Geekbrains

Разработка специализированного сайта по продаже кормов для животных с использованием React

Программа:

Специализация

Тушминцева Мария Михайловна

Санкт-Петербург

2024

Оглавление.

Введение

Тема проекта.

Тема проекта – разработка сайта на основе JavaScript-фреймворке React. Сайт будет специализирован на продаже кормов для животных от производителя и должен будет решать задачу удобного для пользователей и привлекательного предоставления информации о различных видах кормов, их составе, преимуществах и недостатках, подбора корма по выбранным параметрам.

Специализация дипломного проекта

Дипломный проект будет выполнен в рамках специализации: Frontend разработка (React).

Актуальность темы проекта.

В современном мире всё больше людей осознают важность заботы о домашних питомцах и их правильном питании. Увеличивается и выбор соответствующих товаров на рынке. Все больше появляется новых производителей и ритейлеров, все больше новых линеек кормов с учетом тонких особенностей здоровья, конституции, породы животного и множества других факторов.

С точки зрения целевых клиентов – пользователей соответствующих информационных продуктов, то есть потенциальных клиентов – покупателей кормов для животных, актуальна проблема сложности получения информации и ее анализа для взвешенного выбора.

С точки зрения продавца кормов для животных актуален вопрос получения конкурентного преимущества путем повышения доверия со стороны клиентов, увеличение времени сессии потенциального покупателя и, в итоге, повышения уровня продаж.

В связи с этим возникает потребность со стороны потенциального покупателя в возможности использовать интуитивно понятный и функциональный веб-инструмент для выбора корма своему питомца, а со стороны продавца – потребность в создании специализированного сайта, который позволит увеличить лояльность клиентов и поднять уровень продаж кормов для животных.

Цель проекта.

Проект нацелен на разработку и реализацию специализированного сайта по продаже кормов для животных. Сайт должен будет предоставлять достойную репрезентацию имеющихся линеек кормов от производителя с возможностью сравнения различных кормов по ряду параметров, на основе индивидуальных потребностей их питомцев.

Задачи проекта.

- Анализ конкурентов.

- Определение целевой аудитории.

- Разработка структуры и дизайна frontend-решения.

- Написание текстов и наполнение контентом.

- Тестирование и оптимизация frontend-решения.

- Интеграция frontend-решения с backend-системой.

- Запуск сайта.

Используемые инструменты.

- Figma: для создания макета;

- Node.js: для установки зависимостей и работы с сервером;

- npm: для управления пакетами и зависимостями проекта;

- Create React App: для быстрой настройки начального проекта на React;

- React Router: для реализации маршрутизации и навигации по страницам приложения;

- Styled Components или CSS Modules: для стилизации компонентов с использованием CSS-in-JS или модульного подхода;

- Git и GitHub: для контроля версий проекта/

Опыт и компетенции.

Мой опыт работ по созданию сайтов ограничен выполнением учебных и тестовых заданий по курсу обучения «Веб-разработчик».

В процессе работы над проектом я планирую выступать в роли веб-разработчика, а также самостоятельно выполнять задачи по анализу, дизайну, тестированию и прочие сопутствующие задачи самостоятельно.

Глава 1. Теоретическая часть.

1.1 Определение, структура и функции Веб-сайта.

В настоящее время веб-сайт – это повсеместно используемый объект, используемый в различных целях, широко трактуемое понятие и общественно важный феномен, влияние которого на современную жизнь сложно переоценить.

Под веб-сайтом подразумевается объединенная общим доменным именем система взаимосвязанных веб-страниц, опубликованных в открытом доступе в сети Интернет. Веб-страницей (а также веб-сайтом, состоящим из единственной веб-страницы) называется информационный массив или документ, предоставляющий функцию просмотра, копирования, а также, опционально, ввода и передачи данных, проведения вычислений, операций с данными, фото, видео и аудио материалами.

Обычно веб-страница состоит из следующих основных элементов: заголовок, меню навигации, блоки контента, сам контент. Целью формирования архитектуры веб-сайта и его отдельных страниц является удобный интерфейс для пользователей, позволяющий легко ориентироваться по сайту и получать необходимую информацию / реализовывать необходимые функции.

Список функций, которые могут быть реализованы в рамках веб-сайта, дополняется практически ежедневно. Веб сайты могут представлять из себя как простые статичные страницы, связки страниц или сложные динамические платформы, которые взаимодействуют с пользователем в режиме реального времени.

Пример классификации веб-сайтов по целям и выполняемым задачам:

- коммерческие;
- образовательные;
- информационные;
- развлекательные;
- социальные платформы.

Важно отметить, что веб-сайты являются важнейшим инструментом маркетинга и продвижения для коммерческих компаний практически в любых сферах, позволяя компаниям расширять свое присутствие в интернете и привлекать целевую аудиторию.

Таким образом, веб-сайт является важным объектом в жизни общества и практически каждого человека, выступая связующим элементом между пользователями и информацией. Понимание структуры и функций веб-сайта, взгляд со стороны как пользователя, так и владельца важно для эффективной разработки веб-сайта.

1.2 Компетенции программиста веб-разработки.

Специализация программиста по разработке сайтов приобрела высокую значимость с увеличением числа онлайн-платформ и сервисов. Современные компании и социальные институты требуют не только создание сайта на старте деятельности, но и частое обновление для того, чтобы эффективно использовать постоянно появляющиеся новинки и предотвратить моральное устаревание веб-сайта («быть в тренде»). Следовательно, требуются квалифицированные специалисты, способные оперативно внедрять передовые инструменты и подходы к разработке сайтов и их поддержке.

Один из необходимых навыков в работе программиста – это владение языков программирования. Для создания веб-сайтов наиболее распространенными являются:

- HTML (HyperText Markup Language) – основа для формирования структуры большинства веб-страниц, фокусируется на форматировании страницы и организации контента.

- CSS (Cascading Style Sheets) – часто применяется для формирования дизайна страницы и удобного интерфейса пользователя.

- JavaScript – сфокусирован на реализации разнообразных интерактивных элементов, что позволяет реализовывать динамические функции.

Программирование в настоящий момент редко рассматривается в отрыве от международного опыта – то есть мирового сообщества разработки. Крупные IT-компании и отдельные энтузиасты вносят свои вклады в формирование баз кода, называемых библиотеками и фреймворками.

Для программирования в сфере веб одной из наиболее известных и развитых является библиотека React для языка JavaScript, которая позволяет внедрять на веб-сайт самые современные компоненты.

Программисту по созданию сайтов также требуется углубленное понимание по ряду прочих сфер деятельности: дизайна для формирования функциональных и эстетически приятных страниц, проектирования удобного пользовательского интерфейса; backend-

разработки для эффективного взаимодействия клиентской стороны сайта с сервером; баз данных для возможности хранения, оперативного изменения информации, точной обработки пользовательских запросов; поисковой оптимизации (SEO – Search Engine Optimization) для того, чтобы разработанный сайт в будущем имел преимущества в жесткой конкуренции по позиционированию в поисковых системах, что необходимо для привлечения большего числа пользователей.

1.3 HTML – технология гипертекста.

HTML (HyperText Markup Language) — это язык разметки гипертекста, на котором формируется основа веб-страницы. Благодаря данной технологии на странице создается структура – организация текста, изображений, и других статических и динамических элементов. Также важной функцией HTML является создание гиперссылок, то есть связей данной страницы с другими страницами и практически любыми внешними ресурсами – данными любого вида, доступ к которым может быть получен через сеть интернет.

Каждый элемент в HTML создается с помощью тегов, которые заключаются в угловые скобки. Например, тег <h1> обозначает заголовок первого уровня, а тег <p> — абзац текста. Гиперссылки оформляются тегом <a>. Также возможно встраивание мультимедийных элементов непосредственно в данную веб-страницу: теги <img>, <video> и <audio> используются для встраивания изображений, видео и аудио файлов.

Принято считать HTML языком разметки для формирования лишь каркаса сайта, а большинство функций реализовывать с помощью других инструментов. Однако, гипертекстовая разметка также развивается. Современная версия языка HTML – это HTML5. Он содержит множество обновлений, например такие семантические элементы как <header>, <footer>, <article> и <section>, которые позволяют лучше структурировать контент, упрощая работу по созданию страницы, привлекательной для пользователей и оптимизированной для поисковых систем.

HTML является основой веб-разработки. Благодаря простоте и гибкости он относительно прост для изучения новичками, но также необходим и для опытных разработчиков. Кроме того, HTML содержит широкий арсенал встроенных функций. Используя HTML, можно создавать структурированные, интерактивные и мультимедийные веб-страницы, которые соответствуют современным требованиям пользователей. Также в современном языке гипертекста предусматривается эффективное совмещение с языками frontend-разработки, в частности JavaScript.

1.4 Использование библиотеки React при веб-разработке.

Библиотека React, изначально разработанная Facebook, является очень популярным инструментом для разработки веб-сайтов. Она сочетает в себе модульную структуру, высокую производительность и широкий спектр возможностей.

Данная библиотека поддерживается активным интернет-сообществом. Таким образом, библиотека часто пополняется, актуализируется. Имеется большое количество доступных для использования материалов, и это количество продолжает расти. А также существует множество документации и обучающих ресурсов, облегчающих освоение выбранных инструментов из библиотеки. Кроме того, React – это экосистема, включающая множество библиотек, таких как Redux для управления состоянием и React Router для маршрутизации.

Одной из особенностей React является компонентный подход. Используя React разработчик может создавать веб-страницу из независимых и переиспользуемых элементов. Разработанная на этапе предварительного веб-дизайна сложная структура может быть разбита на более мелкие и простые части, благодаря чему процесс разработки и последующее внесение изменений значительно упрощается. Структура гибкая и динамичная, разработчик может контролировать состояние, свойства и работоспособность каждого компонента по отдельности. А каждый компонент напрямую связан с разметкой (с HTML). Таким образом, React значительно упрощает поиск ошибок и организацию кода, а также повышает скорость работы веб-страницы.

React хорошо интегрируется с другими технологиями и фреймворками. Например, разработчики могут легко использовать React вместе с серверными языками, такими как Node.js, или интегрировать его с популярными системами управления контентом (CMS). Это делает React универсальным инструментом для создания как простых, так и сложных веб-приложений.

Еще одной важной особенностью React является виртуальный DOM (Document Object Model – «объектная модель документа»). Классическим подходом является использование «реального DOM», когда каждое изменение в интерфейсе (UI) переписывает древовидную структуру DOM, то есть все элементы в списке, относящиеся к изменяемому объекту UI.

Так как веб-приложение, формируемое по современным стандартам UI, содержит часто очень большое количество изменяемых объектов, переписывание DOM может оказывать очень существенное влияние на производительность сайта.

Альтернативным вариантом является реализованное с помощью React использование «виртуального DOM» (VDOM). В этом случае при обновление определенного компонента

сначала внесенные изменения сравниваются с ранее сохраненными значениями, затем React оптимизирует количество операций, необходимых для обновления интерфейса, и только уже минимизированное изменение перезаписывается.

Библиотека React поддерживает концепцию одностороннего потока данных для управления состоянием приложения. Эта концепция означает, что данные передаются от родительских компонентов к дочерним. Следовательно, изменения в состоянии происходят в предсказуемом порядке, что значительно упрощает отладку и тестирование приложений. Такой подход позволяет легче отслеживать изменения и предотвращает возможные проблемы, связанные с синхронизацией состояний.

Современные подходы к разработке на React также включают использование «функциональных компонентов» и «хуков». Функциональные компоненты, в отличие от классовых, предлагают более простой и лаконичный синтаксис, что делает код более читаемым и удобным для работы. Хуки позволяют добавлять состояние и другие функции React в функциональные компоненты, что дает разработчикам возможность использовать преимущества React без необходимости создания классовых компонентов. Это делает разработку более удобной и понятной.

Следовательно, React предлагает разработчикам множество преимуществ и возможностей для создания современных веб-сайтов. Его компонентный подход, виртуальный DOM, поддержка одностороннего потока данных, использование функциональных компонентов и хуков делают его одним из самых привлекательных инструментов для веб-разработки, позволяя разработчикам создавать высококачественные и производительные веб-приложения.

1.5 Инфраструктура React: Redux, React Router и другие библиотеки.

Существует ряд библиотек JavaScript, которые нацелены на выполнение отдельных функций и созданных специально для взаимодействия с React при создании веб-сайтов.

В случаях, когда разработчику требуется точное управление состоянием при создании сайта со сложной структурой, часто применяется библиотека Redux. Она позволяет централизовать управление состоянием и облегчает взаимодействие между различными компонентами. Redux предлагает мощные инструменты для отслеживания изменений состояния, что особенно полезно в крупных приложениях с множеством взаимодействий.

Другим важным инструментом является React Router. Эта библиотека позволяет реализовать маршрутизацию в приложениях на React, что упрощает переходы между

различными страницами и компонентами. React Router особенно полезен для создания одностраничных приложений (SPA), где необходимо быстро загружать новые данные без перезагрузки страницы.

С точки зрения стилизации компонентов на React существует множество подходов. Одним из самых популярных является библиотека Styled-components, которая позволяет писать CSS прямо в JavaScript. Такой подход делает стилизацию более модульной и удобной, так как стили привязываются к компонентам. Также есть CSS-in-JS решения, такие как Emotion, которые предлагают аналогичные возможности.

Для быстрого старта новых проектов на React разработчики могут воспользоваться утилитой Create React App. Этот инструмент позволяет быстро создать новый проект, автоматически настраивая все необходимые зависимости и конфигурации. Create React App включает в себя такие инструменты, как Webpack и Babel, которые отвечают за сборку и транспиляцию кода.

Еще одним полезным инструментом для работы с API и выполнения асинхронных запросов является Axios — популярная библиотека для выполнения HTTP-запросов. Axios предлагает простой интерфейс для работы с API, поддерживает промисы и позволяет удобно обрабатывать запросы и ответы.

Кроме того, для отладки и разработки приложений на React полезно использовать встроенные инструменты разработчика в браузерах, например, React Devtools. С их помощью разработчики могут отслеживать производительность приложений, анализировать состояние компонентов и выявлять потенциальные проблемы.

Таким образом, экосистема React позволяет сделать процесс создания веб-страниц более удобным и эффективным, благодаря множеству инструментов. Использование таких инструментов, как Redux для управления состоянием, React Router для маршрутизации, Styled-components для стилизации позволяет разработчикам создавать качественные и масштабируемые приложения. Эти инструменты помогают упростить разработку, повысить производительность и улучшить взаимодействие с пользователями.

Глава 2. Практическая часть проекта.

2.1 Определение требований заказчика.

Удобным инструментом для определения требований заказчика к определения требуемой функциональности сайта является бриф. Это документ – заранее сформированный перечень вопросов, позволяющих выстраивать взаимопонимание с заказчиком на начальном этапе.

В минимальном виде бриф включает следующие вопросы:

- целей сайта;

- целевая аудитория;

- функции сайта (например: авторизация, поиск, покупка);

- референсы (подходящие примеры);

- контент, который должен будет размещаться на сайте.

Пример заполненного брифа представлен в Приложении [1].

После получения брифа необходимо изучить конкурентную среду и подготовить спецификацию проекта для описания конечного результата, на основе критериев которого можно будет определить, что сайт соответствует ожиданиям заказчика.

2.2 Конкурентный анализ.

Конкурентный анализ является важным предварительным этапом веб-разработки, особенно для запуска нового проекта. Он позволяет определить сильные и слабые стороны конкурентов, а также лучше понять текущие тренды в дизайне и функциональности. Для того, чтобы провести эффективный конкурентный анализ, рекомендуется выполнить следующие шаги.

Определить цели анализа. Перед тем как приступить к исследованию, важно четко определить цели конкурентного анализа. К примеру: понимание целевой аудитории конкурентов; выявление уникальных предложений и преимуществ; анализ пользовательского опыта (UX) на сайтах конкурентов, оценка структуры, дизайна и функциональности – что имеет наибольшую важность для данного проекта

Провести поиск сайтов конкурентов через поисковые системы, по ключевым словам, связанным с проектом. Выбрать 2-4 сайта среди первых, представленных в результатах.

Провести анализ пользовательского опыта (UX). Оценка навигации и доступности информации. Оценка дизайна и визуальных элементов. Проверка скорости загрузки страниц и отображения на мобильных устройствах.

Выявить ключевые особенности и тренды. На основе собранной информации выделить ключевых особенностей, которые делают сайты конкурентов успешными. Это может включать: уникальный контент и предложения; инновационные функциональные возможности.

По итогу создать таблицу, в которой сравниваются различные аспекты сайтов конкурентов.

Пример таблицы представлен в Приложении [2].

На основе проведенного анализа сформировываются выводы, которые помогут в процессе верстки.

Правильный подход к анализу позволит избежать распространенных ошибок и создать качественный сайт, соответствующий современным требованиям и ожиданиям пользователей, найти оптимальный путь решения поставленной задачи на основе опыта конкурентов.

2.3 Создание спецификации проекта.

Создание спецификации проекта сайта необходимо для фиксирования целей, требований и ожиданий всех участников проекта. Спецификация служит основой для дальнейшей работы над сайтом и играет ключевую роль в согласовании между разработчиком и заказчиком.

На начальном этапе необходимо определить цели, которые должен достичь сайт. Типовые цели и задачи:

- привлечение новых потенциальных клиентов;

- увеличение продаж;

- повышение узнаваемости бренда;

- высокий уровень комфорта пользователя для повышения лояльности.

Четкое понимание целей поможет формулировать задачи, которые необходимо решить в рамках проекта.

Сбор требований – это процесс, в ходе которого разработчик взаимодействует с заказчиком, чтобы понять его ожидания и потребности. Важно задать открытые вопросы и внимательно выслушать ответы.

Ключевые аспекты, которые следует обсудить:

- функциональность сайта (например, наличие интернет-магазина, блога, форм для обратной связи);

- дизайн и пользовательский интерфейс (предпочтения по стилю, цветам, структуре).

- технические требования (выбор платформы, хостинга, интеграции с другими системами).

На основе собранных данных разрабатывается спецификация проекта. Этот документ должен быть структурированным и содержать следующие разделы:

- введение, описывающее проект и его цели;

- функциональные требования, описывающие, что сайт должен делать;

- нефункциональные требования, такие как производительность, безопасность и доступность;

- описание дизайна, включая макеты и прототипы;

- этапы разработки и сроки выполнения.

После создания спецификации необходимо провести ее обсуждение с заказчиком. Важно обеспечить понимание всех аспектов документа и убедиться, что он соответствует ожиданиям. На этом этапе могут возникнуть следующие действия:

- презентация спецификации, в ходе которой разработчик объясняет каждый раздел и отвечает на вопросы;

- обсуждение и внесение правок на основе отзывов заказчика;

- подписание спецификации, что подтверждает согласие сторон с содержанием документа.

Спецификация проекта должна быть живым документом, который может изменяться в процессе разработки. Важно установить процесс управления изменениями, который включает регулярные встречи с заказчиком для обсуждения хода работ и возможных корректировок, оформление всех изменений в спецификации, чтобы избежать недоразумений.

Создание спецификации проекта сайта и ее согласование с заказчиком – это ключевые этапы, которые определяют успех всего проекта. Четко сформулированные требования и открытое взаимодействие с заказчиком помогают избежать множества проблем и недопонимания в процессе разработки. В результате, грамотно составленная спецификация становится основой для успешного завершения проекта и достижения поставленных целей.

2.4 Создание прототипов.

Прототипирование веб-сайта позволяет визуализировать и протестировать основные идеи и концепции перед их реализацией. Данный процесс включает в себя создание различных версий дизайна, функциональности и пользовательского интерфейса для последующей оценки наиболее подходящих вариантов.

Целью прототипирования является выявление и уточнение требований к сайту на ранних этапах разработки. Это позволяет сократить время и затраты на изменения в будущем. Основные задачи прототипирования включают:

- определение структуры сайта: главные разделы и подразделы, которые будут включены в проект;

- создание пользовательских сценариев: ключевые сценарии взаимодействия пользователей с сайтом для понимания, как они могут перемещаться по страницам и использовать функционал;

- визуализация интерфейса: эскизы или макеты страниц, которые отображают расположение элементов интерфейса, таких как кнопки, формы, меню и изображения товаров;

- получение обратной связи: мнения заказчиков, пользователей и заинтересованных сторон о прототипах для дальнейшего улучшения и доработки.

Для прототипирования сайта используются различные инструменты и методы, позволяющие создавать как низкоуровневые, так и высокоуровневые прототипы. К ним относятся:

- эскизы и бумажные прототипы: начальный этап, на котором создаются простые эскизы страниц на бумаге, что позволяет быстро генерировать идеи и вносить изменения на ранних стадиях;

- создание каркасных моделей с помощью инструментов, таких как Figma или Adobe XD, которые позволяют визуализировать структуру сайта и расположение основных элементов без детальной проработки дизайна;

- интерактивные прототипы: создание наглядных моделей, которые позволяют пользователям взаимодействовать с интерфейсом, для того, чтобы выявить потенциальные проблемы в навигации и функциональности до начала разработки.

В частности, отдельные графические элементы будущего сайта были прототипированы в Figma для последующего выбора оптимального варианта при согласовании с заказчиком. Пример процесса прототипирования отдельных графических элементов предоставлен в Приложении [3].

Прототипирование специализированного сайта по продаже кормов для животных проходило в несколько этапов:

- Исследование и анализ. На этом этапе была проведена работа по изучению конкурентов и анализу потребностей целевой аудитории. Это помогло определить ключевые функции и элементы, которые должны быть реализованы на сайте.
- Создание низкоуровневых прототипов. Сначала были разработаны эскизы для основных страниц сайта, таких как главная страница, категория товаров и страница товара. Эти прототипы фокусировались на структуре и навигации.
- Разработка высокоуровневых прототипов. На основе полученной обратной связи от пользователей и заинтересованных сторон были созданы более детализированные интерактивные прототипы. Эти прототипы включали в себя элементы дизайна, такие как цветовая палитра, шрифты и изображения.
- Тестирование и итерации. Прототипы были протестированы среди потенциальных пользователей, что позволило выявить недостатки и получить предложения по улучшению. На основе собранной информации были внесены изменения и доработки.

В результате процесса прототипирования были достигнуты следующие результаты:

- Улучшение навигации. Прототипы позволили выявить возможные проблемы в навигации сайта, что привело к более логичной и удобной структуре.
- Оптимизация пользовательского опыта. Полученная обратная связь помогла определить, какие элементы интерфейса наиболее удобны для пользователей, что улучшило общий пользовательский опыт.
- Согласование требований. Прототипы помогли согласовать требования и ожидания всех заинтересованных сторон, что снизило риск недопонимания на этапе разработки.

Прототипирование сайта помогает сократить время и затраты на внесение изменений, улучшает пользовательский опыт и позволяет создать сайт, соответствующий потребностям целевой аудитории. В следующей главе будет рассмотрен процесс разработки компонентов сайта на основе созданных прототипов.

2.5 Структура проекта.

Формирование структуры сайта включает в себя перечень страниц сайта, их организацию и иерархию, а также логику навигации и взаимодействия между различными компонентами. В данной главе будет рассмотрен процесс структурирования сайта по продаже кормов для животных, включая создание схемы сайта, описание основных разделов и их взаимосвязей.

Целью структурирования сайта является создание логичной и интуитивно понятной структуры, которая обеспечит пользователям удобный доступ к информации и функционалу. Основные задачи, которые необходимо решить на этом этапе, включают:

- определение основных разделов: выделение ключевых разделов сайта, которые должны быть представлены пользователям;

- создание иерархии страниц: установление иерархии страниц и подстраниц, чтобы обеспечить логичное и последовательное движение по сайту;

- разработка навигационной структуры: проектирование навигационного меню и других элементов, которые помогут пользователям легко находить нужную информацию;

- оптимизация пользовательского опыта: обеспечение удобства и простоты в использовании сайта для конечных пользователей.

На этом этапе разрабатывается схема сайта, представляющая собой графическое изображение структуры и иерархии страниц. Схема сайта по продаже кормов для животных должна будет содержать следующие основные разделы:

- главная страница: центральный узел, откуда пользователи могут перейти к другим разделам;

- каталог товаров: раздел, содержащий список всех доступных товаров, сгруппированных по категориям;

- корзина: раздел, где пользователи могут просмотреть добавленные товары, изменить их количество или удалить;

- контакты: раздел с информацией для связи и вариантами контактирования между клиентами и владельцами интернет-сайта.

Простая и ясная структура обеспечивает легкость навигации для пользователей, а расширение функционала, такое как представление блоков с расширенной информацией и изображениями, будет реализовано через применение интерактивных элементов, расположенных на указанных страницах сайта.

Навигационная структура сайта по продаже кормов для животных разрабатывается на основе навигационного меню, доступного на всех страницах, содержащего ссылки для перехода на любой раздел сайта, такие как "Главная", "Каталог товаров", "Корзина", "Контакты".

Также будут применены «хлебные крошки»: вспомогательные элементы навигации на страницах товара и в корзине, что позволяет пользователям легко возвращаться к предыдущим уровням навигации.

В процессе структурирования сайта особое внимание уделяется оптимизации пользовательского опыта. Основные принципы:

- логичное расположение элементов – это размещение элементов интерфейса и навигации в соответствии с ожиданиями пользователей;

- упрощение навигации – это минимизация количества кликов, необходимых для доступа к нужной информации;

- тестирование структуры – это проведение тестов с реальными пользователями для оценки удобства навигации и выявления возможных проблем.

Структурирование сайта является важным этапом, который обеспечивает логичную организацию страниц и удобную навигацию. Создание схемы сайта, определение иерархии страниц и разработка навигационной структуры позволяют пользователям легко находить необходимую информацию и улучшает общий пользовательский опыт.

2.6 Разработка компонентов.

Опираясь на ранее разработанную структуру и прототипы, следует перейти к реализации функциональных элементов сайта. Процесс разработки компонентов включает

выбор технологий, создание базовых и сложных компонентов, интеграцию с API, а также стилизацию интерфейса.

Процесс разработки начинается с создания базовых компонентов, которые будут использоваться на разных страницах сайта. К ним относятся:

- кнопки: разработка многоразового компонента кнопки с заданными стилями и поведением;

- карточки товаров: компонент, отображающий информацию о каждом товаре, включая изображение, название и цену;

- формы: компоненты для ввода данных, такие как форма оформления заказа и форма для обратной связи.

После создания базовых компонентов программист переходит к разработке более сложных элементов, которые требуют взаимодействия с состоянием и API.

Корзина – это компонент, который отображает выбранные товары и позволяет пользователям изменять количество или удалять товары. Интерактивные элементы корзины – это реализация функций фильтрации и сортировки товаров в каталоге, используя состояние и обработчики событий.

Стилизация компонентов может быть выполнена с использованием технологии «Styled-components» или «CSS Modules». Программист создает отдельные стили для каждого компонента, что позволяет достичь хорошей организации кода и избежать конфликтов стилей.

Разработка компонентов является ключевым этапом в создании сайта. Этот процесс включает в себя создание базовых и сложных компонентов, интеграцию с API и стилизацию интерфейса. Правильная реализация компонентов обеспечивает функциональность и удобство использования сайта, что в конечном итоге способствует положительному пользовательскому опыту.

2.7 Реализация маршрутизации.

Реализация маршрутизации веб-сайта на React обеспечивает навигацию между различными страницами и компонентами приложения. В данной главе будет рассмотрен процесс настройки маршрутизации с использованием библиотеки React Router. Представлены основные принципы маршрутизации, создание маршрутов и обработку параметров, а также навигацию и управление состоянием маршрутов.

Первым шагом в реализации маршрутизации является установка библиотеки React Router и импортирование необходимых компонентов из библиотеки в главный файл приложения (обычно это `App.js`), где будет происходить настройка маршрутов.

В React Router используются несколько ключевых компонентов для реализации маршрутизации:

- BrowserRouter: оборачивает приложение и позволяет использовать маршрутизацию в приложении;

- Route: определяет маршруты, связывая URL с определенным компонентом/

На этом этапе также создаются компоненты для каждой из страниц. Эти компоненты содержат логику и интерфейс, которые будут отображаться пользователям.

Для удобства пользователей важно реализовать навигацию между страницами. Это можно сделать с помощью компонента Link, который позволяет создать ссылки на другие маршруты.

После завершения настройки маршрутизации программист проводит тестирование, чтобы убедиться, что все маршруты работают корректно:

- провести проверку всех маршрутов на предмет их правильной работы;

- протестировать переходы между страницами и отображение компонентов;

- убедиться, что параметры маршрутов корректно передаются и обрабатываются.

Реализация маршрутизации является важным этапом в разработке сайта по продаже кормов для животных, обеспечивая удобное перемещение между страницами и доступ к различному контенту. Правильная настройка маршрутов, обработка параметров и навигации способствует улучшению пользовательского опыта.

2.8 Интеграция с API.

Интеграция с API позволяет сайту взаимодействовать с внешними источниками данных, обеспечивая динамическое получение информации о товарах, пользователях и других ресурсах. В этом подразделе будет рассмотрен процесс интеграции с API, включая настройку запросов, обработку ответов и управление состоянием приложения.

Для интеграции сайта выбирается подходящее API, которое предоставляет данные о товарах, их характеристиках, ценах и наличии. Важно, чтобы API поддерживал необходимые

методы запросов и предоставлял данные в формате, удобном для обработки, например, в формате JSON.

Полученные данные интегрируются в компоненты сайта. Например, в компоненте Каталог можно вызвать функцию для получения списка товаров и обновить состояние компонента.

Интеграция с API также требует обновления состояния приложения в ответ на изменения данных. Например, при добавлении товара в корзину или изменении его количества, необходимо обновить состояние как компонента, так и глобального состояния, если используется Redux или Context API.

После завершения интеграции с API следует провести тестирование, чтобы убедиться, что все запросы выполняются корректно, данные отображаются правильно, а ошибки обрабатываются должным образом. Важно проверить, как приложение реагирует на различные сценарии, такие как отсутствие интернет-соединения или ошибки на стороне сервера.

Правильная настройка запросов, обработка ответов и управление состоянием приложения обеспечивают надежность и стабильность работы сайта.

2.9 Обеспечение поддержки и обновления сайта.

Обеспечение поддержки и обновления сайта включает в себя регулярное обслуживание, исправление ошибок, обновление функционала и контента, а также адаптацию сайта к изменяющимся требованиям пользователей и технологическим стандартам. В данном подразделе проектной работы будут рассмотрены основные аспекты, связанные с поддержкой и обновлением сайта, разработанного с использованием React.

Регулярное обслуживание сайта включает в себя мониторинг его работы, анализ производительности и безопасности, а также обновление библиотек и фреймворков. Важно следить за следующими аспектами:

- обновление библиотек и фреймворков: сайты, построенные на React, требуют своевременного обновления зависимостей, использование npm или yarn позволяет легко обновлять пакеты и получать последние функции и исправления безопасности;

- проверка безопасности: регулярный аудит безопасности, включая использование инструментов для статического анализа кода, помогает выявлять уязвимости и предотвращать атаки.

Актуальность контента является ключевым фактором успешной работы сайта. Регулярные обновления помогают поддерживать интерес пользователей и повышают SEO-оптимизацию. Основные действия включают:

- добавление новых товаров: регулярное обновление ассортимента и добавление новых товаров в каталог позволяют привлекать новых клиентов и удерживать существующих;

- обновление информации: проверка актуальности информации о товарах, ценах и акциях, а также обновление статей и описаний на сайте;

- создание нового контента: публикация статей, новостей или блогов, связанных с продуктами или услугами, может повысить интерес к сайту и улучшить его видимость в поисковых системах.

Важно также предусмотреть будущие изменения для адаптации к меняющимся условиям (обновления связанного ПО, деятельность конкурентов, тренды в обществе, ожидания пользователей), следовательно, веб-сайт должен быть подготовлен к изменениям.

Мобильная оптимизация. Обеспечение адаптивного дизайна с использованием таких инструментов, как CSS Grid и Flexbox, а также использование медиазапросов для оптимизации отображения на мобильных устройствах.

Изменение функционала. Внедрение новых функций и улучшений на основе отзывов пользователей и анализа их поведения на сайте.

Создание документации. Подробная документация по коду, архитектуре приложения и процессам обновления помогает быстро ввести в курс дела любое новое заинтересованное лицо.

Регулярное обслуживание, исправление ошибок, обновление контента и адаптация к изменениям позволяют поддерживать сайт в актуальном состоянии и обеспечивать положительный пользовательский опыт. Важно помнить, что успешный сайт требует постоянного внимания и усилий со стороны разработчиков, что в конечном итоге приводит к успешному функционированию и росту.

Глава 3. Финальная часть проекта.

3.1 Создание макета сайта.

Финальный макет сайта определяет внешний вид и функциональность и необходим для окончательного согласования всех основных вопросов с заказчиком перед проведением работы по запуску рабочей версии сайта. Его формирование я провела после завершения проведения конкурентного анализа сайтов конкурентов, оформления спецификации и ее согласования с клиентом.

Первым предварительным шагом перед созданием макета было проведение конкурентного анализа. Я изучила несколько интернет-магазинов, специализирующихся на продаже кормов для животных. Основное внимание уделено следующим аспектам:

- структура и навигация: как организованы разделы и категории товаров, а также как осуществляется навигация по сайту;

- дизайн и пользовательский интерфейс: цветовые схемы, шрифты, размещение элементов и общая эстетику;

- функциональность: какие функции предлагают конкуренты, такие как фильтры, сортировка, сравнение товаров и системы отзывов.

Параллельно с конкурентным анализом проводилась работа над спецификацией, для согласования с заказчиком.

Для реализации макета использовался инструмент Figma.

В макете отражены:

- структура: каркас сайта, с определением основных разделов и навигации главной страницы, каталога, страницы товара, корзины;

- дизайн компонентов: кнопки, карточки товаров, формы и слайдер, интуитивно понятных и эстетически привлекательных.

После завершения макета было проведено обсуждение с клиентом для получения обратной связи. Коррективы не потребовались.

Макет позволил получить ясное представление будущей структуры сайта и перейти к следующим шагам.

3.2 Определение модулей сайта.

Определение модулей было одним из ключевых этапов в создании сайта, так как правильная архитектура модулей позволяет упростить разработку, тестирование и его поддержку.

При проектировании сайта я выделила несколько ключевых разделов и функциональных модулей, которые обеспечивают пользователям полный спектр услуг. Основные модули, которые я определила, включают:

- главная страница: этот модуль представляет общее представление магазина и включает баннеры, популярные товары и новинки;

- каталог товаров: модуль, который отображает список доступных категорий товаров с вложенными внутри несколькими товарами;

- корзина: модуль, который позволяет пользователям просматривать добавленные товары и переходить к оформлению заказа;

- авторизация: модуль, который позволяет пользователям авторизоваться на сайте под своим профилем;

- поиск: модуль, который обеспечивает возможность быстрого поиска товаров по ключевым словам.

Каждый из этих модулей я разбила на более мелкие функциональные компоненты, которые обеспечивают выполнение конкретных задач. Рассмотрим основные функциональные модули подробнее.

Для главной страницы я создала несколько компонентов:

- поиск по группам товарам: компонент, позволяющий пользователям быстро находить нужные товары по группе;

- список товаров: компонент, отвечающий за отображение карточек товаров с краткой информацией (изображение, название, цена);

- пагинация: компонент, который управляет навигацией между страницами каталога;

Для модуля корзины я реализовала:

- список товаров в корзине: компонент, отображающий все добавленные товары с возможностью изменения количества и удаления;

- итоговая стоимость: компонент, вычисляющий общую сумму заказа;

- кнопка оформления: компонент, позволяющий перейти к завершению покупки.

Для организации кода и упрощения разработки использовался компонентный подход, создавая отдельные файлы для каждого компонента и соответствующих стилей.

Правильная архитектура модулей и компонентов обеспечила удобство разработки сайта. Структурирование функциональных модулей позволило создать гибкую и масштабируемую платформу.

3.3 Верстка через Flexbox.

В процессе разработки было решено использовать Flexbox для создания гибких и адаптивных макетов, использование которого значительно упростило работу с различными элементами интерфейса.

Flexbox позволяет легко управлять расположением элементов внутри контейнера, а также обеспечивает адаптивность и отзывчивость дизайна. Я изучила основные свойства Flexbox, такие как display: flex, flex-direction, justify-content и align-items, что дало мне необходимую базу для реализации макета.

При создании структуры компонентов для сайта я определила, какие элементы будут использовать Flexbox.

Каждая карточка товара была обернута в flex-контейнер, что позволяло легко управлять их расположением и адаптировать их под разные размеры экрана. Flex-wrap позволил автоматический переход карточек на новую строку при уменьшении ширины окна браузера.

Для модуля корзины был использован Flexbox для выравнивания элементов формы, таких как поля ввода и кнопки. Это обеспечивало аккуратное и логичное расположение элементов.

Одним из ключевых преимуществ использования Flexbox стало создание адаптивного дизайна. Были настроены медиа-запросы, чтобы изменить свойства Flexbox в зависимости от ширины экрана.

После верстки было проведено тестирование на различных устройствах и браузерах, чтобы убедиться, что все элементы отображаются корректно и адаптивно. При изменении размеров окна браузера все макеты элемента сохраняют свое расположение и функциональность.

3.4 Добавление логики модулей и интеграция с API.

Этап добавления логики модулей и интеграции сайта с API стал решающим для обеспечения динамического взаимодействия пользователей с сайтом и позволил реализовать основные функциональные возможности, такие как получение данных о товарах, оформление заказов и управление пользовательскими сессиями.

Перед тем как приступить к интеграции с API, была определена логика каждого модуля сайта. Это включало в себя понимание того, как модули будут взаимодействовать друг с другом и какие данные они будут использовать. Основные модули, которые я выделила для логики, включают:

Модуль каталога отвечает за отображение списка товаров, фильтрацию. Было определено, что он будет получать данные о товарах из API и обновлять состояние приложения при изменениях.

Модуль корзины позволяет пользователям добавлять, изменять количество и удалять товары из корзины.

Модуль оформления заказа требует логики для сбора данных о пользователе и выбранных товарах.

Функция получения данных о товарах из API вызывалась в эффекте useEffect на странице каталога, что позволило загружать данные при первом рендере компонента.

После добавления логики модулей и интеграции с API было проведено тестирование всех функций на корректность загрузки данных о товарах, работы функциональности корзины и оформления заказа. В процессе отладки были использованы инструменты разработчика браузера для отслеживания состояния и выполнения запросов.

Процесс добавления логики модулей и интеграции с API позволил обеспечить динамическое взаимодействие пользователей с сайтом и реализовать основные функции, которые делают сайт удобным и функциональным.

3.5 Наложение стилей и улучшение пользовательского интерфейса.

На этапе финальной верстки был сделан акцент на наложении стилей и улучшении пользовательского интерфейса. Этот процесс был направлен на создание эстетически привлекательного и интуитивно понятного дизайна, который позволит пользователям легко взаимодействовать с сайтом.

Были использованы CSS-модули для некоторых компонентов, что обеспечивало изоляцию стилей и предотвращало конфликты имен.

Определение цветовой схемы соответствовало тематике интернет-магазина кормов для животных. Сочетание нейтральных и ярких цветов позволило выделить ключевые элементы интерфейса, такие как кнопки и ссылки. Шрифты были подобраны с учетом читаемости и современного стиля, что способствовало созданию приятного визуального восприятия.

Основные опции оформления:

$widthDesktopSite: 1140px;

$colorHighlightedItems: #000000;

$colorCommonDarkItems: #db9718c9.

Стилизация основных компонентов реализована, используя подход Styled Components.

Созданы стильные кнопки с плавными переходами и эффектами при наведении. Это делало их более заметными и привлекательными для пользователей.

Особое внимание уделено стилизации формы, чтобы она была интуитивно понятной и удобной для пользователей. Поля ввода были оформлены с учетом современного дизайна, добавлены подсказки и сообщения об ошибках.

Одним из ключевых аспектов финальной верстки стало обеспечение адаптивности интерфейса. Я использовала медиа-запросы в CSS для изменения стилей в зависимости от размеров экрана, чтобы компоненты автоматически перестраивались, обеспечивая удобство использования на мобильных устройствах и планшетах.

Наложение стилей и улучшение пользовательского опыта способствовали созданию удобной платформы для покупателей.

3.6 Сдача продукта.

Процесс сдачи включал в себя несколько ключевых шагов, направленных на уверенность заказчика в качестве и функциональности продукта.

Во время встречи была проведена демонстрация интернет-магазина. Подробно рассказано о каждом модуле, включая каталог товаров, корзину и процесс оформления заказа. Заказчик мог увидеть, как работает сайт, и оценить его интерфейс и пользовательский опыт.

После демонстрации на сессии для вопросов и ответов заказчик задавал вопросы по поводу доработки дизайна.

Промежуточная сдача специализированного интернет-магазина по продаже кормов для животных заказчику прошла успешно. Задачи приняты к реализации.

Заключение.

В ходе выполнения дипломной работы была изучена теоретическая часть основ создания веб-сайтов на React и достигнута ключевая практическая задача по создания готового продукта.

Создание веб-сайтов на React обладает рядом преимуществ, таких как высокая производительность, возможность создания компонентов и простота в управлении состоянием сайта. Практическое исследование подтвердило, что с использованием React можно эффективно разрабатывать интерактивные интерфейсы, которые удовлетворяют требованиям пользователей.

Проведенное исследование позволило выявить и реализовать основные этапы создания промежуточного и функционального веб-сайта, в условиях отсутствия на момент разработки товара для реализации, включая определение и выполнение требований заказчика, прототипирование и интеграцию с API. Результаты показали, что несмотря на отсутствие медиаконтента, возможно создать функциональный и привлекательный сайт, который можно доработать в будущем. Важным итогом стало успешная проверка всех функций сайта, что подтвердило его работоспособность и соответствие требованиям заказчика.

Практическая значимость данной работы заключается в том, что она предоставляет эффективное решение для компаний, которым необходимо запустить онлайн-магазин в короткие сроки, даже при отсутствии готового контента. Рекомендации по дальнейшим действиям включают активное использование собранного функционала и последующее добавление медиаконтента по мере его готовности. Планы на дальнейшие исследования могут включать изучение методов оптимизации производительности приложения и анализ пользовательского опыта.

В процессе работы была достигнута цель проекта — разработка веб-сайта для заказчика, который отвечает современным требованиям и способен функционировать в условиях отсутствия медиаконтента. Все задачи, поставленные в начале работы, были успешно выполнены, что подтверждает актуальность выбранной темы. Гипотеза о возможности создания полноценного интернет-магазина без готового контента была доказана.

В качестве рекомендаций по совершенствованию объекта исследования можно выделить необходимость внедрения системы управления контентом для более удобного добавления и редактирования товаров. Кроме того, стоит рассмотреть возможность интеграции инструментов аналитики для отслеживания поведения пользователей и улучшения функционала сайта на основе полученных данных. Также полезно будет

исследовать возможности внедрения технологий адаптивного дизайна для оптимизации пользовательского опыта на мобильных устройствах.

Таким образом, данная работа продемонстрирует мои компетенции по разработке веб-приложений с использованием библиотеки React для JavaScript, но и практическое решение для бизнеса, стремящегося к быстрому выходу на рынок в условиях ограниченных ресурсов.

Список литературы

1. Гусев, И. А. React: создание пользовательских интерфейсов. Санкт-Петербург: Питер, 2019. 320 с.

2. Кузнецов, П. А. Программирование на React. Москва: Вильямс, 2022. 295 с.

3. Дьяков, С. В. Веб-разработка на JavaScript. Екатеринбург: Уральское издательство, 2021. 180 с.

4. Барабанов, А. В. Основы разработки веб-приложений. Москва: БХВ-Петербург, 2020. 256 с.

5. Никифоров, Е. В. Основы проектирования веб-сайтов. Новосибирск: Сибирское университетское издательство, 2018. 210 с.

6. Петров, А. С. "Интеграция API в веб-приложения." Вестник программирования 10, номер 1 (2020): 12-18.

7. Тихонов, С. А. "Прототипирование веб-приложений." Информационные технологии и системы 7, номер 3 (2022): 99-105.

8. React. "Документация по React." Доступно по ссылке: https://reactjs.org/docs/getting-started.html. (Дата обращения: 30.11.2024).

9. MDN Web Docs. "HTML: Основы." Доступно по ссылке: https://developer.mozilla.org/ru/docs/Web/HTML. (Дата обращения: 30.11.2024).

10. W3Schools. "CSS Tutorial." Доступно по ссылке: https://www.w3schools.com/css/. (Дата обращения: 30.11.2024).

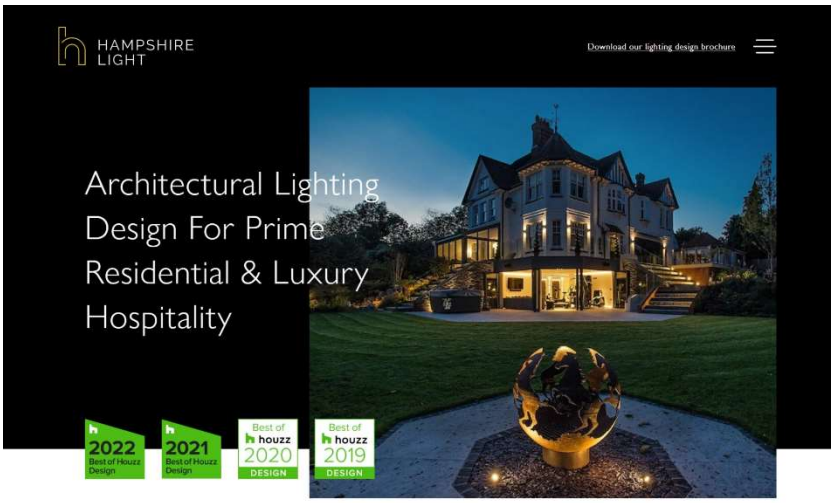
Приложения

Приложение 1.

Информация о бренде	
Название	
Краткое описание, особенности	Производитель кормов для собак и кошек, нацелен на заботу о здоровье питомцев
Цели сайта	Донесение ценностей товара до потенциальных покупателей и продажа через интернет-магазин
Целевая аудитория	
Возраст, пол	Молодые люди и взрослые в возрасте от 18 до 45 лет, преимущественно женщины
Интересы	Владельцы домашних питомцев, которые уделяют серьезное внимание здоровью животных
Функции сайта	
Оставьте из предложенного приоритетные, остальные удалите	- Авторизация пользователя: возможность создания аккаунта для удобства покупок. - Каталогизация товаров: распределение кормов по типам и линейкам. - Интернет-магазин: возможность добавления товаров в корзину и оформления заказов.

Контент	
Что необходимо разместить на сайте?	Категории товаров, характеристики товаров, описание, цены.
Создание, продвижение, поддержка	
Каков ваш бюджет на разработку сайт	Мы планируем выделить бюджет в размере 50000 рублей на начальную разработку сайта
Каковы ваши сроки на разработку сайта	Запустить сайт в течение 2 месяцев
Как вы планируете продвигать сай	Планируем использовать SEO, контекстную рекламу и активное присутствие в социальных сетях для привлечения клиентов
Как вы планируете поддерживать и обновлять сайт после его запуска	Интересует поддержка
Референсы	
Примеры сайтов, которые нравятся / визуальные образы	Подбор шрифтов и логотипа – из ранее разработанного (фото прилагается). Общая концепция дизайна и подбор цветов – из понравившегося сайта в другой сфере (скрин-шот прилагается).

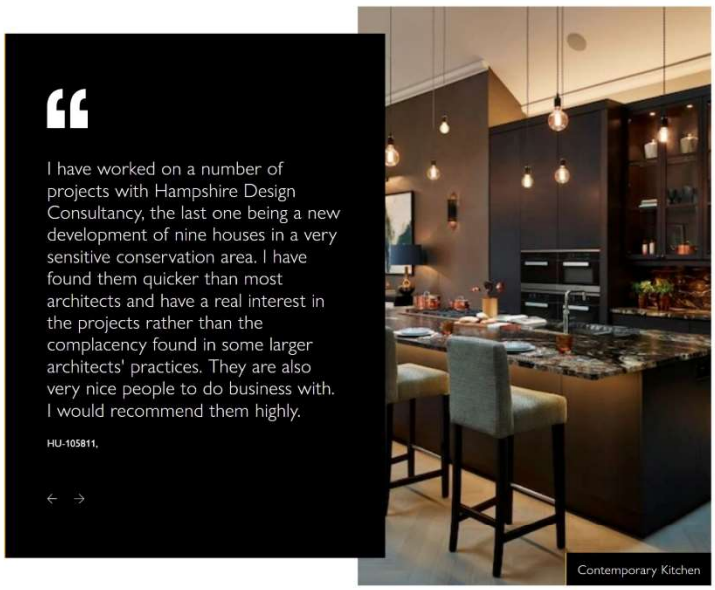

А ВЕЛЕС ХАН
СУХОЙ КОРМ ДЛЯ
СОБАК И КОШЕК
ПРЕМИУМ И СУПЕР
ПРЕМИУМ КЛАССА

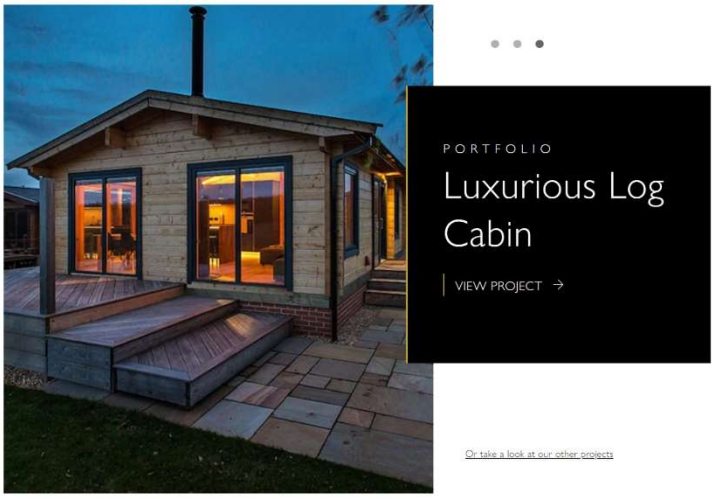


HAMPSHIRE LIGHT

Award-Winning Lighting Design Trusted By Professionals

For the last 20 years, our lighting design team have transformed high-end spaces using unrivalled passion for creative and technical excellence. With our complete end-to-end consultative process, expertise in control systems and detailed plans, Hampshire Light are the number one choice for leading professionals throughout Hampshire and surrounding counties.





Приложение 2.

Результаты конкурентного анализа			
Сайты и аспекты	**A: original.triol.pet**	**B: goldtakin.ru**	**C: fava-pets.ru**
Цели сайта	Привлечение клиентов наиболее емкого среднего сегмента	Продажи оптом и мелким оптом, для перепродавцов	Повышение рейтинга своего производства
Целевая аудитория	Широкий слой населения	Владельцы зоомагазинов, мелкооптовые покупатели	Премиум-сегмент
Дизайн	Лаконичный с ярким оформлением	Простой, слабо проработанный с точки зрения дизайна	Современный, с большим количеством интерактивных элементов
Навигация	Одностраничный сайт	Простая. Частично не функционирует	Простая, с дополнительными контекстными путями переходов
Функциональность	Поиск магазинов на интерактивной карте или списком	Переход на страницу в «Озоне» и форма заявки на оптовую закупку	Возможность просмотра подробной информации
Контент	Краткое описание товаров, краткая презентация производителя	Ограниченное количество статей	Презентации и описания вариантов товара
SEO и продвижение	Слабая оптимизация	Средняя оптимизация,	Сильная SEO-стратегия, активно

		требует улучшений	продвигается
Скорость загрузки	Быстрая загрузка страниц	Загрузка средняя, можно оптимизировать	Быстрая загрузка, хорошо оптимизирована
Мобильная версия	Адаптивный дизайн, удобен на мобильных устройствах за счет простоты	Мобильная версия требует доработки	Проработанная адаптация сайта
Уникальные предложения	Товары привлекательно и удобно оформлены	Нацеленность на оптовые продажи	Акцент на высочайшее качество товаров и отзывы

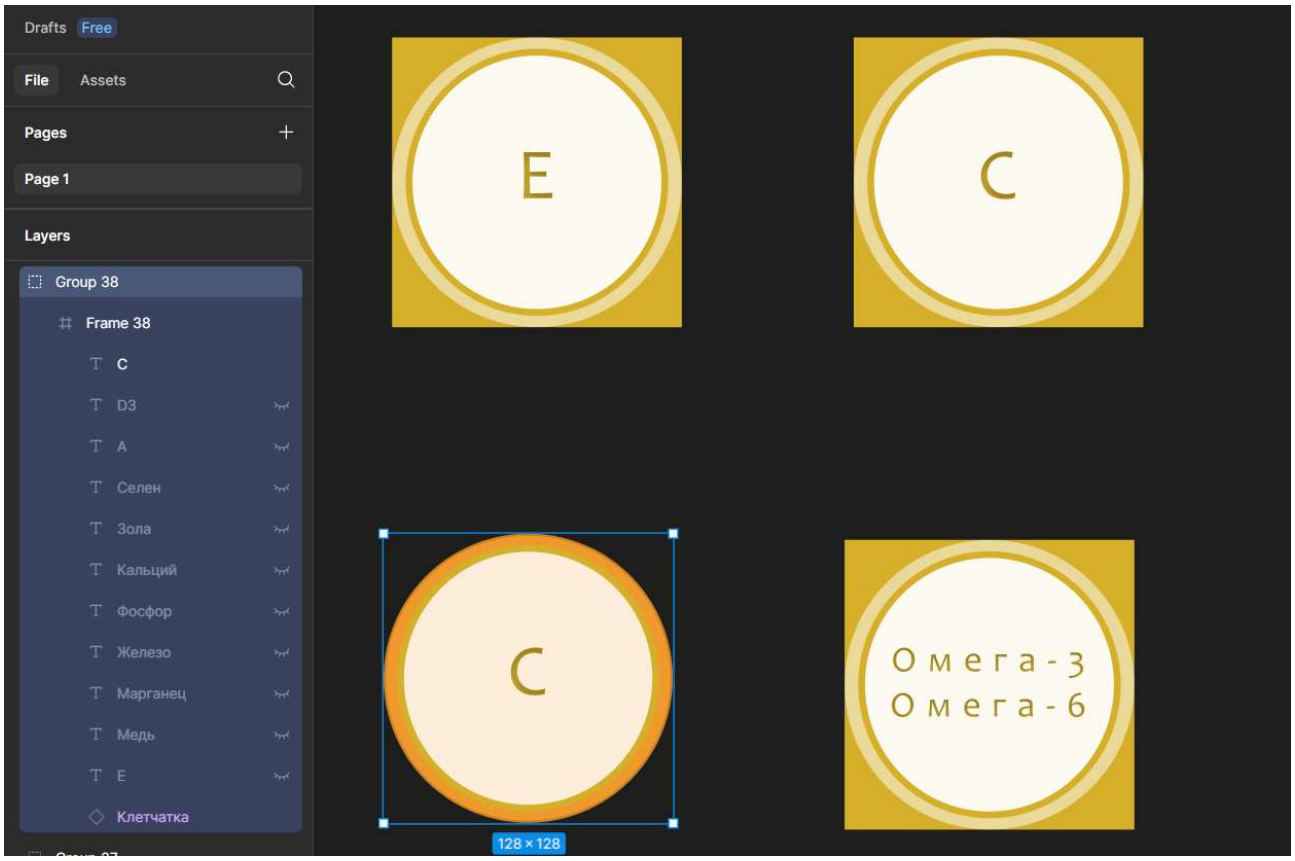
Вывод

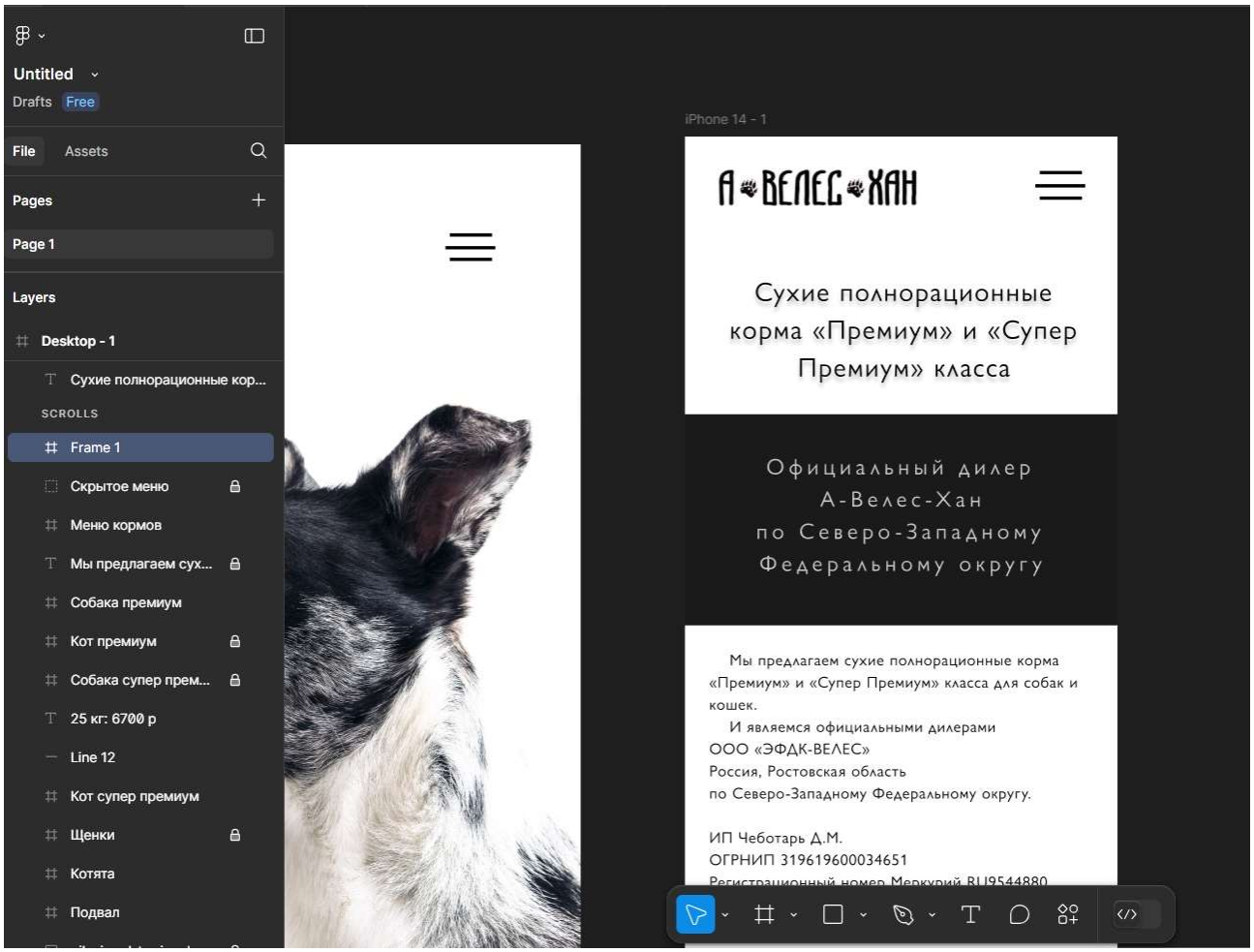
Сайт А представляет собой пример самого распространенного типа коммерческих сайтов. Он делает акцент на самом товаре, его представлении и упаковке, нацелен на широкую аудиторию. Современные тренды в организации сайта используются слабо.

Сайт В нацелен на оптовые и мелкооптовые продажи. Крайне прост и недоработан, так как целей на создание сложной современной структуры или проработанного дизайна не ставилось.

Сайт С выделяется современным дизайном с обилием анимированных элементов, продуманным визуальным стилем.

Приложение 3.




Untitled
Drafts Free
File Assets
Pages
Page 1
Layers
Desktop - 1
Сухие полнорационные кор...
SCROLLS
Frame 1
Скрытое меню
Меню кормов
Мы предлагаем сух...
Собака премиум
Кот премиум
Собака супер прем...
25 кг: 6700 р
Line 12
Кот супер премиум
Щенки
Котята
Подвал
iPhone 14 - 1
А-ВЕЛЕС-ХАН
Сухие полнорационные корма «Премиум» и «Супер Премиум» класса
Официальный дилер А-Велес-Хан по Северо-Западному Федеральному округу
Мы предлагаем сухие полнорационные корма «Премиум» и «Супер Премиум» класса для собак и кошек.
И являемся официальными дилерами
ООО «ЭФДК-ВЕЛЕС»
Россия, Ростовская область
по Северо-Западному Федеральному округу.
ИП Чеботарь Д.М.
ОГРНИП 319619600034651

Приложение 4.

Состав кода:

App.js

CartPage.js

CatalogPage.js

Footer.js

Header.js

MainPage.js

ProductItem.js

productsData.js

RegistrationPage.js

Contact.js18:28

Отдельные файлы с кодом:

App.js

```
import "./style.scss";
import { Helmet } from "react-helmet";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";

import MainPage from "./components/MainPage";
import CatalogPage from "./components/CatalogPage";
import CartPage from "./components/CartPage";
import RegistrationPage from "./components/RegistrationPage";
import Contact from "./components/Contact";



function App() {
  return (
```

```
      <>
        <Helmet>
          <link rel="preconnect" href="https://fonts.googleapis.com" />
          <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
          <link
href="https://fonts.googleapis.com/css2?family=Mulish:ital,wght@0,200..1000;1,200..1000&family=Poiret+One&display=swap"
            rel="stylesheet"
          />
        </Helmet>
        <Router>
          <Routes>
            <Route path="/" element={<MainPage />} exact />
            <Route path="/catalog" element={<CatalogPage />} />
            <Route path="/cart" element={<CartPage />} />
            <Route path="/registration" element={<RegistrationPage />} />
            <Route path="/contact" element={<Contact />} />
          </Routes>
        </Router>
      </>
    );
  }

  export default App;
```

CartPage.js

```
import React, { useContext, useState, useEffect } from "react";
import { Link } from "react-router-dom";
import Header from "./Header";
import Footer from "./Footer";
import { CartContext } from "../contexts/CartContext";
import products from "./productsData";

import { ReactComponent as CloseIcon } from "../img/close_icon.svg";

const CartPage = () => {
  const { cartItems, removeFromCart, updateCartItemQuantity, clearCart } =
useContext(CartContext);
  const [totalPrice, setTotalPrice] = useState(0);

  // Определение общей стоимости товаров в корзине
  useEffect(() => {
    let total = 0;
    cartItems.forEach((item) => {
      const product = products.find((product) => product.id === item.id);
      total += product.price * item.quantity;
    });
    setTotalPrice(total);
  }, [cartItems]);


  const handleQuantityChange = (event, productId) => {
    const newQuantity = parseInt(event.target.value);
    updateCartItemQuantity(productId, newQuantity);
  };


  const handleRemoveItem = (productId) => {
    removeFromCart(productId);
```

```
};


const handleClearCart = () => {
  clearCart();
};

return (
  <div className="box-content">
    <Header />
    <content className="content">
      <div className="head center">
        <h1 className="head_title">Корзина</h1>
      </div>
      <div className="cart-box center">
        {cartItems.length === 0 ? (
          <h2>Ваша корзина пуста</h2>
        ) : (
          <div className="cart-box__left">
            <div className="cart-box__products">
              {cartItems.map((item) => {
                const product = products.find((product) => product.id === item.id);
                if (product)
                  return (
                    <div className="cart-box__product_card" key={item.id}>
                      <div className="cart-box__card_img">
                        <img src={product.imageUrl} alt="Product image" />
                      </div>
                      <div className="cart-box__card_text">
                        <h4 className="cart-box__card_title">{product.title}</h4>
                        <p className="cart-box__card_content">
                          Стоимость: 
                          <span className="cart-box__card_price cart-box__card_value">
                            {product.price.toFixed(0)} руб.
                          </span>
```

```
                </p>
                <p className="cart-box__card_content">
                  Вкус: 
                  <span className="cart-box__card_value">
                    {product.taste}
                  </span>
                </p>18:24


          <label htmlFor="quantity" className="cart-box__card_content">
                    Количество:
                  </label>
                  <input
                    id="quantity"
                    type="number"
                    min="1"
                    value={item.quantity}
                    onChange={(e) => handleQuantityChange(e, item.id)}
                  />
                </div>
                <a className="cart-box__close_icon" onClick={() =>
handleRemoveItem(item.id)}>
                  <CloseIcon />
                </a>
              </div>
            );
          })}
        </div>
        <div className="cart-box__buttons">
          <div className="cart-box__shopping_button">
            <a className="cart-box__shopping_button_title" onClick={handleClearCart}>
              Очистить корзину
            </a>
          </div>
          <div className="cart-box__shopping_button">
```

```
        <Link className="cart-box__shopping_button_title" to="/catalog">
          Продолжать покупки
        </Link>
      </div>
    </div>
  </div>
)}
<div className="cart-box__right">
  <div className="cart-box__shipping-form">
    <h4 className="cart-box__shipping-title">АДРЕС ДОСТАВКИ</h4>
    <input
      className="cart-box__shipping-field"
      type="text"
      id="country"
      placeholder="Country"
      value="Россия"
      required
    />
    <input
      className="cart-box__shipping-field"
      type="text"
      id="territory"
      placeholder="Регион"
      required
    />
    <input
      className="cart-box__shipping-field"
      type="text"
      id="postcode"
      placeholder="Индекс"
      required
    />
    <div className="cart-box__shipping_button">
      <a className="cart-box__shipping_button_title" href="#">
        Отправить
```

```
            </a>
          </div>
        </div>
        <div className="cart-box__checkout-box">
          <div className="cart-box__subtotal">
            <div>Сумма:</div>
            <div>{totalPrice.toFixed(0)} руб.</div>
          </div>
          <div className="cart-box__grandtotal">
            <div>Итого:</div>
            <div className="cart-box__totalprice">{totalPrice.toFixed(0)} руб.</div>
          </div>
          <hr className="cart-box__checkout-line" />
          <div className="cart-box__checkout-button">
            <a className="cart-box__checkout_button_title" href="/#">
              Оформить заказ
            </a>
          </div>
        </div>
      </div>
    </content>
    <Footer />
  </div>
 );
}

export default CartPage;
```

CatalogPage.js

```
import React, { useState } from "react";

import Header from "./Header";
```

```
import ProductItem from "./ProductItem";
import { HandySvg } from "handy-svg";
import drop_down_arrow from "../img/drop-down_arrow.svg";

import products from "./productsData";



function pagination(array, pageType, pageNumber) {
return array.slice((pageNumber - 1) * pageType, pageNumber * pageType);
}

function CatalogPage() {

const [currentPage, setCurrentPage] = useState(1);
const pageType = 9;
const [selectedTypes, setSelectedTypes] = useState([]);
const handlePageChange = (pageNumber) => {
setCurrentPage(pageNumber);
};


const handleTypeChange = (type) => {
if (selectedTypes.includes(type)) {
setSelectedTypes(selectedTypes.filter((s) => s !== type));
} else {
setSelectedTypes([...selectedTypes, type]);
}
};


const displayedProducts = pagination(
products.filter(
(product) =>
selectedTypes.length === 0  selectedTypes.includes(product.type)
```

```
),
pageType,
currentPage
);


const totalPages = Math.ceil(
products.filter(
(product) =>
selectedTypes.length === 0  selectedTypes.includes(product.type)
).length / pageType
);

return (
<div className="box-content">
<Header />
<content className="content">
<div className="head center">
<h1 className="head_title">Корма</h1>
<nav className="breadcrums">
<a href="/" className="breadcrums__link">
Главная
</a>
<a
href="/catalog"
className="breadcrums__link breadcrums__link__highlight"
>
Корма
</a>
</nav>
</div>
<div className="filter-sorter center">

<div className="sorter">
```

```
<details className="sorter__details">
<summary className="sorter__summary">
<span className="sorter__title">Фильтр</span>
<HandySvg src={drop_down_arrow} width="11" height="6" />
</summary>
<div className="sorter__content">
<div className="sorter__item">
<input
type="checkbox"
className="sorter__checkbox"
id="sorter__type-dog"
defaultChecked="true"
onChange={() => handleTypeChange("dog")}
checked={selectedTypes.includes("dog")}
/>
<label className="sorter__label" htmlFor="sorter__type-dog">
Собаки
</label>
</div>
<div className="sorter__item">
<input
type="checkbox"
className="sorter__checkbox"
id="sorter__type-cat"
defaultChecked="true"
onChange={() => handleTypeChange("cat")}
checked={selectedTypes.includes("cat")}
/>
<label className="sorter__label" htmlFor="sorter__type-cat">
Кошки
</label>
</div>
<div className="sorter__item">
<input
type="checkbox"
```

```
className="sorter__checkbox"
id="sorter__type-baby"
defaultChecked="true"
onChange={() => handleTypeChange("baby")}
checked={selectedTypes.includes("baby")}
/>
<label className="sorter__label" htmlFor="sorter__type-baby">
Малыши
</label>
</div>

</div>
</details> >
</div>
</div>
<div className="products spec_catalog center">
<div className="product_items product_items_spec">
{displayedProducts.map((product) => (
<ProductItem
key={product.id}
id={product.id}
title={product.title}
description={product.description}
price={product.price}
imageUrl={product.imageUrl}
/>
))}
</div>

<div className="catalog_nav">
<nav className="catalog_nav__box">
<div className="catalog_nav__pagination">
{Array.from(
{ length: totalPages },
(_, index) => (
```

```
<a
key={index}
onClick={() => handlePageChange(index + 1)}
className="catalog_nav__link"
>
{index + 1}
</a>
)
)}
</div>
</nav>
</div>
</div>
</content>

</div>
);
}

export default CatalogPage;
```

Header.js

```
import React, { useContext } from 'react';
import { Link } from "react-router-dom";

import { CartContext } from "../contexts/CartContext";

import { HandySvg } from 'handy-svg';
import header_logo from "../img/header_logo.svg";

import { ReactComponent as Menu_icon } from "../img/menu_icon.svg";
import { ReactComponent as User_icon } from "../img/user_icon.svg";
import { ReactComponent as Cart_icon } from '../img/cart_icon.svg';
```

```
function Header() {
    // Получение количества товаров в корзине из контекста
    const { cartItems } = useContext(CartContext);
    const totalItems = cartItems.reduce((total, item) => total + item.quantity, 0);

    return (
        <header className="header center">
            <div className="header_left">
                <Link className="logo" to="/">
                    <img src={header_logo} className="logo" alt="Letter b logo" />
                </Link>

            </div>
            <div className="header_right">
                <div className="main_menu">
                    <label htmlFor="main_menu_button" className="main_menu_button">
                        <Menu_icon className="menu_icon" />
                    </label>
                    <input type="checkbox" id="main_menu_button" defaultChecked="false" />
                    <div className="main_menu_bar">
                        <h4 className="main_menu_title">Меню</h4>
                        <ul>
                            <li>
                                <Link className="main_section" to="/catalog">Корма для собак</Link>

                            </li>
                            <li>
                                <Link className="main_section" to="/registration">Личный кабинет</Link>

                            </li>
                            <li>
                                <Link className="main_section" to="/cart">Корзина</Link>

                            </li>
                            <li>
```

```
                        <Link className="main_section" to="/contact">Контакты</Link>

                    </li>
                  </ul>
                </div>
              </div>
              <Link className="user" to="/registration">
                <User_icon className="user_icon" />
              </Link>
              <Link className="cart" to="/cart">
                <Cart_icon className="cart_icon" />
                {totalItems > 0 && <span className="cart_count">{totalItems}</span>}
              </Link>
            </div>
          </header>
    );
}

export default Header;
```

MainPage.js

```
import { Link } from "react-router-dom";

import Header from "./Header";
import Footer from "./Footer";
import ProductItem from "./ProductItem";
import top_brand_img from "../img/top_brand_img.jpeg";



import { ReactComponent as FeatureDelivery_icon } from "../img/feature_icon1.svg";
import { ReactComponent as FeatureDiscount_icon } from "../img/feature_icon2.svg";
import { ReactComponent as FeatureQuality_icon } from "../img/feature_icon3.svg";
```

```
import products from "./productsData"; // Импорт массива продуктов

import "../style.scss";

function MainPage() {
return (
<div className="box-content">
<Header />
<content className="content">
<div className="top_brand"><div className="top_brand_title">
<div className="top_brand_text">
<h1 className="tb_large">
Сухие полнорационные корма
«Премиум» и «Супер Премиум» класса</h1>
<h3 className="tb_small">Официальный дилер А-Велес-Хан по Северо-Западному
Федеральному
округу</h3>
</div>
</div>
<div className="top_brand_img">
<img className="top_brand_pic" src={top_brand_img} alt="dog" width="70%" />
</div>

</div>
<div className="offers center gallery-wrapper">
<div className="small_offers">
<div className="offer offer__protein">
<h4 className="offer_subtitle">Энергия и наращивание мышц, костей,
суставов</h4>
</div>
<div className="offer offer__fat">
<h4 className="offer_subtitle">Источник энергии</h4>
</div>
<div className="offer offer__fiber">
<h4 className="offer_subtitle">Нормализует работу кишечника</h4>
```

```
</div>

<div className="offer offer__ash">
<h4 className="offer_subtitle">Источник минеральных веществ</h4>

</div>
<div className="offer offer__calcium">
<h4 className="offer_subtitle">Улучшает формирование зубов и костной ткани</h4>

</div>
<div className="offer offer__phosphorus">
<h4 className="offer_subtitle">Способствует мышечной активности и свертыванию
крови</h4>

</div>
<div className="offer offer__iron">
<h4 className="offer_subtitle">Строительный материал для клеток организма, профилактика
анемии</h4>

</div>
<div className="offer offer__copper">
<h4 className="offer_subtitle">Поддерживает целостность костей и кровеносных
сосудов</h4>

</div>
</div>
</div>
<div className="products center">
<p className="products_title">Наши корма</p> > Любимая Шахты: <div
className="product_items">
{products.slice(6).map((product) => (
<ProductItem
key={product.id}
id={product.id}
```

```
title={product.title}
description={product.description}
price={product.price}
imageUrl={product.imageUrl}
/>
))}
</div>
<div className="products_button">
<Link className="browse_button" to="/catalog">
Все продукты
</Link>
</div>
</div>
<section>
<div className="features center">
<figure>
<div className="feature">
<FeatureDelivery_icon className="feature_icon" />
<h4 className="feature_title">Быстрая доставка</h4>
<p className="feature_description">
Наша команда профессионалов гарантирует своевременную и аккуратную доставку прямо к вашему порогу.
</p>
</div>
</figure>
<figure>
<div className="feature">
<FeatureDiscount_icon className="feature_icon" />
<h4 className="feature_title">Гибкая система скидок</h4>
<p className="feature_description">
Скидки на определённые товары и индивидуальные предложения для владельцев породистых кошек и собак, а также для постоянных клиентов.
</p>
</div>
</figure>
```

```
<figure>
<div className="feature">
<FeatureQuality_icon className="feature_icon" />
<h4 className="feature_title">Гарантия качества</h4>
<p className="feature_description">
Гарантируем высокое качество наших кормов для домашних животных. Мы тщательно проверяем каждый ингредиент и следим за соблюдением всех стандартов производства
</p>
</div>
</figure>
</div>
<section>
<div className="subscride_info center">

<div className="card-container">
<div className="card"><a href="/#">
<div className="card--display"><i className="material-icons">Составы</i>

</div>
<div className="card--hover">
<p><span className="card-span">сухой белок мяса (30%), рис, свёкла, морковь, жир индейки, растительная
клетчатка, рыбная мука, сушёный белок яйца, рыбий
жир, минеральные вещества, витаминный комплекс,
экстракт бархатцев, экстракт
Юкки Шидигера, экстракт расторопши, глюкозамин и хондроитин,
топинамбур, морские водоросли, растительное масло, экстракт розмарина,
пробиотик</span>
</p>
</div></a>
<div className="card--border"></div>
</div>
</div>
<div className="card-container">
<div className="card"><a href="/#">
```

```
<div className="card--display"><i className="material-icons">Витамины и
микроэлементы</i>

</div>
<div className="card--hover">

<p>
витамин А – 28000 МЕ/кг, витамин D3 – 1000 МЕ/кг, витамин Е
– 380 мг/кг, витамин С – 140 мг/кг, витамины группы В,
железо – 175 мг/кг, йод – 1,4 мг/кг, медь – 25 мг/кг,
марганец – 42,5 мг/кг, цинк – 250 мг/кг, селен – 0,24 мг/кг.
</p>

</div></a>
<div className="card--border"></div>
</div>
</div>
<div className="card-container">
<div className="card"><a href="/#">
<div className="card--display"><i className="material-icons">Питательный состав</i> >
Любимая Шахты: </div>
<div className="card--hover">

<p>
сырой белок 27%, сырой жир 15%, сырая клетчатка 2,7%, сырая
зола 7,2%, влажность 10%, кальций 1,5%, фосфор 0,9%, омега-6
жирные кислоты 3,4%, омега-3 жирные кислоты 0,3%, метионин и
цистин.
</p>

</div></a>
<div className="card--border"></div>
</div>
</div>
</div>
```

```
</section>
</section>
</content>
<Footer />
</div>
);
}

export default MainPage;
```

Header.js

```
import React from 'react';
import { Link } from "react-router-dom";


function Footer() {
  return (
    <footer className="footer">
      <section>
        <div className="contacts center">
          <p className="copyright_text">© 2024 А-ВЕЛЕС-ХАН</p>
          <a className="contact-text_link" href="tel:+79118357578"
          >+7 (911) 835-75-78 </a>
          <div className="social_nets_bar">
            <p class="footer-text">
              Ленинградская область, пос. Янино-1, ул. Заводская, уч. 17
            </p>

          </div>
        </div>
      </section>
    </footer>
  );
}
```