Частное учреждение профессиональная образовательная организация

ТЕХНИКУМ «БИЗНЕС И ПРАВО**»**

**ОТЧЕТ О ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

*практика по профилю специальности*

*ПП03 Участие в интеграции программных модулей*

в

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*наименование организации*

в период с 19 января 2024 г. по 29 февраля 2024 г.

|  |  |
| --- | --- |
|  | студента (ки) \_\_\_\_ курса очной формы обучения,  группы \_\_\_\_\_\_\_\_\_\_\_\_  специальности 09.02.03 Программирование в компьютерных системах  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *фамилия, имя, отчество*  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *фамилия, имя, отчество*  Дата\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

г. Белореченск

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
| Введение | 3 |
| 1. ОРГАНИЗАЦИОННЫЕ ВОПРОСЫ | 5 |
| 2. ПРОЦЕССЫ СОЗДАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ | 8 |
| 3. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПО | 29 |
| 4. ТЕСТИРОВАНИЕ И ОТЛАДКА ПО | 30 |
| 5. ИНТЕГРАЦИЯ СИСТЕМЫ | 34 |
| 6. ДОКУМЕНТИРОВАНИЕ | 37 |
| ЗАКЛЮЧЕНИЕ | 41 |
| СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ | 43 |
| ПРИЛОЖЕНИЕ |  |

ВВЕДЕНИЕ

Актуальность исследования – в современном мире магазины строительных материалов стали неотъемлемой частью повседневной жизни миллиардов людей по всему миру. Важность магазинов строительных материалов (включая интернет-магазины) для нашей жизни становится все более очевидной, поскольку они предоставляют возможность быстрого обеспечения строительных и ремонтных нужд, создания и улучшения жилищных условий. Интернет-магазины строительных материалов также играют важную роль, обеспечивая доступ к широкому ассортименту товаров и удобные условия покупки, что увеличивает удобство и доступность приобретения необходимых материалов и инструментов.

Многие малые компании и стартапы, работающие в сфере строительных материалов, сталкиваются с проблемой ограниченного бюджета и неспособности привлечь достаточное количество подписчиков и клиентов через традиционные жизненные методы маркетинга.

В такой ситуации на помощь может прийти разработка приложения для компании по строительным материалам. Создание приложения представляет собой стратегию, при которой предприятие может эффективно расширить свое онлайн-присутствие и привлечь новых клиентов. Этот метод может быть особенно полезен для компаний, желающих улучшить взаимодействие с клиентами и предоставить им удобный доступ к продуктам и услугам в онлайн-формате.

Например, успешное создание интернет-приложения для компании по строительным материалам позволило ей значительно увеличить свою клиентскую базу и улучшить взаимодействие с партнерами. Благодаря приложению клиенты могут легко ознакомиться с ассортиментом товаров, делать заказы и получать необходимую информацию о продукции и услугах. Это обеспечивает удобство и доступность взаимодействия как для клиентов, так и для компании, что способствует ее успешному развитию в онлайн-сфере.

Объект исследования – разработка информационной системы для компании строительных материалов, которая стремится к развитию в онлайн-сфере.

Предмет исследования – предметом исследования являются разработка и оптимизация стратегий продвижения компании в онлайн-пространстве, а также методы обеспечения увеличения онлайн-присутствия бренда и привлечения новых клиентов через создание приложения.

Цель исследования – рассмотреть и определить эффективные методы привлечения потенциальных партнеров и клиентов через различные заявки и просмотр информации о товарах в онлайн-среде, создав удобное приложение для компании по строительным материалам, которая стремится к расширению своего онлайн-присутствия.

Методы исследования – в рамках исследования будут использованы методы анализа данных о целевой аудитории, а также анализ существующих стратегий конкурентов. Кроме того, будет проведен ряд экспериментов для оценки эффективности различных подходов к привлечению партнеров и клиентов через интернет-приложение компании.

Задачи исследования:

* изучить особенности функционирования интернет-магазинов в сфере строительных материалов, включая методы привлечения клиентов и партнеров;
* провести обзор существующих интернет-магазинов, конкурирующих в данной нише, с целью изучения их стратегий и методов привлечения клиентов и партнеров;
* выбрать оптимальный метод реализации интернет-магазина для компании, обеспечивающий удобное взаимодействие с клиентами и партнерами;
* провести тестирование разработанной системы и осуществить необходимые корректировки и адаптации в соответствии с полученными результатами;

1. ОРГАНИЗАЦИОННЫЕ ВОПРОСЫ

При изучении организационных вопросов на предприятии ООО «Сириус» были рассмотрены основные структурные подразделения, такие как отделы бухгалтерии, закупок, логистики и продаж. Были изучены инструкции по охране труда, технике безопасности, схемы аварийных проходов и выходов, а также ознакомлены с пожарным инвентарем.

Организация локальной сети на предприятии также была рассмотрена в контексте обеспечения связи и обмена информацией между сотрудниками. Особое внимание уделялось правилам внутреннего распорядка и нормам охраны труда, включая технику безопасности при работе с вычислительной техникой.

В компании были изучены и применены соответствующие инструкции, правила и нормы, касающиеся охраны труда и техники безопасности:

1. Инструкция по охране труда на объекте предприятия (утверждено 15.03.2017);
2. Правила техники безопасности при обращении с тяжелыми строительными материалами (утверждены 13.05.2017);
3. Нормы по оформлению и эксплуатации рабочих мест в магазине строительных материалов (утверждены 07.06.2017);
4. Инструкция по пожарной безопасности и эвакуации на предприятии (утверждена 05.05.2017);
5. Правила внутреннего распорядка для сотрудников магазина (утверждены 20.11.2023);

В рамках отчета была представлена структурная схема предприятия ООО «Сириус», отражающая организационную структуру и взаимосвязи между отделами. Также был предоставлен перечень изученных инструкций, правил и норм, а также технических характеристик ПК и периферийных устройств на рабочем месте. Была описана организация локальной сети на предприятии и представлен перечень установленного программного обеспечения на рабочих местах с указанием версий.

ООО «Сириус» специализируется на предоставлении широкого ассортимента строительных материалов для дома, сада и ремонтных работ. В компании частное руководство ответственно осуществляет общее управление и разработку стратегии развития. Организационная структура включает несколько ключевых отделов.

Отдел бухгалтерии является одним из важнейших подразделений предприятия. Его основная задача - обеспечение финансового благополучия компании. Отдел бухгалтерии отвечает за ведение учета финансовых операций, подготовку финансовой отчетности, налоговую отчетность и контроль исполнения бюджета предприятия.

Отдел закупок играет ключевую роль в обеспечении постоянного наличия необходимых товаров в продаже. Специалисты отдела проводят анализ рынка строительных материалов, взаимодействуют с поставщиками, оценивают предложения и заключают договоры. Важной частью работы отдела является также поддержание оптимальных запасов товаров на складе.

Логистический отдел отвечает за оптимизацию и управление потоком материалов и информации внутри предприятия. Его задачи включают в себя контроль за уровнем запасов, организацию поставок и распределение товаров на складе. Логисты также разрабатывают эффективные схемы логистики поставок для обеспечения своевременного выполнения заказов.

Отдел продаж занимается прямым взаимодействием с клиентами и продвижением товаров. Специалисты отдела консультируют клиентов, предоставляют информацию о продукции и услугах, разрабатывают маркетинговые стратегии и управляют клиентской базой. Они также отвечают за обработку заказов и координацию с отделом закупок и логистикой для обеспечения своевременной поставки товаров.

На момент прохождения практики в ООО «Сириус» работает команда из примерно 50 сотрудников, каждый из которых обладает высоким профессионализмом и регулярно повышает свою квалификацию.

При разработке проекта приложения для магазина строительных материалов на фреймворке Laravel PHP в компании использовались определенные периферийные устройства и программное обеспечение. Ниже представлено подробное описание технических характеристик используемых периферийных устройств, таких как мониторы, клавиатуры, мыши, а также перечень установленного программного обеспечения, включая операционную систему, среду разработки, серверное ПО и инструменты разработки.

Технические характеристики ПК и периферийных устройств на рабочем месте:

1. Процессор: Intel Core i7-10700K;
2. Оперативная память: 16 ГБ DDR4;
3. Жесткий диск: SSD 512 ГБ;
4. Принтер: HP LaserJet Pro MFP M28w;
5. Сетевое оборудование: маршрутизатор TP-Link Archer C7, коммутатор D-Link DES-1016D;
6. Монитор: Dell UltraSharp U2419H, 24 дюйма, разрешение экрана 1920x1080 пикселей, IPS-панель;
7. Клавиатура: проводная клавиатура Logitech K120;
8. Мышь: Logitech MX Master 3, беспроводная мышь;

В ходе работы над проектом использовалось следующее программное обеспечение:

1. Операционная система: Windows 10 Pro (версия 10.0.19042);
2. Фреймворк PHP Laravel (версия 11.0.7);
3. Интегрированная среда разработки для PHP PhpStorm (версия 2023.3.5);
4. Система управления версиями проекта Git (версия 2.35);
5. Менеджер зависимостей проекта PHP Composer (версия 2.7.2);
6. Среда выполнения JavaScript-программ Node.js (версия 16.13);

2. ПРОЦЕССЫ СОЗДАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Современный бизнес в условиях цифровой трансформации требует активного онлайн-присутствия для успешного развития и конкуренции на рынке. Однако компания, в рамках которой была проведена производственная практика, столкнулась с серьезной и главной проблемой – отсутствием интернет-магазина, что ограничивает ее потенциал и возможности на современном рынке. Отсутствие онлайн-присутствия препятствует компании в привлечении новых клиентов, увеличении объемов продаж и улучшении обслуживания существующих клиентов.

В условиях динамично развивающегося цифрового рынка, предприятие столкнулось с недостатком онлайн-присутствия, что затрудняет её конкуренцию с другими участниками рынка. Отсутствие интернет-магазина приводит к упущению возможностей для увеличения объемов продаж, привлечения новых клиентов и повышения уровня обслуживания. В современных условиях потребители все больше предпочитают совершать покупки онлайн, что делает важным наличие функционального и удобного интернет-магазина для компании.

Одним из ключевых аспектов, ограничивающих рост компании, является ограниченный охват аудитории. Отсутствие интернет-магазина снижает доступность продукции для широкой аудитории потребителей, что препятствует привлечению новых клиентов и увеличению объемов продаж. В ситуации, когда покупатели все больше предпочитают делать покупки онлайн из-за удобства и доступности, ограниченный охват аудитории становится серьезным препятствием для развития бизнеса.

Кроме того, недостаточная адаптация к требованиям рынка является серьезным вызовом для компании. В условиях быстро меняющейся цифровой среды компания, не имеющая интернет-магазина, рискует оставаться в отстающих от конкурентов. Отсутствие возможности онлайн-продаж ограничивает компанию в применении современных маркетинговых инструментов, таких как таргетированная реклама, контент-маркетинг и социальные сети, что снижает ее конкурентоспособность и способствует потере доли рынка.

Таким образом, для успешного развития и выхода на новые рынки компания нуждается в автоматизации процесса создания и запуска интернет-магазина с нуля. Это позволит расширить клиентскую базу, увеличить объемы продаж и повысить уровень сервиса, что в конечном итоге приведет к укреплению позиций компании на рынке.

На данном этапе необходимо первым делом описать примерную архитектуру разрабатываемой системы. Для того, чтобы сделать систему интернет-магазина, будет использоваться клиент-серверная архитектура. В клиент-серверной архитектуре имеется три звена:

1. Представление данных — на стороне клиента;
2. Прикладной компонент — на выделенном сервере приложения, здесь происходит вся бизнес-логика, которая в дальнейшем будет взаимодействовать с клиентом;
3. Управление ресурсами — сервер БД (базу данных), который и представляет запрашиваемые данные;

Для данного проекта выбрана эта архитектура, так как у нее есть ряд преимуществ перед другими архитектурами:

1. Высокую степень гибкости и масштабируемости;
2. Высокую безопасность;
3. Высокую производительность (т. к. задачи приложения распределены между серверами);

На рисунке 3 представлена структура клиент–серверной архитектуры:

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 – Клиент–серверная архитектура

Следующий шагом для успешного функционирования интернет-магазина необходимо обеспечить высокий уровень надежности, безопасности и удобства использования для всех категорий пользователей. Важно учитывать основные требования к программному обеспечению, опыт и заранее продуманные аспекты разработки.

Надежность и стабильность — обеспечивает стабильную работу системы в любых условиях, даже при возможных сбоях серверов или баз данных. Это гарантирует, что пользователи могут использовать приложение без прерываний.

1. Приложение должно обеспечивать стабильную работу в любых условиях, включая возможные сбои в работе серверов или баз данных;
2. Обработка ошибок должна быть организована таким образом, чтобы пользователи получали информативные сообщения об ошибках, а администраторы имели возможность мониторинга состояния системы и оперативного реагирования на возможные проблемы;

Безопасность — защищает конфиденциальность информации пользователей и предотвращает несанкционированный доступ к данным.

1. Приложение должно обеспечивать высокий уровень безопасности для защиты конфиденциальной информации пользователей;
2. Должны быть реализованы меры по защите от несанкционированного доступа, атак на основе внедрения кода, а также другие виды атак на безопасность данных;
3. Пароли и другие конфиденциальные данные пользователей должны храниться в зашифрованном виде;

Удобство использования — создает удобный и интуитивно понятный интерфейс для всех категорий пользователей, что повышает удовлетворенность пользователей и облегчает работу с приложением.

1. Интерфейс приложения должен быть интуитивно понятным и удобным для всех категорий пользователей;
2. Все функции приложения должны быть легко доступны и понятны для пользователей;
3. Навигация по сайту должна быть логичной и понятной, а элементы управления должны быть расположены таким образом, чтобы облегчить пользовательский опыт;

Обработка ошибок и валидация данных — обеспечивает надежную обработку ошибок и валидацию данных на всех этапах работы приложения, что улучшает опыт использования и защищает систему от атак.

1. Приложение должно активно обрабатывать ошибки и осуществлять валидацию данных на всех этапах работы;
2. Ошибки сервера, такие как коды состояния HTTP 503, 403, 404, должны обрабатываться с предоставлением информативных сообщений пользователю;
3. Валидация данных ввода пользователей должна предотвращать возможные ошибки и атаки на безопасность;

Расширяемость и поддержка - обеспечивает легкость добавления новых функций и поддержку в долгосрочной перспективе, что позволяет приложению оставаться актуальным и конкурентоспособным.

1. Приложение должно быть легко расширяемым и поддерживаемым в долгосрочной перспективе;
2. Архитектура приложения должна быть модульной и гибкой, чтобы облегчить добавление новых функций и внесение изменений в существующий код;
3. Код приложения должен быть хорошо документирован, чтобы обеспечить удобство поддержки и развития программного обеспечения;

Сервис по онлайн накрутке социальных сетей предназначен для предоставления клиентам возможности создавать свои задания для накрутки своей активности в социальных сетях. В разработку входит создание удобной и интуитивно понятной онлайн-платформы для клиентов и администраторов сервиса.

В первую очередь, предусмотрено использование капчи и другим мер против спама с целью защиты от атак ботов и предотвращения DDoS-атак, что повышает общую безопасность веб-ресурса. Для хорошей безопасности информационной системы должно быть приобретено домен, хостинг, и SSL-сертификат – это всё поможет обеспечить безопасное соединение и общение между клиентом и сервером.

В информационной системе должна быть предусмотрена отдельная административная часть. Данная часть системы будет доступна только авторизованному пользователю, который имеет больше привилегий, чем обычный пользователь.

Из функциональных требований исходят следующие пункты.

Управление сервисами и услугами для каждого сервиса:

* Создание, редактирование и удаление информации о сервисах (название, изображение);
* Создание, редактирование и удаление информации об услугах для конкретного сервиса (название, цены);

Управление транзакциями:

* Возможность пополнять средства на внутренний баланс системы;
* Просмотр своих транзакций, а также просмотр всех транзакций в системе (административная часть);
* Возможность отменять транзакции разных типов у пользователей;

Управление пользователями:

* Авторизация и аутентификация пользователей;
* Изменение информации о себе (текстовые данные, изображение);
* Возможность просмотра информации пользователей администраторами;
* Возможность добавлять, редактировать и удалять пользователей от имени администратора;

Интерфейс для пользователей:

* Просмотр посадочной страницы, своих транзакций, созданных заданий, информации о себе, а также страница формирования нового заказа (задания);

Дизайн системы должен быть интуитивно понятный и удобный для пользователей, соответствуя их потребностям и ожиданиям. Создание прототипов и макетов позволяет заказчику и разработчикам предварительно визуализировать функционал системы и согласовать детали дизайна.

Требования к системе разработаны в соответствии со стандартом качества программного обеспечения ISO 9126:2001, описывающим многоуровневую модель характеристик качества и соответствующий им набор атрибутов.

Требования к информационной системе представлены в таблице 1:

Таблица 1 – Требования к системе

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибуты | Метки | | Требования |
| Удобство | | | |
| Простота использования | Среднее время, необходимое  пользователю, чтобы найти  необходимый элемент в программе | | Поиск отдельного элемента интерфейса не должен занимать более 1 секунды |
| Привлекательность | Соответствие интерфейса  требованиям | | 99% элементов должны быть работоспособны для  пользователя |
| Обучаемость | Показатель, затрачиваемый  пользователями на обучение  работе | | Пользователь должен изучить работу ПО за первые 3-5 минут  использования |
| Производительность | | | |
| Временная эффективность | Время выполнения компонента программы | - Приложение должно  обслуживать любую функцию не дольше 1 сек  (не включая задержки в сети); | |
| Эффективность использования ресурсов | Объемы ресурсов требуемых для выполнения задач | - Постоянная память,  используемая программой – не более 120МБ;  - Приложение должно потреблять не более 3МБ  памяти на каждый неактивный сеанс с  пользователем; | |
| Удобство установки | Легкость установки | Необходима установка:  XAMPP, MySQL, PhpMyAdmin | |
| Переносимость | | | |
| Способность к сосуществованию | Способность ПО сосуществовать с другими  программами в общем окружении | Программа не должна  занимать большие объемы памяти (максимум 1ГБ) | |
| Удобство замены другого ПО данным | Совместимость структур данных | - Система должна быть  разработана по общим  принципам и не требовать навыков от пользователя;  - Должна поддерживать  внедрение новых версий;  - Система должна быть применима вместо других  программных систем для решения тех же задач в определенном окружении; | |

Требования к численности и квалификации персонала системы и режиму его работы:

1. В системе существуют 2 группы пользователей: пользователи и администраторы. Пользователи могут просматривать свои настройки профиля, редактировать профиль, просмотреть транзакции, пополнять баланс, просматривать свои созданные задания и создавать новые задания.
2. Администратор выполняет следующие функции: просмотра, занесение данных, редактирование, удаление данных, и отмена операций. Данный функционал относится к работе с пользователями, заданиями (заказам), транзакциями, сервисами и услугами сервиса.
3. Пользователи системы должны обладать базовыми навыками взаимодействия с компьютером и использования веб-ресурсов. Опыт работы с персональными компьютерами или устройствами, доступ в Интернет и уверенное понимание основных принципов работы веб-интерфейсов будет полезным для комфортного использования системы.

Кроме того, система предоставляет возможность быстрого и удобного доступа к актуальной информации о сервисах, услугах, транзакциях и заданий, обеспечивая удовлетворение потребностей как пользователей, так и администраторов.

В таблице 2 представлены требования к надежности.

Таблица 2 – Требования к надёжности

|  |  |  |
| --- | --- | --- |
| Атрибуты | Метрики | Требования |
| Надежность | | |
| Устойчивость к отказу | Коэффициент аварийных  отказов | - Не менее 90% ошибок  в программе должны  обрабатываться без  экстренного завершения;  - Обеспечение бесперебойного питание  активного сетевого  оборудования;  - Постоянная работа сервера; |
| Способность к восстановлению | Отношение количества числа успешного восстановление работы системы к ее аварийному завершению | - Восстановление  работоспособности  системы не должно превышать 5 минут; |
| Продолжение таблицы 2 | | |
| 1 | 2 | 3 |
| Уровень зрелости | Среднее время работы без сбоев | - Система должна  работать 24 часа в сутки;  - Система должна оповещать пользователя при некорректных действиях в системе;  - Данные, которые вводит пользователь должны сохраняться; |

Средняя доступность приложения должна составлять не менее 99%

Среднее время между сбоями — это среднее время, за которое компонент или модуль может выполнять свои функции без перерыва.

Измеряется от начала работы до момента следующего сбоя.

Среднее время работы без сбоев должно составлять не менее 1500 часов.

В таблице 3 представлен перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

Таблица 3 – Перечень и критерии отказов для каждой функции

|  |  |
| --- | --- |
| Функции | Критерии отказа |
| Отображение данных | Отсутствие соединения с БД |
| Взаимодействие с товарами | Отсутствие соединения с БД |
| Поиск | Отсутствие соединения с БД |
| Добавление | Отсутствие соединения с БД |
| Редактирование | Отсутствие соединения с БД |
| Удаление | Отсутствие соединения с БД |
| Отмена транзакций | Отсутствие соединения с БД |
| Авторизация и регистрация | Отсутствие соединения с БД |

Для нормального функционирования информационной системы на базе PHP Laravel требуется обеспечить соответствующую инфраструктуру:

* наличие сервера для хостинга приложения;
* персональные устройства для работы с приложением;
* надежное интернет-соединение для доступа к системе;
* программа XAMPP для локальной разработки и тестирования приложения;
* обязательное создание резервных копий информации для ее сохранности и возможности восстановления при необходимости;
* требование к наличию сервера базы данных MySQL для хранения данных приложения;
* мониторинг производительности и доступности системы для быстрого выявления и устранения проблемных ситуаций;
* настройка системы мониторинга безопасности для обнаружения и предотвращения угроз безопасности приложения;
* регулярное обновление программных компонентов и фреймворка для обеспечения безопасности и работоспособности приложения;

При анализе предметной области разрабатываемой системы были выделены следующие сущности, на основе которых будет разрабатываться база данных системы:

* Пользователи;
* Транзакции пользователей;
* Баланс пользователей;
* Сервисы (социальные сети);
* Услуги сервисов;
* Заказы (задания);

База данных информационной системы должна состоять из следующих связанных таблиц:

1. Users - содержащая в себе информацию о пользователях системы;
2. Balance Transactions - содержащая в себе информацию о транзакциях;
3. User Balances - содержащая в себе информацию о текущем балансе пользователя;
4. Services - содержащая в себе информацию о сервисах;
5. Service Items - содержащая в себе информацию о услугах для конкретного сервиса;
6. Orders – содержит в себе информацию об заказах (заданиях) пользователей;

Данные о пользователях представлены в таблице Users:

Таблица 4 – Данные о пользователях

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор пользователя | Первичный ключ |
| login | VARCHAR | 255 | Логин пользователя | NOT NULL |
| name | VARCHAR | 255 | Имя пользователя | NOT NULL |
| email | VARCHAR | 255 | Почта пользователя | NOT NULL |
| password | VARCHAR | 255 | Хешированный пароль пользователя | NOT NULL |
| avatar | VARCHAR | 255 | Уникальный файл аватар пользователя | - |
| role | ENUM | 255 | Роль пользователя | NOT NULL |

Данные о транзакциях системы будут размещаться в таблице BalanceTransactions

Таблица 5 – Данные о транзакциях системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор транзакции | Первичный ключ |
| user\_id | BIGINT | 20 | Идентификатор пользователя | Внешний ключ |
| amount | INT | 20 | Сумма транзакции | NOT NULL |
| type | VARCHAR | 255 | Тип транзакции | NOT NULL |
| Cancelled | BOOLEAN | - | Отмена транзакции | - |

Данные о текущем балансе пользователя будут хранится в таблице UserBalances

Таблица 6 – Данные о текущем балансе пользователей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор  баланса | Первичный ключ |
| user\_id | BIGINT | 20 | Идентификатор пользователя | NOT NULL |
| balance | INT | 20 | Баланс пользователя | NOT NULL |

Данные о сервисах хранятся в таблице Services

Таблица 7 – Данные о сервисах

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор  сервиса | Первичный ключ |
| name | VARCHAR | 255 | Имя сервиса | NOT NULL |
| logo | VARCHAR | 255 | Уникальный файл логотипа сервиса | NOT NULL |

Данные о услугах сервиса будут находится в таблице ServiceItems

Таблица 8 – Данные о услугах сервиса

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор услуги | Первичный ключ |
| service\_id | BIGINT | 20 | Идентификатор сервиса | Внешний ключ |
| name | VARCHAR | 255 | Имя услуги | NOT NULL |
| price\_low | INT | 20 | Цена за низкое качество | NOT NULL |
| price\_medium | INT | 20 | Цена за среднее качество | NOT NULL |
| price\_high | INT | 20 | Цена за высокое качество | NOT NULL |

Данные о всех заказах (заданиях) пользователей должны располагаться в таблице Orders

Таблица 9 – Данные о заданиях пользователей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Тип | Размер | Комментарий | Ограничение |
| id | BIGINT | 20 | Идентификатор заказа | Первичный ключ |
| user\_id | BIGINT | 20 | Идентификатор пользователя | Внешний ключ |
| service\_id | BIGINT | 20 | Идентификатор сервиса | Внешний ключ |
| service\_item\_id | BIGINT | 20 | Идентификатор услуги сервиса | Внешний ключ |
| source\_link | VARCHAR | 255 | Ссылка на ресурс | NOT NULL |
| quantity | VARCHAR | 255 | Количество элементов накрутки | NOT NULL |
| total\_price | INT | 20 | Итоговая стоимость | NOT NULL |
| status | VARCHAR | 255 | Статус заказа | NOT NULL |

Сайт должен быть структурирован на четко взаимосвязанные разделы, что сможет обеспечивать удобную навигацию и легкость в поиске информации для пользователей.

Навигационное меню, посадочная страница и панель управления разработаны с учетом удобства пользователей, обеспечивая легкость перехода между разделами и доступ к основной информации.

Нефункциональные требования играют важную роль в обеспечении эффективной работы и безопасности сервиса. Интерфейс должен быть интуитивно понятным для всех пользователей и адаптивным для корректного отображения на различных устройствах. Безопасность приложения несет важное значение и предполагает реализацию системы аутентификации и авторизации для администраторов и пользователей, а также защиту от уязвимостей веб-приложения, включая SQL-инъекции, XSS и CSRF атаки. Производительность включает оптимизацию загрузки изображений и данных для быстрого отображения страниц каталога, а также минимизацию времени ответа сервера на запросы клиентов.

Нефункциональные требования выполняются для обеспечения безопасности, производительности и удобства использования электронного каталога цветочного магазина:

Интерфейс:

* Интуитивно понятный интерфейс для всех категорий пользователей;
* Адаптивный дизайн для корректного отображения на различных устройствах;

Безопасность:

* Реализация системы аутентификации и авторизации для администраторов и пользователей;
* Защита от уязвимостей веб-приложения, таких как SQL-инъекции, CSRF-атак и XSS-атаки;

Производительность:

* Оптимизация загрузки изображений и данных для быстрого отображения страниц каталога;
* Минимизация времени ответа сервера на запросы клиентов;

Инновационные технологии, безупречная функциональность и надежная безопасность — вот то, что делает этот электронный каталог цветочного магазина неповторимым. Позвольте погрузиться в детали технических требований, определяющих основу его функционирования.

3. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПО

При разработке приложения для магазина строительных материалов, компания опиралась на множество различных инструментов и технологий, которые обеспечивали не только высокое качество, но и эффективность процесса разработки.

Один из ключевых языков, используемых в проекте — это PHP. PHP выбран в качестве основного языка программирования благодаря своей широкой распространенности в веб-разработке и мощным функциональным возможностям. Этот язык позволяет быстро и гибко создавать веб-приложения, обрабатывать формы, работать с базами данных и многое другое.

Для ускорения процесса разработки и создания более надежного и масштабируемого кода был выбран фреймворк Laravel. Laravel — это один из самых популярных и мощных фреймворков PHP, предоставляющий разработчикам готовые решения для множества задач, таких как маршрутизация, аутентификация, работа с базами данных и многое другое. Использование Laravel значительно упростило и ускорило разработку приложения, а также обеспечило его высокую надежность и безопасность.

Помимо использования фреймворка Laravel, в проекте также активно применялся шаблонизатор Blade. Blade является интегрированным в Laravel шаблонизатором, который предоставляет удобные и мощные средства для работы с HTML и PHP кодом. Благодаря Blade имеется возможность создавать динамические представления, интегрировать PHP код в HTML шаблоны, использовать условия и циклы для управления данными, а также расширять функциональность собственными директивами. Использование Blade значительно упростило процесс создания пользовательского интерфейса приложения, делая его более гибким, понятным и легко поддерживаемым.

Один из важных аспектов серверной части — это работа с базой данных. Для хранения и управления данными в проекте использовалась реляционная база данных MySQL. MySQL — это мощная и надежная система управления базами данных, которая широко используется в веб-разработке благодаря своей производительности и надежности. В рамках проекта была спроектирована оптимизированная структура базы данных, обеспечивающая эффективное хранение и быстрый доступ к данным о клиентах, продукции, отзывах, заявках и другой информации.

Для управления зависимостями проекта и установки сторонних библиотек использовался Composer. Composer — это удобный инструмент для управления зависимостями в проектах на PHP, который позволяет быстро и легко добавлять, удалять и обновлять пакеты, необходимые для работы приложения.

В процессе разработки дизайна и пользовательского интерфейса приложения использовался инструмент Figma. Это мощный и удобный инструмент для создания дизайна и прототипирования интерфейса, который позволяет командам дизайнеров и разработчиков работать над проектом в реальном времени, обмениваться идеями и вносить изменения на лету. Использование Figma позволило создать современный и удобный интерфейс приложения, который соответствует требованиям пользователей и айдентике, и эргономичности компании.

Для написания и отладки кода на PHP использовалась интегрированная среда разработки PhpStorm. PhpStorm предоставляет разработчикам удобное и эффективное рабочее пространство для создания веб-приложений. Благодаря мощным инструментам автодополнения, интегрированному отладчику и поддержке широкого спектра PHP-фреймворков, PhpStorm значительно упрощает процесс разработки, позволяя сосредоточиться на создании качественного и надежного кода.

Все эти инструменты и технологии в совокупности обеспечили успешную разработку и реализацию приложения для магазина строительных материалов, которое предоставляет пользователям удобный и надежный инструмент для поиска, выбора и покупки строительных материалов.

При проектировании базы данных для приложения было уделено значительное внимание разработке хорошо продуманной структуре данных. Было потрачено много времени на анализ требований к приложению и выработку оптимальной схемы БД, которая обеспечивала эффективное хранение и управление информацией.

В ходе проектирования были учтены различные сценарии использования приложения и специфика работы с данными. Были проведены обсуждения с заказчиком для определения основных сущностей, их атрибутов и взаимосвязей. Для каждой таблицы были выбраны определенные типы полей, учитывая не только типы данных, но и их размеры, ограничения и связи между таблицами.

Проектирование базы данных также включало в себя работу над обеспечением целостности данных и оптимизацией производительности. Были предусмотрены необходимые ограничения целостности, индексы и другие механизмы для обеспечения быстрого доступа к данным и предотвращения возможных ошибок при работе с базой данных.

В результате тщательного проектирования базы данных удалось создать структуру, которая эффективно поддерживает функциональность приложения, обеспечивает надежное хранение данных и удовлетворяет потребностям как заказчика, так и конечных пользователей.

При разработке серверной части приложения для магазина строительных материалов на базе фреймворка Laravel, было уделено значительное внимание обеспечению эффективного взаимодействия с клиентской частью. В основе этой части лежат модели данных, миграции и контроллеры, обеспечивающие управление данными и бизнес-логикой приложения.

При разработке серверной части приложения использовались следующие ключевые аспекты фреймворка:

* Модели данных - описывают структуру данных и обеспечивают удобное взаимодействие с базой данных;
* Миграции - обеспечивают автоматизацию создания и изменения структуры базы данных;
* Контроллеры - обрабатывают запросы от клиентской части и управляют бизнес-логикой приложения;
* Маршрутизация - определяет, какие HTTP запросы должны быть направлены на какие контроллеры;

При обработке запросов от клиентской части, сервер возвращает данные в формате JSON. Это обеспечивает гибкость и удобство при передаче информации между клиентом и сервером. JSON ответы содержат различные данные, такие как информацию о продуктах, онлайн-заявках, отзывах, пользователях и других сущностях приложения, а также проработанные сообщения об ошибках или подтверждения об успешном выполнении операций.

Клиентская часть приложения разработана с учетом удобства использования для конечного пользователя. В ней предусмотрены различные экранные формы, которые представляют собой интерфейс для взаимодействия пользователя с приложением. Эти формы включают в себя, например, страницы для оформления заказов, ввода личных данных, отображения списка товаров и многое другое.

Для обеспечения безопасности и производительности серверная часть приложения применяет ряд мероприятий:

1. Валидация данных: перед отправкой данных на сервер производится их валидация на стороне клиента, чтобы предотвратить отправку некорректной информации.
2. Обработка ошибок: в случае возникновения ошибок при выполнении операций на стороне сервера или при обмене данными, клиентская часть приложения обеспечивает корректное отображение сообщений об ошибках для пользователя.
3. Обработка исключений: при возникновении исключительных ситуаций, таких как недоступность сервера или проблемы с сетевым соединением, клиентская часть приложения предпринимает соответствующие действия для информирования пользователя и восстановления работы приложения.

2.2 Тестирование веб-приложения

После завершения разработки кода, дизайн, и интеграции всех компонентов приложения, наступает этап тестирования, на котором проверяется работоспособность, корректность и соответствие функционала веб-приложения заявленным требованиям и ожиданиям пользователей.

Тестирование веб-приложения включает в себя различные виды тестирования, такие как функциональное тестирование, тестирование пользовательского интерфейса, тестирование производительности, тестирование безопасности и другие. Каждый вид тестирования направлен на проверку определенных аспектов функциональности и качества веб-приложения.

В случае учебной практики будет использоваться функциональное тестирование и тестирование пользовательского интерфейса (UI). Будет применено 35 тест-кейсов, которые проверят работоспособность информационной системы. Суть данного тестирования заключается в проверке работы всех обязательных функций сайта и его корректное отображение во всех актуальных браузерах.

Таблица 10 - Тест-кейсы, пройденные в ходе тестирования системы

|  |  |  |  |
| --- | --- | --- | --- |
| № Тест-кейса | Название проверки | Результат | Замечание и рекомендации |
| 1 | 2 | 3 | 4 |
| Кроссбраузерное тестирование | | | |
| 1 | Приложение корректно отображается и функционирует во всех актуальных браузерах (Brave, Microsoft Edge, Chrome, Firefox, Safari, Opera, Яндекс Браузер) | Положительно |  |
| 2 | Приложение корректно отображается и функционирует во всех актуальных версиях браузеров | Положительно |  |
| 3 | Корректная обработка тайм-аутов при работе с интерфейсом | Положительно |  |
| 4 | Работоспособность при удалении cookies во время использования приложения | Положительно |  |
| 5 | Работоспособность при удалении cookies после использования приложения | Положительно |  |
| Работа с формами | | | |
| 6 | Обязательные поля должны быть заполнены до подтверждения ввода данных. | Положительно |  |
| 7 | Обязательные поля должны быть явно обозначены. | Положительно |  |
| 8 | Если пользователь нажал кнопку подтверждения ввода данных с незаполненными обязательными полями, эти поля должны быть подсвечены и отображено сообщение об ошибке | Положительно |  |
| Продолжение таблицы 10 | | | |
| 1 | 2 | 3 | 4 |
| 9 | Сообщение об ошибке не отображается, если не заполнены необязательные поля | Положительно |  |
| 10 | Текстовые поля допускают ввод специальных символов | Положительно |  |
| 11 | При вводе количества знаков меньше минимального отображается сообщение | Положительно |  |
| 12 | При вводе количества знаков больше максимального отображается сообщение об ошибке | Положительно |  |
| 13 | Если обязательное поле заполнено не полностью, отображается сообщение об ошибке | Отрицательно | Доработать валидацию для полей |
| 14 | Сообщение об ошибке отображается при попытке ввести недопустимые символы (например, ввод текста в поле, принимающее только числа) | Положительно |  |
| 15 | Нельзя отменить повторно отменённую операцию | Положительно |  |
| 16 | Пополнять баланс нельзя на отрицательную сумму | Положительно |  |
| Авторизация и работа с профилем | | | |
| 17 | Создайте новый аккаунт | Положительно |  |
| 18 | Авторизуйтесь с существующим аккаунтом | Положительно |  |
| 19 | При вводе неверного пароля отображается сообщение об ошибке и ссылка на страницу восстановления пароля | Отрицательно | Дать возможность пользователю восстановить пароль |
| 20 | Отредактируйте профиль пользователя | Положительно |  |
| 21 | Загрузите в профиль аватар/картинку | Положительно |  |
| 22 | Отредактируйте аватар/картинку | Положительно |  |
| 23 | При выполнении действий (например, выход из аккаунта) и манипуляциях с данными (например, удаление файла) отображается сообщение с подтверждением | Отрицательно | Дополнить регистрацию аккаунта подтверждением по почте |
| 24 | При смене пароля пользователю приходит подтверждение по электронной почте | Отрицательно | Дополнить безопасность смены пароли на серверной части |
| Навигация | | | |
| 25 | Корректность работы кнопок и ссылок в меню навигации | Положительно |  |
| Продолжение таблицы 10 | | | |
| 1 | 2 | 3 | 4 |
| 26 | Отображение страницы с ошибкой 404, если пользователь указал неверный путь к файлу или URL | Положительно |  |
| 27 | Отображение страницы с ошибкой 403, если пользователь указал зашёл на запрещенный ему ресурс | Положительно |  |
| 28 | Отображение страницы с ошибкой 503, если ответ от сервера клиенту не пришёл (истекло время ожидания) | Положительно |  |
| 29 | Числовые поля не допускают ввода текста, отображается сообщение об ошибке | Положительно |  |
| 30 | При попытке ввода отрицательных чисел, отображается сообщение об ошибке, если предусмотрен ввод строго положительных чисел | Положительно |  |
| 31 | Поля специальных форматов (сумма денег, даты) отображаются корректно | Положительно |  |
| 32 | На странице для пользователей с ролью ADMIN отображается панель администрирования | Положительно |  |
| 33 | На странице для пользователей с ролью USER не отображаются элементы администрирования сайта | Положительно |  |

В данном разделе представлены выводы по результатам испытаний, а также общая статистика по дефектам, которые были выявлены в результате тестирования информационной системы. В период испытаний был выявлено 4 дефекта из 33 тест-кейсов.

Рисунок N – Диаграмма результатов выполнения тест-кейсов

2.3 Руководство пользователя

В приложении имеется отличный функционал для пользователя – это возможность управлять своими транзакциями (просмотр, пополнение баланса) и управлять своими заданиями (просмотр всех заданий и добавление нового задания). Функционал может иметь только авторизованный пользователь, а также пользователь с ролью USER или ADMIN.

Чтобы пользователь мог работать с своим балансом, то нужно перейти из навигации панели сайта в раздел «Пополнение», где будет предоставлен следующий интерфейс как на рисунке N.

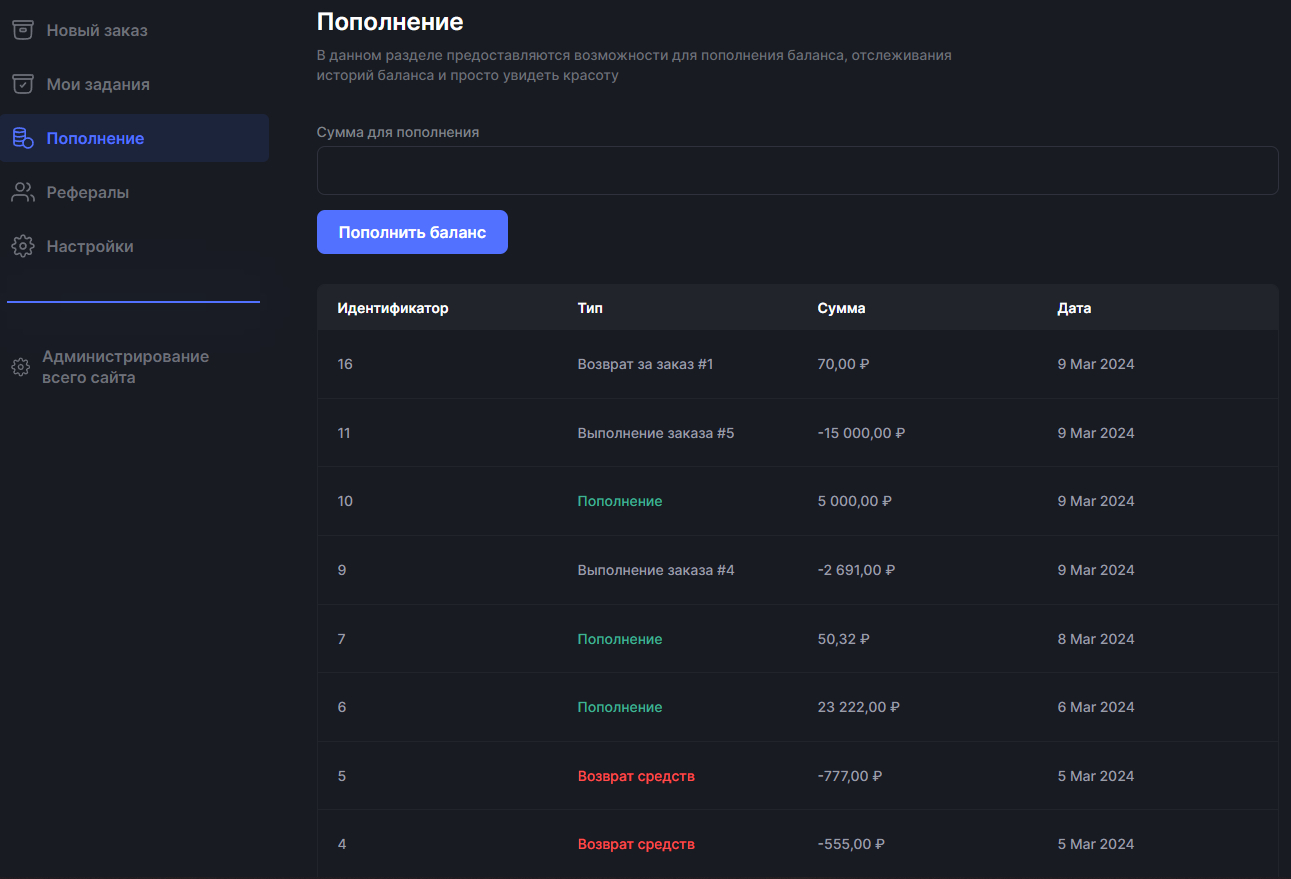


Рисунок 22 – Интерфейс раздела «Пополнение»

На данной странице можно увидеть все транзакции, которые имеют разный тип операции. Здесь же пользователь может легко пополнить свои средства через простую форму, вбив желаемую сумму (минимум 50 рублей, максимум 30000) в поле. После успешной отправки запроса на сервер пользователю придёт ответ:

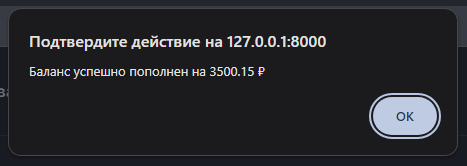


Рисунок N – Ответ сервера после успешного пополнения баланса

После того как пользователь имеет какой-то баланс на своём счету он может создать новое задание по накрутке определённой активности. Перейдя в раздел «Новый заказ», пользователь встретит удобный и красивый интерфейс как на рисунке N.

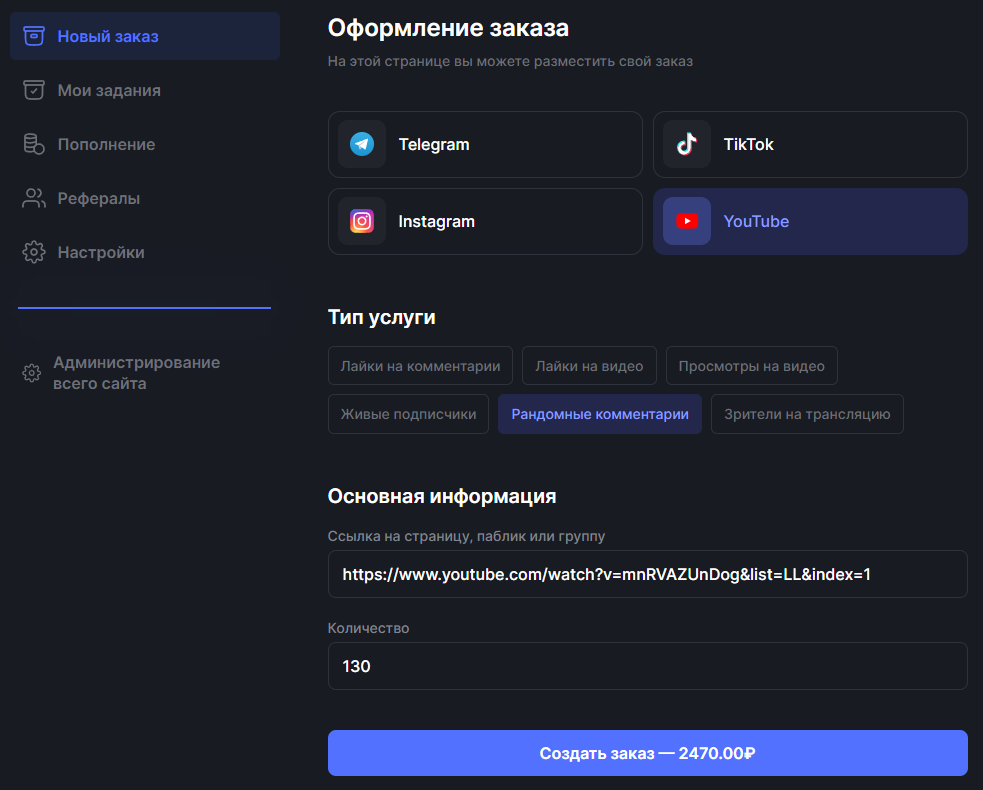


Рисунок N – Интерфейс раздела «Новый заказ»

На данной странице пользователю предоставляется самый главный функционал информационной системы – оформление заказа. Оформлением заказа является формирование задания, в котором будет выбран определённый сервис, тип услуги, и вписана основная информация. Пользователь может выбрать любой доступный сервис и любую доступную услугу накрутки, которая ему будет необходима. Дальше пользователю нужно вбить основную информацию – количество (от 50 до 13000) и адрес какой-либо страницы в какой-то социальной сети (по которому в дальнейшем будет исполняться заказ для пользователя). После этого всего в кнопке «Создать заказ» пользователь увидит итоговую стоимость заказа. Если пользователь нажмёт на кнопку, то запрос придёт на сервер. Если у пользователя недостаточно средств на балансе, то ему покажет сообщение от сервера о необходимости пополнении баланса, чтобы в дальнейшем оформить заказ.

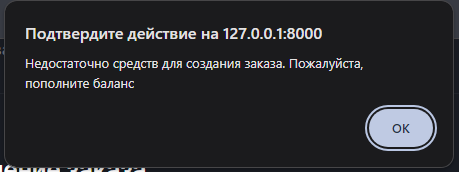


Рисунок N – Ответ от сервера о нехватке баланса при оформлении заказа

В случае если у пользователя хватило средств на создание заказа его автоматически переведёт в раздел «Мои задания». В данном разделе пользователь сразу увидит все свои созданные задания, а также в самом верху отобразится последнее созданное им задание. С помощью этой страницы пользователь может отслеживать статусы своих заказов, а также идентификаторы заказы, чтобы в случае неправильного оформления заказа пользователь мог легко обратиться в техподдержку сайта, потребовав отмены заказа.

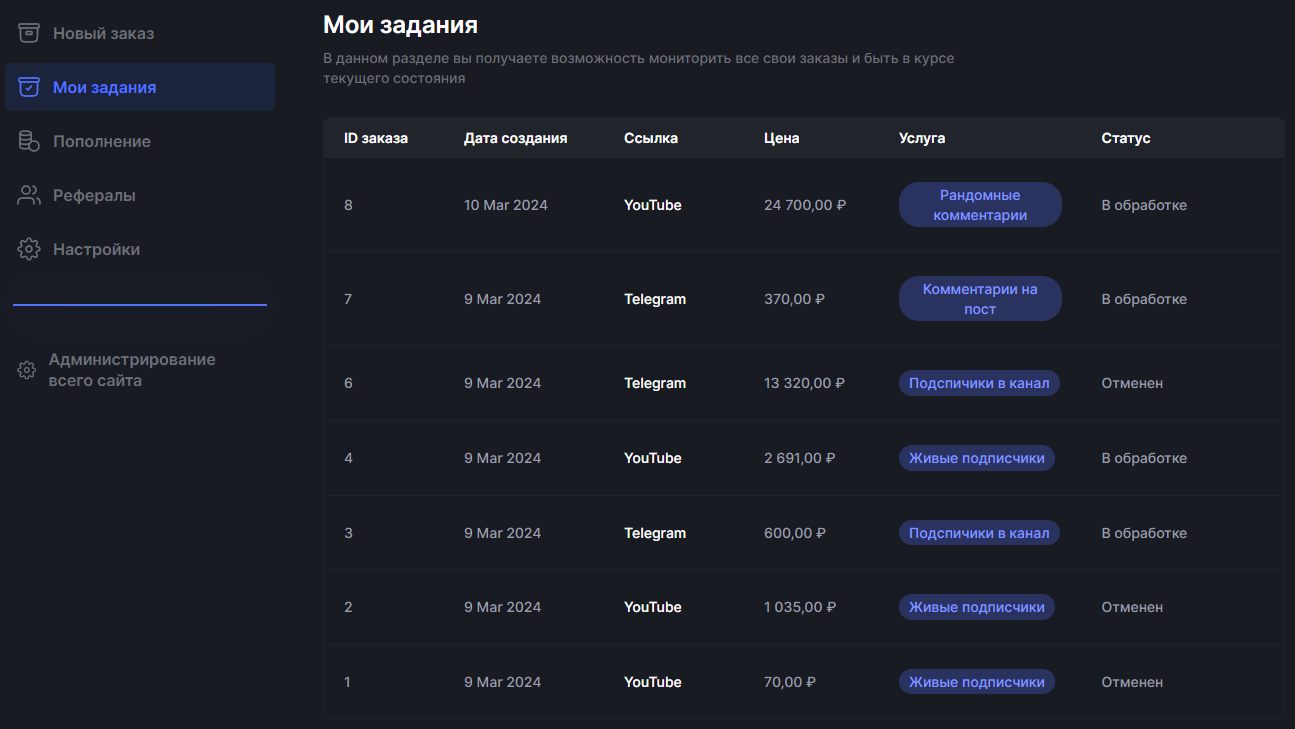


Рисунок N – Интерфейс раздела «Мои задания»

2.4 Руководство администратора

В информационной системе имеется функционал отдельно для администратора. Все действия, ссылки для администрирования начинаются с префикса «admin» в URL адресе. Добавлять информацию можно в следующие таблицы – пользователи, сервисы и типы услуг к сервисам.

Под каждую категорию работы с информационной системы созданы отдельные страницы и разделы. Чтобы двигаться по разделам была создана удобная фиксированная навигация, дав быстрый переход между страницами администратору.

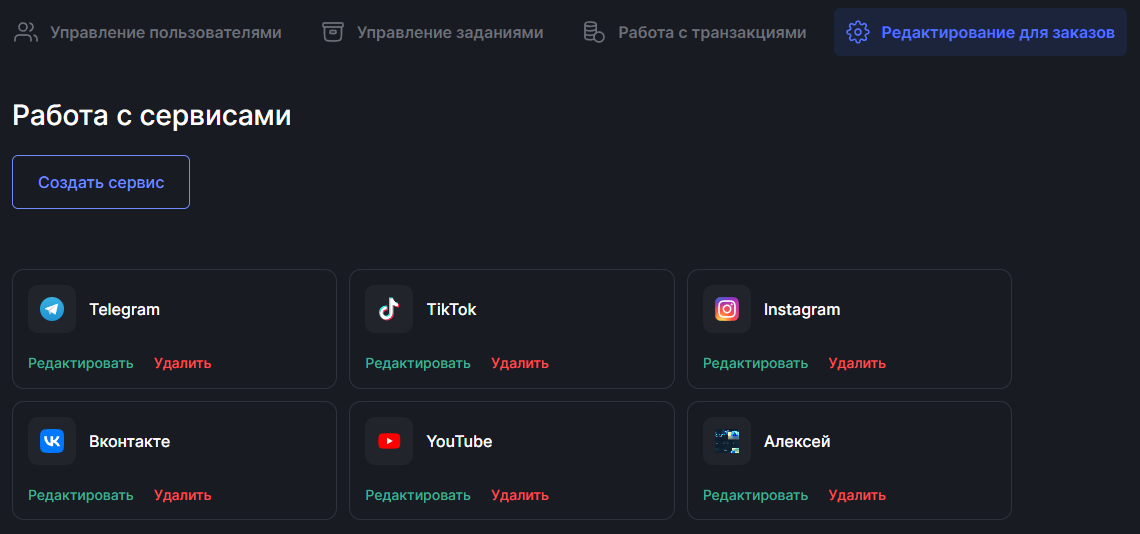


Рисунок N – Интерфейс раздела администрирования с навигацией

В интерфейсе администрирования можно заниматься добавлением элементов сущностей. Чтобы вызвать окно или страницу добавления достаточно будет нажать кнопку «Создать сервис» в определенном интерфейсе системы. Когда была нажата кнопка, то откроется интерфейс, в котором будут доступны поля для заполнения новыми данными.

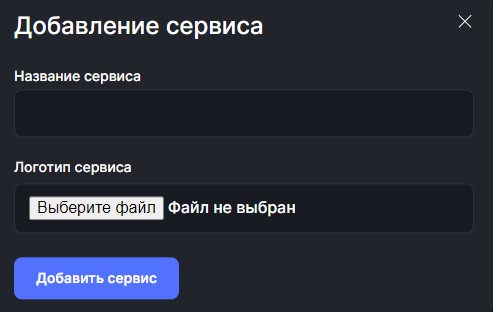


Рисунок N – Руководство администратора для добавления данных

Дальше идет заполнение данных администратором. Но стоит учитывать, что если были введены некорректные данные, то в итоге будут выведены свои ошибки для каждого поля, в которых будет описано что не так заполнено в поле. После того как будет нажата кнопка «Добавить» в модальном окне, то после успешного создания, созданный элемент отобразится в интерфейсе без перезагрузки страницы.

В информационной системе имеется функция редактирования данных. Редактировать информацию можно в следующих таблицах – сервисы, типы услуг к сервисам и пользователи. Чтобы вызвать интерфейс редактирования нужно сначала определиться какой элемент сущности будет изменяться.

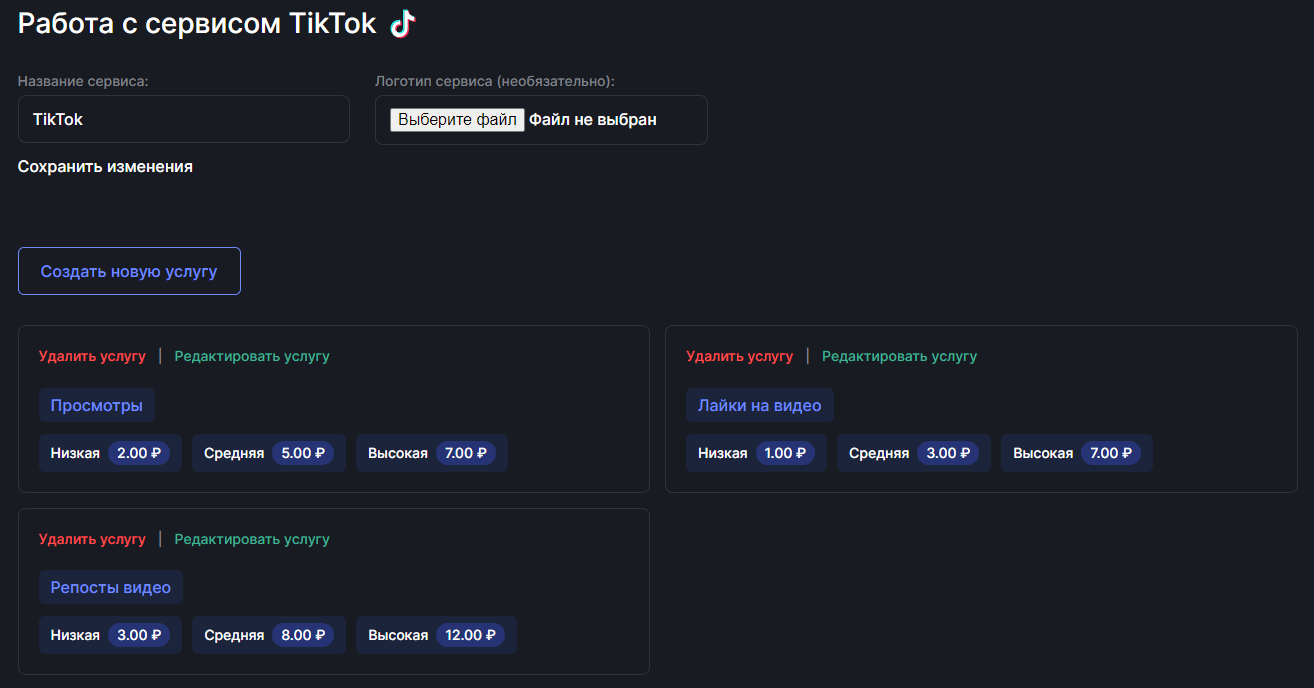


Рисунок N – Руководство администратора для редактирования данных

Когда мы нажали на кнопку, то откроется сам интерфейс редактирования в модальном окне или же на отдельной странице редактирования, и будут заполнены поля значениями, взятыми из определенной записи.

Далее идет заполнение новыми данными под старые. Стоит учитывать, что если были введены некорректные данные, то в итоге будут выведены свои ошибки для каждого поля, в которых будет описано что не так заполнено в поле, и потребуется изменить данные чтобы они стали корректными, иначе данные не отправятся на сервер.

После того как была нажата кнопка «Сохранить», изменения вступят в силу. Администратор на клиенте увидит перенаправление на другую страницу (в случае успеха) или же обновление данных без перезагрузок (в зависимости от того, что именно было изменено). Далее запись, которая изменилась, обновится в интерфейсе системы.

В информационной системе имеется функция удаления существующих данных. Удалять информацию можно из следующих таблицах – пользователи, сервисы, и типы услуг к сервисам. Чтобы вызвать удаление достаточно будет нажать на кнопку удаления у определенной записи, и откроется всплывающее окно с подтверждением удаления данных.

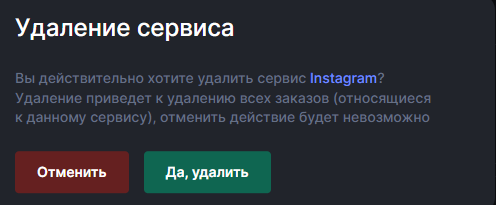


Рисунок N – Руководство администратора для удаления данных

Если была нажата кнопка «Да, удалить», то удалится запись. После этого обновится список данных в интерфейсе системы, где удаленная запись не будет уже отображаться.

Если же была нажата кнопка «Отменить», то запись не будет удалена из базы данных, и закроется всплывающее окно подтверждения удаления записи.

В информационной системе у администратора есть возможность отменять некоторые операции в информационной системе. На данный момент имеется возможность отменять транзакции с типом «Пополнение» в разделе «Работа с транзакциями», а также в разделе «Управление заданиями» можно отменять созданные задания пользователем, нажав на кнопку «Отменить задание».

Отличие данного действия от удаления заключается в том, что администратор не удаляет полностью данные из БД, программа их фиксирует в БД дальше, создав при этом поверх старой записи новую запись, в которой, например будет указана другая дата выполнения операции и статус операции.

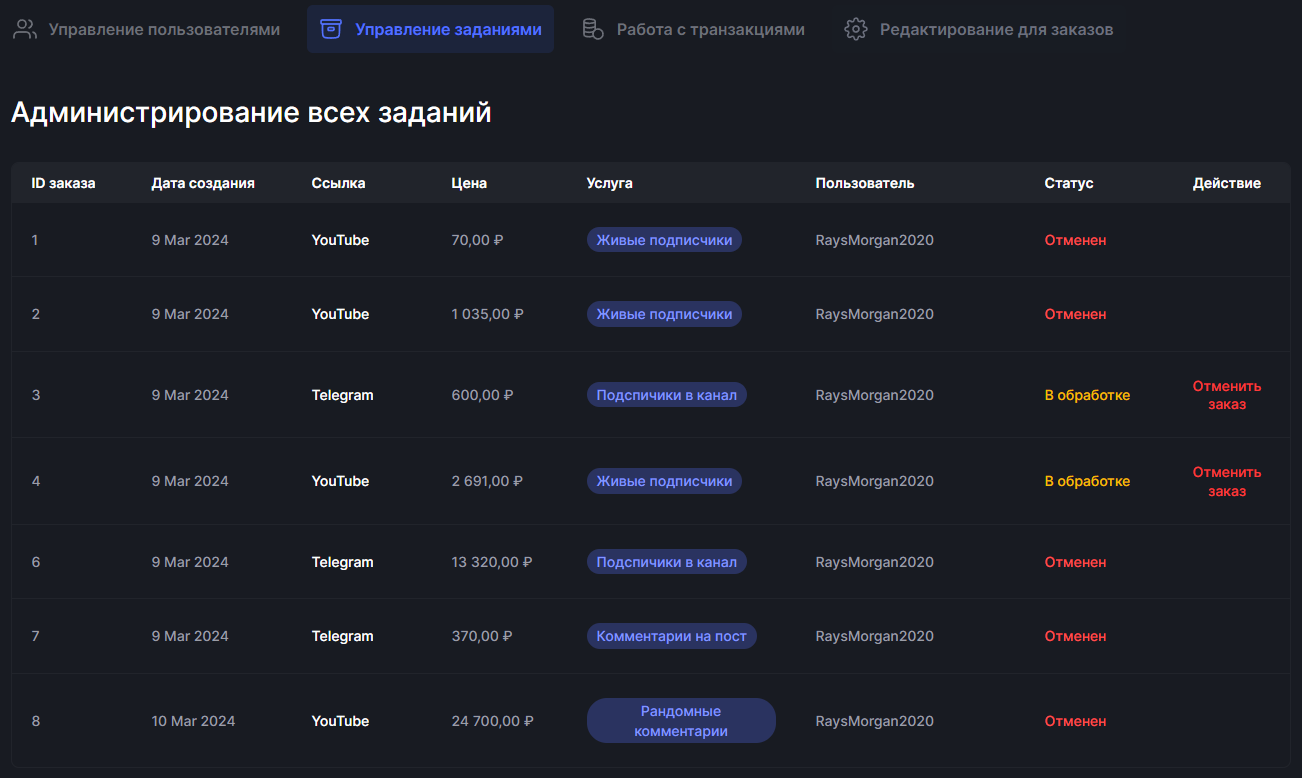


Рисунок N – Руководства администратора для отмены операций

Когда будет нажиматься кнопка отмены какой-либо операции то будет всплывать диалоговое окно, в котором запросится подтверждение на данное выполнение действия. Если администратор нажмёт «Ок», то операция будет выполнена успешно и изменения вступят в силу. В интерфейсе обновится страница и добавится новая запись с новой датой создания, идентификатором, а также статусом операции.

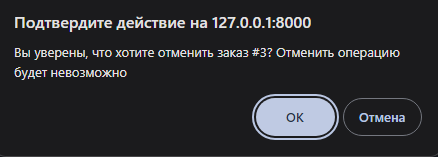


Рисунок N – Руководства администратора для отмены операции

Статус операции будет отличаться от первоначальной операции. Повторно отменять отменённые операции администратор не сможет, кнопка для таких записи будет скрыта автоматически.

ЗАКЛЮЧЕНИЕ

Разработка интернет-приложения по продаже строительных материалов на базе фреймворка PHP Laravel с шаблонизатором Blade позволила создать функциональное и эстетичное приложение. Используя современные инструменты разработки, включая фреймворк Laravel, достигнута гибкость и масштабируемость приложения. Целью проекта было создание удобного интерфейса для взаимодействия с интерфейсом и функционалом, обеспечивая пользователям уникальный и легкий опыт в выполнения действия в пару кликов.

Использование фреймворка Laravel значительно облегчило расширение функциональности сервиса с возможным будущим функционалом. Масштабируемость этого фреймворка позволила без проблем внедрять новые возможности и обрабатывать увеличивающееся количество запросов.

Одной из ключевых задач было обеспечение безопасности, интуитивной навигации и удобства в управлении контентом для администраторов. Используя Blade шаблоны в Laravel, удалось создать интуитивно понятный интерфейс для администраторов и пользователей, обеспечив безопасную и удобную среду для работы с данными о клиентах, онлайн-заявках, товаров, отзывах и заявками на партнёрство, созданные от обычных пользователей.

Инфраструктура на основе PHP Laravel и шаблонизатора Blade позволила быстро реализовать функционал добавления, редактирования и удаления сущностей, а также создать эффективную систему создания заданий, что значительно облегчило процесс пользовательского опыта в информационной системе.

Задачи, описанные в начале исследовательской работе решены, а именно создать трёхуровневую архитектуру приложения, быстрый доступ к базе данных, добавлению различных пользовательских заданий, просмотра, редактирования и удаления товаров, пользователей, а также управление другим контентом на сайте.

В результате выполнения учебной практики, была создана динамическая информационная система для внесения и хранения данных о онлайн-заявках, заявок на партнёрство, и отзывов. Для его разработки применялось актуальное программное обеспечение, без помощи которого не было бы возможности добиться поставленной цели. Это программное обеспечение включает в себя:

* язык программирования PHP;
* коммерческая кроссплатформенная интегрированная среда разработки для PHP – PhpStorm;
* фреймворк PHP Laravel;
* веб-сервер XAMPP;
* реляционная система управления БД MySQL;
* приложение для администрирования СУБД MySQL PhpMyAdmin;

Этот проект стал хорошим примером использования современных технологий разработки в области интернет-приложений для строительных компаний, обеспечивая пользователей качественным и интуитивно понятным интерфейсом, а администраторов — легким управлением контентом. Всё это было возможно благодаря гибкости и функциональности PHP Laravel и Blade шаблонизатора, которые позволили создать удивительную информационную систему, удовлетворяющий потребности как пользователей, так и администраторов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вандюк, Джон К. PHP Laravel. Руководство по разработке системы управления сайтом / Вандюк, Джон К., Мэтт Вестгейт, - М.: Вильямс, 2019. - 400 c.
2. Гаевский, А.Ю. 100% самоучитель. Создание Web-страниц и Web-сайтов. HTML и JavaScript / А.Ю. Гаевский, В.А. Романовский. - М.: Триумф, 2020. - 464 c.
3. Дакетт, Джон HTML и CSS. Разработка и дизайн веб-сайтов / Джон Дакетт. - Москва: Наука, 2019. - 480 c.
4. Дронов, В. JavaScript в Web-дизайне / В. Дронов. - М.: СПб: БХВ, 2019. - 880 c.
5. Дронов, В. PHP, MySQL и Dreamweaver. Разработка интерактивных Web-сайтов / В. Дронов. - М.: БХВ-Петербург, 2019. - 480 c.
6. Дронов, В.А. PHP 5/6, MySQL 5/6 и Dreamweaver CS4. Разработка интерактивных Web-сайтов / В. А. Дронов. - М.: БХВ-Петербург, 2019. - 820 c.
7. Дронов, Владимир JavaScript и AJAX в Web-дизайне / Владимир Дронов. - Москва: Высшая школа, 2019. - 736 c.
8. Дронов, Владимир PHP, MySQL и Dreamweaver MX 2004. Разработка интерактивных Web-сайтов / Владимир Дронов. - М.: "БХВ-Петербург", 2019. - 448 c.
9. Колисниченко, Д. PHP 5/6 и MySQL 6. Разработка Web-приложений / Д. Колисниченко. - М.: БХВ-Петербург, 2021. - 560 c.
10. Колисниченко, Д.Н. PHP 5/6 и MySQL 6. Разработка Web-приложений / Д.Н. Колисниченко. - М.: БХВ-Петербург, 2021. - 364 c.
11. Кузнецов PHP. Практика создания Web-сайтов / Кузнецов, М.В. и. - М.: БХВ-Петербург, 2020. - 895 c.
12. Кузнецов, М. PHP 5. Практика создания Web-сайтов / М. Кузнецов, И. Симдянов, С. Голышев. - М.: БХВ-Петербург, 2020. - 960 c.
13. Леонтьев Борис PHP 5.0 для начинающих, или как создать динамический WEB-сайт / Борис Леонтьев. - М.: Новый издательский дом, 2019. - 176 c.
14. Петюшкин Алексей HTML в Web-дизайне / Алексей Петюшкин. - Москва: Машиностроение, 20. - 400 c.
15. Шкрыль, А. PHP — это просто. Программируем для Web-сайта / А. Шкрыль. - М.: БХВ-Петербург, 2021. – 360 с.
16. ГОСТ 19.201-78 «Техническое задание».
17. ГОСТ 19.504-79 ЕСПД «Руководства программиста»
18. Дакетт Д. – HTML и CSS. Разработка и дизайн веб-сайтов, Эксмо, 2019, 480 с.
19. Дронов В. – Laravel 8. Быстрая разработка веб-сайтов на PHP, БХВ-Петербург, 2020 688 с.

ПРИЛОЖЕНИЯ

Kernel.php – ядро приложения, который выполняет роль запуска

<?php  
  
namespace App\Http;  
  
use App\Http\Middleware\AdminMiddleware;  
use App\Http\Middleware\Authenticate;  
use App\Http\Middleware\CheckDatabaseConnection;  
use Illuminate\Foundation\Http\Kernel as HttpKernel;  
  
class Kernel extends HttpKernel  
{  
 protected $routeMiddleware = [  
 'admin' => AdminMiddleware::class,  
 'checkdb' => CheckDatabaseConnection::class,  
 'auth' => Authenticate::class,  
 ];  
  
 protected $middleware = [  
 // \App\Http\Middleware\TrustHosts::class,  
 \App\Http\Middleware\TrustProxies::class,  
 \Illuminate\Http\Middleware\HandleCors::class,  
 \App\Http\Middleware\PreventRequestsDuringMaintenance::class,  
 \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,  
 \App\Http\Middleware\TrimStrings::class,  
 \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,  
 ];  
  
 protected $middlewareGroups = [  
 'web' => [  
 \App\Http\Middleware\EncryptCookies::class,  
 \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,  
 \Illuminate\Session\Middleware\StartSession::class,  
 \Illuminate\View\Middleware\ShareErrorsFromSession::class,  
 \App\Http\Middleware\VerifyCsrfToken::class,  
 \Illuminate\Routing\Middleware\SubstituteBindings::class,  
 ],  
  
 'api' => [  
 // \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,  
 \Illuminate\Routing\Middleware\ThrottleRequests::class . ':api',  
 \Illuminate\Routing\Middleware\SubstituteBindings::class,  
 ],  
 ];  
  
 protected $middlewareAliases = [  
 'auth' => Authenticate::class,  
 'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,  
 'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,  
 'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,  
 'can' => \Illuminate\Auth\Middleware\Authorize::class,  
 'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,  
 'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,  
 'precognitive' => \Illuminate\Foundation\Http\Middleware\HandlePrecognitiveRequests::class,  
 'signed' => \App\Http\Middleware\ValidateSignature::class,  
 'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,  
 'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,  
 ];  
}

StoreUserRequest.php – отвечает за хранение запросов для пользователя

<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class StoreUserRequest extends FormRequest

{

    public function authorize(): bool

    {

        return auth()->user()->role === 'ADMIN';

    }

    public function rules(): array

    {

        return [

            'login' => 'required|string|min:5|max:60|unique:users',

            'name' => 'required|string|min:2|max:50',

            'email' => 'required|email|unique:users,email',

            'password' => 'required|string|min:8|max:60',

            'role' => 'required|in:ADMIN,USER',

        ];

    }

}

UpdateUserRequest.php – отвечает за обновление запросов пользователя

<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class UpdateUserRequest extends FormRequest

{

    public function authorize(): bool

    {

        return auth()->user()->role === 'ADMIN';

    }

    public function rules(): array

    {

        return [

            'login' => 'required|string|min:5|max:60|unique:users,login,' . $this->user->id,

            'name' => 'required|string|min:2|max:50',

            'email' => 'required|email|unique:users,email,' . $this->user->id,

            'password' => 'nullable|string|min:8|max:60',

            'role' => 'required|in:ADMIN,USER',

        ];

    }

}