для ускорения разработки.

3.2.2 MySQL

MySQL - это одна из самых популярных систем управления базами данных (СУБД). Это программа, которая позволяет создавать и управлять базами данных реляционного типа с использованием языка стандартизированных запросов (SQL). MySQL отличается своей доступностью как по финансовой стороне, так и по уровню сложности. Она проста в управлении и имеет развитое сообщество пользователей [1].

С практической точки зрения, система MySQL обладает рядом преимуществ:

- высокая скорость работы благодаря отказу от некоторых стандартов языка;

- простая установка и интуитивно понятный интерфейс, что делает её доступной даже для новичков;

- отсутствие ограничений по количеству пользователей;

- гибкость за счёт открытого исходного кода;

- богатый выбор сторонних инструментов (плагинов), что позволяет настроить систему под конкретные потребности;

- поддержка больших таблиц, содержащих до 50 миллионов строк;

- высокий уровень безопасности;

- широкий функционал, который позволяет использовать систему в различных сферах.

Эта СУБД считается надежной и стабильной уже долгие годы. Она поддерживает разные виды таблиц, быстро выполняет команды и может быть модифицирована под индивидуальные нужды проекта.

3.2.3 Blade

При разработке представлений можно использовать PHP либо язык шаблонизатора Blade. Код представлений будет более чистым, если использовать Blade [14].

Преимущества Blade:

- Простота использования. Для вывода данных достаточно использовать двойные фигурные скобки. Это позволяет избежать сложностей с выводом значений, экранированием строк и безопасностью.

- Безопасность. В отличие от PHP, где необходимо экранировать все выводимые данные, в Blade используются автоматические кавычки, что делает код более безопасным.

- Документированность. Blade хорошо документирован, что облегчает изучение и использование этого инструмента.

- Поддержка сообщества. Laravel имеет большое сообщество разработчиков, которые активно используют Blade и делятся своим опытом и решениями проблем.

- Расширяемость. Blade поддерживает пользовательские директивы, что позволяет расширять функциональность шаблонов без необходимости написания большого количества кода.

Выбор в пользу Blade добавляет дополнительный шаг в обработку шаблонов, т.к. все шаблоны Blade будут скомпилированы в PHP-код. Но благодаря кэшированию это не влияет на производительность приложения.

3.3 Архитектура Laravel

3.3.1 Паттерн MVC

Архитектура Laravel основана на шаблоне MVC (Model-View-Controller). Это вариант архитектуры, при котором компоненты программы делятся на три части:

- модель (model) предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность;
- представление (view) показывает пользователю эти данные;
- контроллер (controller) направляет данные от пользователя к системе и наоборот.

Модель MVC предназначена для разделения логических частей приложения и создания их независимо друг от друга.

3.3.2 Жизненный цикл запроса

Точкой входа всех запросов является файл public/index.php. Он загружает сгенерированное Composer определение автозагрузчика и извлекает экземпляр приложения Laravel из файла bootstrap/app.php.

В зависимости от типа запроса, он отправляется либо в ядро HTTP, либо в ядро Console.

Ядро HTTP находится в файле app/Http/Kernel.php и расширяет класс Illuminate/Foundation/Http/Kernel. Оно определяет массив bootstrappers — загрузчиков, которые будут выполняться перед обработкой запроса. Эти загрузчики настраивают обработку ошибок, настраивают логи, определяют окружение приложения и выполняют другие задачи, которые необходимо выполнить перед фактической обработкой запроса.

Ядро HTTP также определяет список HTTP-middleware, через которые должны пройти все запросы, прежде чем они будут переданы приложению. Эти middleware обрабатывают чтение и запись HTTP-сессии, определяют, находится ли приложение в режиме обслуживания, проверяют CSRF-токены и многое другое.

Одним из наиболее важных действий по начальной загрузке ядра является загрузка сервис-провайдеров — поставщиков услуг приложения. Сервис-провайдеры несут ответственность за начальную загрузку различных компонентов фреймворка, таких как базы данных, очереди, компоненты валидации и маршрутизации. Все сервис-провайдеры приложения настраиваются в массиве providers файла config/app.php.

После создания экземпляров провайдеров для них будет вызываться метод register. Как только все провайдеры будут зарегистрированы, для каждого из них будет вызван метод boot. Метод boot() в Laravel используется для определения кода, который должен быть выполнен при регистрации поставщика услуг. Этот метод обычно используется для регистрации пользовательских привязок, конфигураций и слушателей событий.

Одним из наиболее важных сервис-провайдеров в Laravel-приложении является App/Providers/RouteServiceProvider. Этот сервис-провайдер загружает файлы маршрутов, содержащиеся в каталоге routes.

После загрузки приложения и регистрации всех сервис-провайдеров запрос будет передан для обработки маршрутизатору. Маршрутизатор отправит запрос в маршрут или контроллер, а также запустит указанное middleware для данного маршрута. Если запрос проходит через все middleware соответствующего маршрута, будет выполнен метод маршрута или контроллера, а ответ, возвращённый методом маршрута или контроллера, будет отправлен обратно через цепочку middleware, давая приложению возможность изменить или проверить исходящий ответ.

Когда ответ возвращается через middleware, метод handle ядра HTTP возвращает объект ответа, а файл index.php вызывает метод send, который отправляет содержимое ответа в веб-браузер пользователя.

3.3.3 Сервис-провайдеры

Сервис-провайдеры являются центральным местом начальной загрузки всех возможностей Laravel. Под загрузкой подразумевается регистрация нужных сервисов, включая регистрацию привязок к служебному контейнеру, прослушивателей событий, middleware и даже маршрутов.

Все классы сервис-провайдеров хранятся в config/app.php. Провайдеры, имеющиеся там изначально, загружают основные компоненты Laravel, такие как почтовый рассыльщик, очередь, кэш и другие. Многие из этих классов провайдеров являются «отложенными». Это означает, что они будут загружаться не при каждом запросе, а только при необходимости предоставляемых ими услуг. Пользователь может создавать собственные сервис-провайдеры для своего приложения.

Все сервис-провайдеры расширяют класс Illuminate/Support/ServiceProvider. Класс провайдера содержит 2 метода — register и boot. Метод register предназначен для привязки к сервисному контейнеру. Внутри любого из методов класса-провайдера всегда доступно свойство $app, которое предоставляет доступ к сервисному контейнеру. Метод boot используется для расширения возможностей фреймворка, например, для создания новых blade-директив и запускается только после выполнения всех методов register.

3.3.4 Сервис-контейнеры

Сервис-контейнер или контейнер служб – это инструмент для управления зависимостями классов и выполнения внедрения зависимостей. Зависимости классов «вводятся» в класс через конструктор в виде аргументов или, в некоторых случаях, через методы-сеттеры. При создании класса или вызове методов фреймворк смотрит на список аргументов и, если нужно, создаёт экземпляры необходимых классов и сам подаёт их на вход конструктора или метода.

Все контроллеры регистрируются через сервис-контейнер, поэтому при получении класса контроллера из контейнера автоматически получаются все зависимости, указанные в аргументах конструктора и других методах контроллера.

3.3.5 Фасады

Фасады предоставляют «статический» интерфейс для классов, доступных в сервис-контейнере приложения. Все фасады Laravel расширяют базовый класс Illuminate/Support/Facades/Facade. Базовый класс Facade использует магический метод __callStatic(), чтобы делегировать вызовы из фасада объекту, извлеченному из контейнера.

Фасады предоставляют краткий и понятный синтаксис, который позволяет пользоваться функциями laravel без запоминания длинных названий классов, которые должны внедряться вручную. В дополнении к фасадам, Laravel предлагает множество глобальных вспомогательных функций, которые упрощают взаимодействие с общими функциями Laravel. Они доступны глобально и не требуют импортирования каких-либо классов для их использования. Многие вспомогательные функции выполняют те же задачи, что и соответствующие фасады.

3.4 Структура каталогов

Структура каталогов приложения включает следующие основные каталоги:

1. App — содержит подкаталоги Console, Exceptions, Http, Models и Providers. Каталог Console содержит пользовательские команды Artisan. Каталог Exceptions содержит обработчик исключений. Каталог Http содержит контроллеры, middleware и запросы форм. Каталог Models содержит классы моделей Eloquent. Каталог Providers содержит сервис провайдеры, подготавливающие приложение к входящим запросам.

2. Bootstrap — содержит файл app.php, загружающий фреймворк. Также в папке bootstrap находится папка cache, которая содержит сгенерирован-

ные фреймворком файлы для оптимизации производительности, например, кэш-файлы маршрутов и сервисов.

3. Config — содержит файлы конфигурации приложения.

4. Database — содержит миграции базы данных, фабрики моделей и данные для заполнения.

5. Public — содержит index.php, являющийся точкой входа для всех запросов, поступающих в приложение, и ссылку на storage.

6. Resources — содержит представления (view).

7. Routes — содержит все определения маршрутов для приложения.

8. Storage — разделён на подкаталоги storage/app, storage/framework и storage/logs. Каталог storage/framework используется для хранения кэшей. Storage/log хранит файлы журналов (логи) приложения. В каталоге storage/app/public хранятся изображения товаров.

9. Tests — содержит автоматические тесты.

10. Vendor — хранит зависимости, установленные через менеджер пакетов Composer.

3.5 Структура базы данных

В таблице 3.1 показана структура таблицы plant_categories.

Таблица 3.1 – таблица plant_categories

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer
	да	name	string
	да	description	text

В таблице 3.2 показана структура таблицы plant_genera.

Таблица 3.2 – таблица plant_genera

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string
	да	description	text

В таблице 3.3 показана структура таблицы product_types.

Таблица 3.3 – таблица product_types

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string

В таблице 3.4 показана структура таблицы plant_colors.

Таблица 3.4 – таблица plant_colors

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string

В таблице 3.5 показана структура таблицы climate_zones.

Таблица 3.5 – таблица climate_zones

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	zone_number	integer
	да	lower_temp_limit	integer
	да	upper_temp_limit	integer

В таблице 3.6 показана структура таблицы light_modes.

Таблица 3.6 – таблица light_modes

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string

В таблице 3.7 показана структура таблицы soils.

Таблица 3.7 – таблица soils

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string

В таблице 3.8 показана структура таблицы life_cycles.

Таблица 3.8 – таблица life_cycles

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	name	string

В таблице 3.9 показана структура таблицы plants.

Таблица 3.9 – таблица plants

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer
	да	name	string
	да	description	text
	да	quantity	integer
	да	price	decimal
	нет	image	text
	да	flower_diameter	decimal
foreign key	да	genus	string
foreign key	да	product_type	string

Продолжение таблицы 3.9

Тип ключа	Обязательность	Имя столбца	Тип данных
foreign key	да	life_cycle	string
foreign key	да	soil	string
foreign key	да	landing_place	string
foreign key	да	climate_zone	integer
foreign key	да	flower_color1	string
foreign key	нет	flower_color2	string
foreign key	нет	flower_color3	string
foreign key	да	leaf_color1	string
foreign key	нет	leaf_color2	string
foreign key	нет	leaf_color3	string

В таблице 3.10 показана структура таблицы users.

Таблица 3.10 – таблица users

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer
	да	name	string
	да	email	string
	нет	email_verified_at	timestamp
	да	password	string
	да	is_admin	tinyInteger

В таблице 3.11 показана структура таблицы orders.

Таблица 3.11 – таблица orders

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer

Продолжение таблицы 3.11

Тип ключа	Обязательность	Имя столбца	Тип данных
	да	status	integer
	нет	name	text
	нет	phone	text
	нет	address	text
	нет	email	string
foreign key	нет	user_id	integer

В таблице 3.12 показана структура таблицы order_plant.

Таблица 3.12 – таблица order_plant

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer
	да	plant_count	integer
foreign key	да	order_id	integer
foreign key	да	plant_id	integer

В таблице 3.13 показана структура таблицы plant_plant_category.

Таблица 3.13 – таблица plant_plant_category

Тип ключа	Обязательность	Имя столбца	Тип данных
primary key	да	id	integer
foreign key	да	plant_category_id	integer
foreign key	да	plant_id	integer

3.6 Список контроллеров

Каталог Controller содержит следующие контроллеры:

- Controller – родительский контроллер для остальных контроллеров;

- MainController – отвечает за отображение некоторых страниц, в том числе главной страницы сайта;

- HomeController – отвечает за отображение заказов пользователя;

- BasketController – отвечает за добавление товара в корзину и отображение корзины.

Каталог Controller также содержит подкаталоги Admin и Auth.

3.6.1 Контроллеры подкаталога Admin

Подкаталог Admin предназначен для контроллеров, предоставляющих методы для администратора сайта:

- CategoryController – ресурсный контроллер для управления категориями;

- OrderController – отвечает за отображение заказов у администратора;

- PlantController – ресурсный контроллер для управления товарами.

3.6.2 Контроллеры подкаталога Auth

Подкаталог Auth объединяет контроллеры для регистрации и авторизации:

- ConfirmPasswordController – предназначен для обработки подтверждения пароля;

- ForgotPasswordController – отвечает за отправку электронных писем для сброса пароля;

- LoginController – обрабатывает аутентификацию ;

- RegisterController – обрабатывает регистрацию новых пользователей;

- ResetPasswordController – содержит логику сброса пароля;

- VerificationController – отвечает за переход пользователей на другую страницу после верификации.

3.7 Список представлений

Каталог views содержит подкаталоги frontend, orders, auth, layouts и admin.

3.7.1 Представления подкаталога frontend

Подкаталог frontend содержит страницы для просмотра и заказа товаров:

- index.blade.php – главная страница со всеми товарами;
- categories.blade.php – страница категорий товаров;
- category.blade.php – страница категории;
- product.blade.php – страница товара;
- basket.blade.php – корзина;
- order.blade.php – страница оформления заказа;

3.7.2 Представления подкаталога orders

Подкаталог orders содержит страницы заказов пользователя:

- home.blade.php – страница заказов;
- order.blade.php – страница заказа.

3.7.3 Представления подкаталога auth

Подкаталог auth содержит страницы авторизации и регистрации:

- login.blade.php – страница аутентификации;
- register.blade.php – страница регистрации;
- verify.blade.php – страница верификации;

В этом подкаталоге также есть раздел passwords, объединяющий страницы для пароля:

- confirm.blade.php – форма для подтверждения пароля;
- email.blade.php – форма для ввода адреса электронной почты для сброса пароля;
- reset.blade.php – форма для создания нового пароля.

3.7.4 Представления подкаталога layouts

Подкаталог layouts содержит представления, используемые другими представлениями:

- master.blade.php – HTML-каркас для других представлений с подключёнными стилями и меню;
- card.blade.php – шаблон карточки товара.

3.7.5 Представления подкаталога admin

Подкаталог admin делится на разделы categories, orders и plants.

Раздел categories содержит страницы для работы с категориями:

- index.blade.php – страница категорий;
- show.blade.php – страница категории;
- form.blade.php – страница добавления или редактирования категории.

Раздел orders содержит страницы для просмотра заказов:

- index.blade.php – страница заказов;
- show.blade.php – страница заказа;

Раздел plants содержит страницы для работы с товарами:

- index.blade.php – страница товаров;
- show.blade.php – страница товара;
- form.blade.php – страница добавления или редактирования товара.

3.8 Стили приложения

Стили приложения можно найти по пути public/assets/css. Они представлены двумя файлами: bootstrap.min.css и starter-template.css. Эти стили подключаются в представлении layouts/master.blade.php. Файл bootstrap.min.css позволяет использовать возможности фреймворка Bootstrap для стилизации контента с помощью отдельных классов и компонентов. Одна из ключевых особенностей фреймворка – использование сетки, которая позволяет легко управлять распределением контента на разных размерах экранов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бретт, М. PHP и MySQL. Исчерпывающее руководство / М. Бретт. – Санкт-Петербург : Питер, 2016 г. – 544 с. – ISBN 978-5-496-01049-8. – Текст : непосредственный.

2. Веру, Л. Секреты CSS. Идеальные решения ежедневных задач / Л. Веру. – Санкт-Петербург : Питер, 2016 г. – 336 с. – ISBN 978-5-496-02082-4. – Текст : непосредственный.

3. Гизберт, Д. PHP и MySQL / Д. Гизберт. – Москва : НТ Пресс, 2013 г. – 320 с. – ISBN 978-5-477-01174-2. – Текст : непосредственный.

4. Дэкетт, Д. HTML и CSS. Разработка и создание веб-сайтов / Д. Дэкетт. – Москва : Эксмо, 2014 г. – 480 с. – ISBN 978-5-699-64193-2. – Текст : непосредственный.

5. Грофф, Дж. Р. SQL: полное руководство / Грофф Дж. Р., Вайнберг П. Н., Оппель Э. Дж. — 3-е изд. — Санкт-Петербург : Диалектика, 2020 г. — 960 с. - ISBN 978-5-907114-26-5. – Текст : непосредственный.

6. Шварц, Б. MySQL. Оптимизация производительности / Шварц Б., Зайцев П., Ткаченко В., Заводны Дж., Ленц А., Бэллинг Д. — 2-е изд. — Санкт-Петербург : Символ-Плюс, 2010 г. — 832 с. - ISBN 978-5-93286-153-0. – Текст : непосредственный.

7. Кингсбери, Н. Основы озеленения сада / Кингсбери Н. — 1-е изд. — Москва : Кладезь-Букс, 2003 г. — 208 с. - ISBN: 5-93395-034-3. – Текст : непосредственный.

8. Кривко, Н. П. Питомниководство садовых культур / Под редакцией Кривко Н. П. — Санкт-Петербург : Лань, 2015 г. — 368 с. - ISBN: 978-5-8114-1761-2. – Текст : непосредственный.

9. Коновалова, Т. Ю. Декоративные кустарники, или 1000 растений для вашего сада. Иллюстрированный справочник / Т. Ю. Коновалова, Н. А. Шевырева — Москва : Фитон+, 2004 г. — 192 с. - ISBN: 5-93457-070-6. – Текст : непосредственный.

10. Берд, Р. Цветущие деревья и кустарники: дизайн вашего сада и советы по выращиванию / Ричард Берд — Москва : АРТ-РОДНИК, 2003 г. — 159 с. - ISBN: 5-88896-089-6. – Текст : непосредственный.

11. Хессайон, Д. Г. Всё о декоративных растениях и кустарниках / Хессайон Д. Г. — 2-е изд.. — Москва : Кладезь-Букс, 2007 г. — 127 с. - ISBN: 978-5-93395-238-1. – Текст : непосредственный.

12. Ганичкина, О. А. Декоративные растения вашего сада: деревья, кустарники, цветы / Ганичкина О.А. — Москва : Эксмо, 2008 г. — 222 с. - ISBN: 978-5-699-28077-3. – Текст : непосредственный.

13. Марковский, Ю. Б. Хвойные растения для декоративного сада / Ю. Б. Марковский, И. В. Успенский. — Москва : Фитон XXI, 2016 г. — 232 с. - ISBN: 978-5-906171-91-7. – Текст : непосредственный.

14. Кириченко, А. В. Laravel для WEB-разработчиков. Практическое руководство по созданию профессиональных сайтов / А. В. Кириченко, Е. В. Дубовик. — Санкт-Петербург :Наука и Техника, 2021 г. - 432 с. - ISBN: 978-5-94387-726-1. – Текст : непосредственный.

15. Дронов, В. А. Laravel. Быстрая разработка современных динамических Web-сайтов на PHP, MySQL, HTML и CSS / В. А. Дронов. — Санкт-Петербург :БХВ-Петербург, 2018 г. - 755 с. - ISBN: 978-5-9775-3845-9. – Текст : непосредственный.