Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт педагогики, психологии и социологии
Кафедра современных образовательных технологий

УТВЕРЖДАЮ
Заведующий кафедрой
_____ И. А. Ковалевич
«_____» _____ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 – Прикладная информатика

Разработка веб-сайта для магазина розничной торговли

Руководитель старший преподаватель _____ О.В. Шайдурова

Научный консультант доцент, канд.техн.наук _____ И.А. Ковалевич

Выпускник _____ А.И. Марьясов

Красноярск 2018

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка веб-сайта для магазина розничной торговли» содержит 64 страницы текстового документа, 19 использованных источников, 36 рисунков, 2 приложения.

ВЕБСАЙТ, ИНТЕРНЕТ, СТРОИТЕЛЬНЫЕ МАТЕРИАЛЫ, АДАПТИВНЫЙ ДИЗАЙН, HTML, FLEX BOX, CSS GRID.

Цель бакалаврской работы: разработка веб-сайта для магазина розничной торговли.

Объект бакалаврской работы: магазин строительных материалов «ДомоХозяин».

Задачи бакалаврской работы:

1 Охарактеризовать веб-сайты магазинов строительных материалов.

2 Описать работу магазина строительных материалов «ДомоХозяин» и сформировать требования к проекту веб-сайта.

3 Описать основные этапы и методы проектирования веб-сайтов.

4 Провести исследование технологий и модулей, использующихся для современной верстки веб-сайтов и выбрать оптимальные пути адаптации сайта под большинство устройств.

5 Разработать веб-сайт для магазина строительных материалов «ДомоХозяин».

В результате бакалаврской работы был спроектирован и разработан веб-сайт, адаптированый к различным устройствам, содержащий сведения об организации, каталог товаров, поиск товара по каталогу и возможность регистрации посетителя.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цифровые технологии не стоят на месте, все больше и больше внедряясь во все сферы человеческой жизни. Современный человек не может обойтись без цифровых технологий, которые позволяют ему узнать множество интересного, могут обучить чему-то новому, развлечь в необходимый момент, и, самое главное, современные цифровые технологии значительно упрощают людям жизнь. В мире постоянно появляются новые технологии: это и всевозможные гаджеты и различная современная техника, большой выбор развлекательных устройств, необычные приложения для учебы, разнообразные интернет-сайты.

Нельзя исключить цифровые технологии и из торговой сферы. В век интернет-магазинов, интернет-порталов и аукционов, для любого коммерческого предприятия, целью которого является привлечение новых клиентов, важно иметь свой веб-сайт, удобный для пользователей, интересный и отражающий деятельность магазина.

Для успеха в коммерческой деятельности нужно четко понимать, какие цели должен преследовать веб-сайт и какие задачи решать. Некоторым организациям требуется сайт-визитка, который будет давать общую информацию о деятельности данной компании или, как это часто бывает в наше время, он просто нужен для поддержания имиджа компании, ведь современные тенденции таковы, что сайт иметь просто необходимо. Другим организациям больше подойдет сайт-презентация – это сайт, дающий более полное представление о компании и ее деятельности, а также предоставляющий определенную полезную информацию для посетителей. Еще бывают сайты-каталоги – где посетитель может получить полную информацию о производимой или продаваемой продукции, ознакомиться с прайс-листом или пообщаться с представителями организации. И еще более сложный вариант –

это интернет-магазин, по сути являющийся сайтом, на котором можно совершить покупку-заказ товара онлайн, очень популярное направление современного бизнеса.

На рынке строительных материалов нашего города сейчас наблюдается высокая конкуренция, появляются международные магазины, конкурировать с которыми довольно сложно. Так что проектирование хорошего, качественного веб-сайта становится приоритетной задачей для небольшого магазина.

Целью бакалаврской работы является разработка веб-сайта для магазина розничной торговли.

Задачи:

1 Охарактеризовать веб-сайты магазинов строительных материалов.

2 Описать работу магазина строительных материалов «ДомоХозяин» и сформировать требования к проекту веб-сайта.

3 Описать основные этапы и методы проектирования веб-сайтов.

4 Провести исследование технологий и модулей, использующихся для современной верстки веб-сайтов и выбрать оптимальные пути адаптации сайта под большинство устройств.

5 Разработать веб-сайт для магазина строительных материалов «ДомоХозяин».

1 Теоретические материалы по теме выпускной работы

1.1 Понятия – «интернет» и «веб-сайт»

Интернет это гигантская информационная система, подчинившая себе весь мир. Появившись в 1973 году, она довольно быстро заняла передовые позиции на рынке развлечений, поиска и хранения информации, общения, торговли, сегодня трудно представить себе жизнь без сети.

Интернет опутал планету подобно паутине, и именно этот термин дал ему официальное называние – World Wide Web (Всемирная паутина). Аббревиатуру от названия (www) можно увидеть перед каждым адресом веб-сайта. Браузер – это программа, которая делает сложную информационную структуру сети доступной для человеческого восприятия.

Интернет – это совокупность компьютерных сетей, имеющих единое адресное пространство. Информация хранится на специальных компьютерах – серверах, они же предоставляют персональным компьютерам доступ к интернету. Сервера бывают разных типов: веб-сервера, FTP для обмена файлами, почтовые, чаты и системы трансляции (радио, телевидение). Каждый компьютер имеет в сети свой идентификационный номер (IP), каждый веб-сайт – адрес. Связь с интернетом обеспечивает провайдер, с которым пользователи заключают соглашение. Несмотря на различие способов подключения, технология связи одна – это семейство протоколов TCP/IP (Transmission Control Protocol/Internet Protocol). Протокол – это правило, по которому компьютер связывается с сервером. В семье протоколов наибольшей популярностью пользуется протокол HTTP, который отвечает за работу с текстом. Любая страница в интернете содержит текст, поэтому HTTP используется при работе с любым сайтом. Не менее важны протоколы POP и SMTP, отвечающие за получение и отправку электронных сообщений, а также FTP для передачи и получения файлов. Важнейшим критерием работы интернета для любого пользователя является – скорость передачи информации. Технический прогресс развивается очень быстро, и с каждым днем работа с сетью становится все

доступнее. Количество веб-сайтов растет в геометрической прогрессии, появляется все больше возможностей для осуществления практически всех видов человеческой деятельности [1].

Интернет – это общение, развлечение, образование и торговля. Общение по сети позволяет связаться с любым пользователем в любой точке планеты. Интернет предоставляет богатый спектр развлечений: это и фильмы, и музыка, и фотографии, и личное видео, которыми делятся пользователи в различных социальных сетях или выставляют на своих сайтах. Большинство вузов, идя «в ногу» со временем, открыли своим студентам новые возможности благодаря обучению через интернет. Современные технологии позволяют обучаться тем, кто физически или по другим причинам не может присутствовать в аудиториях. Если раньше, когда дистанционное образование только появлялось, оно не всеми воспринималось всерьез, то сейчас это официально действующая система подготовки новых кадров, дающая профессиональные знания [2].

Интернет-магазины осуществляют свою торговлю используя электронные платежные системы и пользуются рекламой в сети, так как она предоставляет большие возможности для этого. Заказ через интернет можно сделать в любое время дня и ночи, и, не выходя из дома, получить свою покупку [3].

Веб-сайт это совокупность логически связанной гипертекстовой информации, оформленной в виде отдельных страниц и доступной в сети Интернет.

Подобное определение веб-сайта было правильным в начале существования Интернета, когда сеть и веб-сайты использовались в основном как развлекательная система. До конца 90-х годов веб-сайты представляли собой в основном статичные страницы. Для создания веб-сайта требовалось только знание языка гипертекстовой разметки HTML. Если же страница предоставляла какие-то программные средства это были исключительно средства, которые мог предоставить сервер, на котором расположен веб-сайт. О комфорте и красоте дизайна в то время особо не приходилось говорить [4].

Языки программирования развиваются, расширяются каналы передачи информации. Сейчас Интернет уже является самодостаточной отраслью экономики, а веб-сайты стали полноправными представительствами фирм в Интернете. Сегодня миллионы людей утром встают и «идут на работу» в Интернет, а в качестве офиса этих людей выступают веб-сайты.

Веб-сайт это совокупность программных, информационных, а также медийных средств, логически связанных между собой. По сути же веб-сайт это отражение успешности фирмы, ее лицо [5].

Веб-сайт выполняет следующие основные задачи:

- реклама продукции, услуг, идей. Правильно спроектированный и оформленный веб-сайт легко приведет клиента к заключению о необходимости покупки товара, или услуг, или идей, пропагандируемых на нем;
- продажа товаров, услуг, информации, идей. Современный человек живет в таком темпе, что у него нет большого количества времени для походов по магазинам. Возможность заказа товаров и услуг, с компьютера, а в последнее время и с телефона или планшета, значительно расширяет возможности и клиента, и продавца;
- бесплатное предоставление информации или услуг. Но в большей степени это средство привлечения посетителей к данному веб-сайту для получения, к примеру, статистической информации, либо для показа рекламы, если это рекламная площадка;
- поддержка клиентов.

Для создания веб-сайтов используется язык разметки гипертекстовых документов HTML. Суть технология HTML заключается в том, что в обычный текстовый документ вставляются управляющие символы (тэги), которые действуют по определенным законам языка html и в результате мы получаем веб-документ. Браузер при загрузке веб-документа расшифровывает теги и выводит информацию на экран [6].

1.2 Характеристика веб-сайтов магазинов строительных материалов

Торговля – одно из самых древних занятий человека. Начиная со времен натурального обмена и до наших дней, люди продают и покупают различные товары. Времена меняются, всемирная паутина теперь плотно связана с торговлей.

Открытие магазина розничной торговли нужно для того, чтобы получать прибыль путем продажи товаров конечным потребителям по розничным ценам, которые, в то же время, являются рыночными. Для привлечения клиентов администрации магазинов прибегают к разным стратегиям, основной из которых является реклама. В наше время самая основная реклама поступает через сеть и интернет.

В сфере торговли строительными материалами владельцы компаний выбирают различные направления для своих сайтов. Рассмотрим некоторые из них (не будем называть конкретные магазины, чтобы никого не компрометировать, а обозначим их: магазин 1, магазин 2 и т. д.) и сделаем относительно них определенные выводы:

1. Магазин 1 – работал раньше исключительно с оптовиками, а с недавнего времени в нем появилась возможность покупки в розницу. Кроме сайта, другой рекламы не имеет. Значит на сайт возложена вся ответственность по привлечению клиентов. Они выбрали сайт-каталог, так как другой полезной информации для посетителя он не содержит. Индексная страница сайта представлена на рисунке 1, дизайн невзрачный, шрифт мелкий и вообще масштаб выбран не очень удачно. Прайс-лист можно как скачать, так и просмотреть в режиме онлайн, однако прайсы очень редко обновляются, и реальная цена на некоторые товары сильно не соответствует представленной на сайте. Есть фотографии некоторых товаров, правда, чтобы просмотреть их, нужно открыть новое окно в браузере, что не очень удобно.

Рисунок 1 – Индексная страница веб-сайта магазина 1

Мобильная версия веб-сайта магазина 1 не предусмотрена. Заходя со смартфона, сталкиваемся с тем, что мелкий шрифт стал практически не читаем, если конечно не приблизить, но это не удобно. На рисунке 2 скриншот индексной страницы сайта с мобильного телефона.

Рисунок 2 – Индексная страница веб-сайта магазина 1 с экрана смартфона

Общее впечатление от веб-сайта магазина 1 отрицательное: мелкий масштаб, не красивый дизайн и не продуманная мобильная версия. У них достаточно низкие цены и с хороший веб-сайт мог бы увеличить прибыль.

2. Магазин 2 – достаточно крупная сеть, имеет магазины в нескольких городах, занимается как оптовой, так и розничной торговлей строительными материалами. На рисунке 3 представлена индексная страница веб-сайта магазина 2.

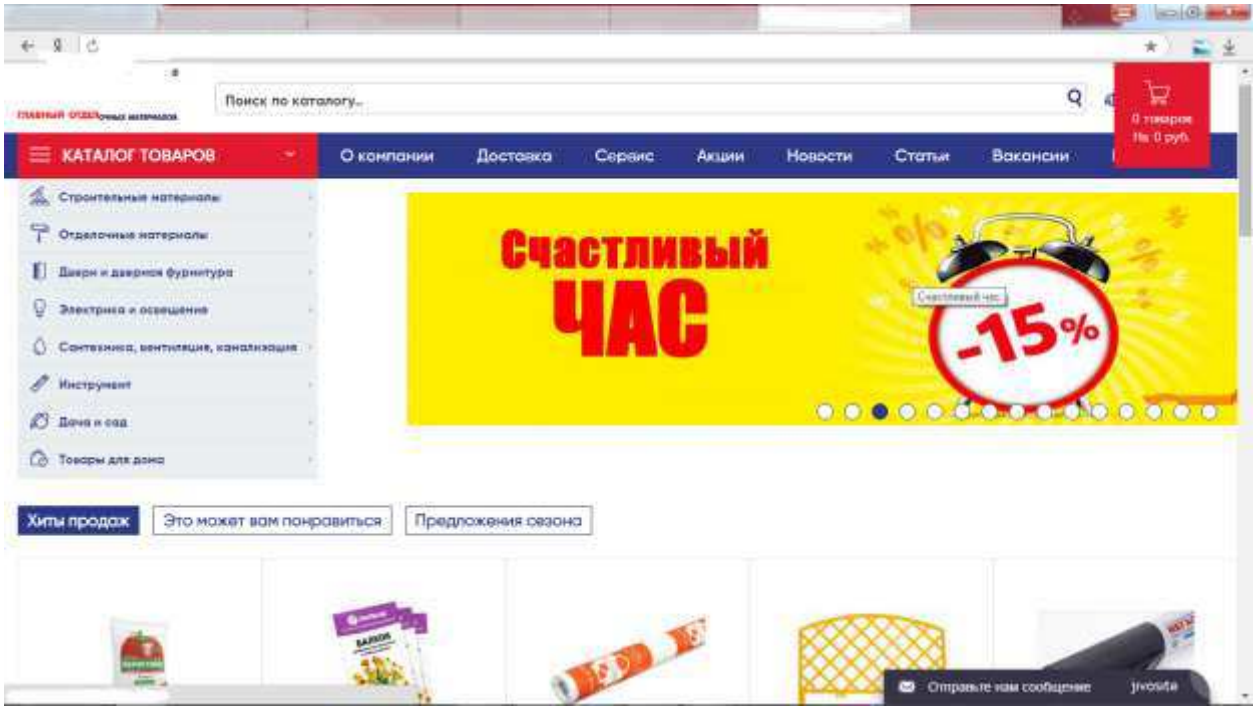

Рисунок 3 – Индексная страница веб-сайта магазина 2

Сайт достаточно мощный, не плохо проработаны навигация, поиск, потенциальному покупателю легко ориентироваться в каталоге товаров, каждый товар иллюстрирован фотографией. Дизайн достаточно прост, но все элементы расставлены грамотно. Динамичные рекламные окна. Единственный минус – медленно грузит страницы. Страница продажи квартир на сайте магазина строительных материалов кажется неуместной. Для мобильной версии сайта магазин 2 выбрал адаптивный дизайн. В принципе со смартфона тоже удобно использовать данный сайт, весь интерфейс выстроен в одну строку, теперь каждый элемент занимает весь экран, удобно читать, искать,

путешествовать по сайту. На рисунке 4 можно увидеть веб-сайт магазина 2, если зайти на него с мобильного устройства.

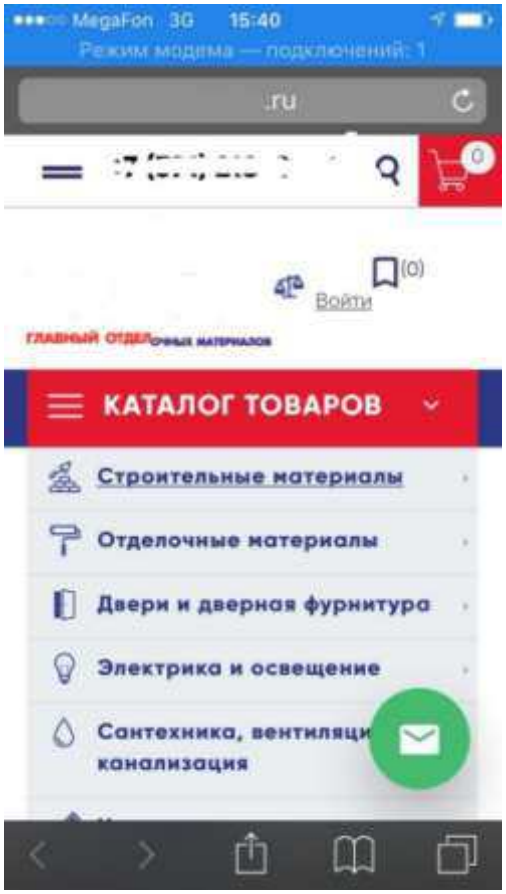

Рисунок 4 – Индексная страница веб-сайта магазина 2 с экрана смартфона

Опираясь на проведённый анализ, сделаем вывод: магазин 2, имеет достаточно неплохой сайт-каталог – где посетитель может получить полную информацию о продаваемой им продукции, ознакомиться с прайс-листом или пообщаться с представителями организации. Использование адаптивной верстки для мобильной версии сайта себя оправдывает, удобно пользоваться сайтом со смартфона.

3. Магазин 3 – международный магазин строительных материалов имеет магазины во многих странах, в большинстве городов России и несколько магазинов в нашем городе. Главная страница веб-сайта магазина 3 изображена на рисунке 5. Неплохой дизайн, есть возможность регистрации и удобный интерфейс. Еще на сайте представлено множество материала, который

привлекает посетителей. Каталог удобный и с фотографиями, но, если пройти дальше индексной страницы и нажать на каталог, ничего не происходит. Еще один минус – неудобный поиск. Грузится сайт очень долго, богатство контента играет свою роль. Для входа с мобильных устройств у магазина 3 разработана мобильная версия сайта. Так же очень удобный интерфейс, все элементы подстроены под небольшой экран мобильного, часть контента вырезано, а это уже не очень хорошо. Самое главное, что в мобильной версии решена проблема с поиском и нажатием на кнопку каталог (за ее отсутствием), каталог сделан удобно и из любой страницы сайта можно легко им пользоваться. Еще один минус – грузится долго, как и основная версия. На рисунке 6 можно увидеть веб-сайт магазина 3, с экрана мобильного.

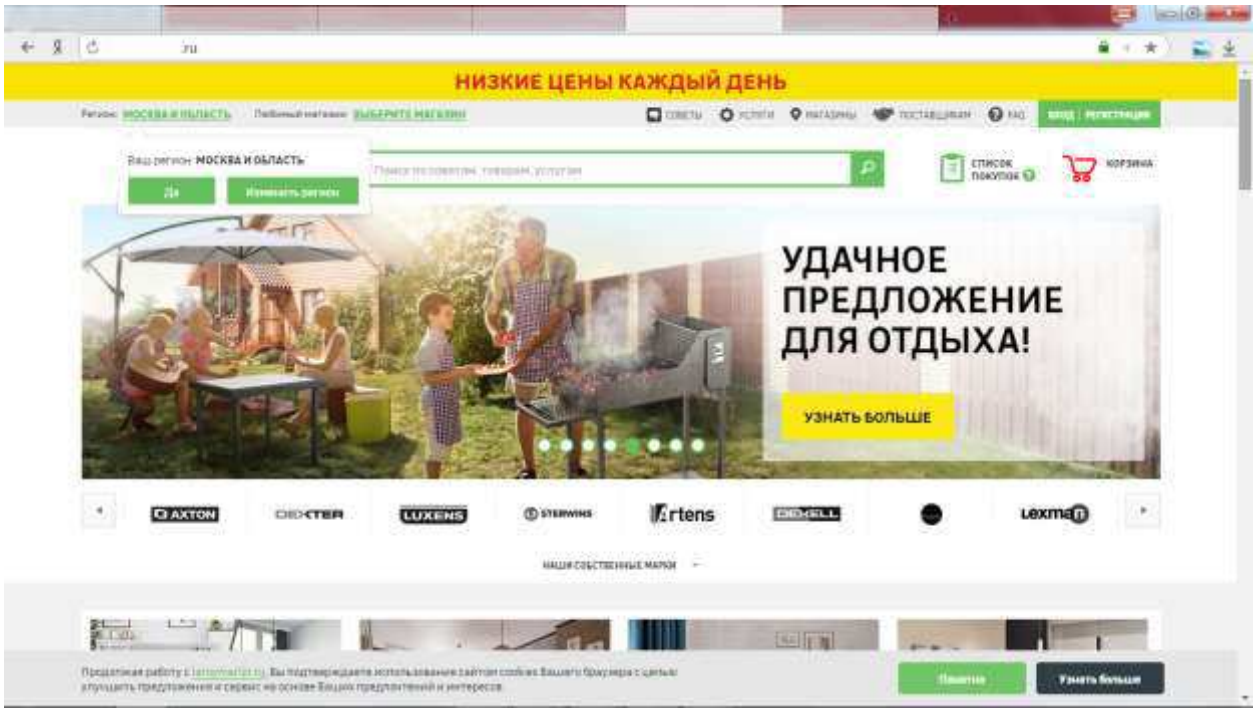

Рисунок 5 – Индексная страница веб-сайта магазина 3

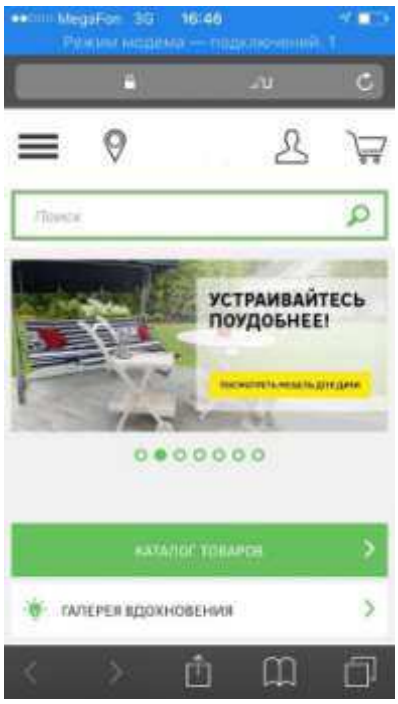

Рисунок 6 – Индексная страница веб-сайта магазина 3 с экрана смартфона

1.3 Специфика работы магазина строительных материалов «ДомоХозяин» и составление требований к веб-сайту

Магазин строительных материалов «ДомоХозяин» работает с 2017 года, ассортимент магазина уже насчитывает несколько тысяч наименований. Магазин находится в отдаленном от центра районе города, поэтому его администрация вынуждена завышать цены, так как нет достаточного количества клиентов и получается маленький товарооборот. Данную проблему можно решить с помощью веб-сайта. Главная задача веб-сайта – привлечение новых клиентов, что позволит снижать цены на товар и привлечь еще больше клиентов. Веб-сайт должен совместить в себе черты сайта-презентации и сайта-каталога, предоставляя полную информацию о деятельности магазина для потенциальных покупателей и определенную полезную информацию для посетителей – в виде текстового контента и видеоматериала. Сайт должен содержать каталог и возможность скачать прайс лист. Дизайн должен быть интересным и привлекательным. Должен быть удобный интерфейс, навигация и поисковая система по сайту. Сайт должен быть адаптивен и привлекателен для

любых устройств, начиная от мобильных телефонов и заканчивая, большими экранами мониторов. Потому что с каждым днем, все больше людей пользуются интернетом с помощью мобильных устройств и сайты, не адаптированные специально под их устройства, их не привлекают, и они уходят к конкурентам.

2 Разработка веб-сайтов

2.1 Этапы проектирования веб-сайтов

Прежде чем начать проектирование и разработку Web-сайта, нужно составить план, разбить работу на этапы:

1 Этап – определение целей создания сайта.

Именно от того, насколько точно будет поставлена цель, зависит эффективность работы на дальнейших этапах. Не нужно недооценивать момент постановки цели, ведь если она поставлена неправильно, то чем позже это станет понятно, тем больше работы придется переделывать.

2 Этап – проведение исследований по теме.

Необходимо разобраться в теме, на которую будет разрабатываться веб-сайт. Просмотреть и проанализировать веб-сайты потенциальных конкурентов. Возможно получится выявить какие-то ошибки, что-то не удобное для посетителей, и не допустить их в создании своего веб-сайта. Либо наоборот, какие-то моменты получится использовать в своей концепции. Хорошо, если получится заранее подготовить контент для будущего сайта.

3 Этап – определение типа сайта, разработка ТЗ и структуры.

На этом этапе нужно составить ТЗ (техническое задание) на создание веб-сайта. Техническое задание включает: выбор типа сайта и его функционала, стиля дизайна, описание структуры сайта и структуры каждой страницы.

4 Этап – разработка макета дизайна веб-сайта.

Можно сказать, что это презентация будущего сайта, его визуализация (демонстрация будущего сайта в виде графического элемента, без активных ссылок, кнопок и других динамических модулей). Макет нужен для того, чтобы доработать расположение и размеры элементов сайта, подправить структуру, исходя из соображений удобства поиска и комфортного использования информации пользователями. За основу для разработки макета берутся: логотип, слоган, корпоративные цвета, какие-то графические элементы дизайна. Важно прорисовать в макете все блоки, которые будут на сайте. Если

графические материалы еще не готовы, то нужно использовать любые другие материалы, но не оставлять «пустые места». При составлении макета веб-сайта нужно придерживаться технического задания.

5 Этап – html-css вёрстка.

Html – основа любого сайта. От того, насколько грамотно будет написан код сайта, будет зависеть скорость обработки этого кода браузером, а в дальнейшем и скорость отображения сайта на экранах посетителей. Код сайта должен быть написан в соответствии со стандартом, все элементы должны быть использованы логично и правильно. Чем короче будет код, тем лучше. Должна быть возможность корректировки, чтобы легко добавлять или менять информацию на сайте. Важно, чтобы страницы одинаково отображались в разных браузерах как в новых, так и в старых.

Если html отвечает за структуру любого веб-сайта, то css – за его внешний вид. Раньше «верстальщику» приходилось с помощью тегов и их атрибутов создавать внешний вид сайта прямо в html коде. Теперь все теги, отвечающие за внешний вид, устарели и мало где используются, потому что css дает больше возможностей и проще в использовании.

В веб-дизайне нет жестких правил. Главное – привлечь на свой сайт как можно большее число посетителей, а для этого необходимо понимание потребностей аудитории и четкое представлениео том, как сайт будет использоваться.

6 Этап – программирование и настройка функционала сайта.

На предыдущем этапе мы получили сверстанный веб-сайт, но работоспособен он только на половину. В дальнейшем его придется программировать для подключения к страницам различного функционала. Необходимо разработать удобную систему управления контентом и администрирования, что бы можно было легко управлять будущим сайтом. Возможно, появится необходимость в разработке дополнительных функций на сайте: возможность регистрации, профили пользователей, обратная связь, форум и т. д. Для получения такого движка сайта, может быть использован

один или несколько языков программирования, например: Java, PHP или Phyton.

В итоге производится ключевой тест работоспособности скриптов, проверка на наличие ошибок и производительность функционала сайта в целом.

7 Этап – заполнение сайта контентом (информацией).

Качественно и интересно подобранный контент (текст и графика), это то звено, которое заставит посетителя задержаться на нашем веб-сайте. Информация должна быть подобрана в соответствии с общей тематикой сайта.

8 Этап – размещение? сайта в интернете.

Размещаем веб-сайт на выбранном доменном имени, регистрируем в крупных поисковиках и каталогах. Далее подбираем ключевые слова, которые более точно характеризуют тематику сайта и вид деятельности организации.

9 Этап – продвижение сайта и реклама в интернете.

Когда сайт полностью готов к работе, нобходимо привлекать на него посетителей. Для «раскрутки» сайта можно воспользоваться контекстной или баннерной рекламой, SEO, SMO и другими методами.

2.2 Верстка HTML и CSS

Верстка – это термин, обозначающий процесс подготовки контента для публикации в интернете, или, точнее, разметки контента с помощью HTML-тегов, описывающих содержимое и его функции.

Язык разметки гипертекстовых страниц (HTML – Hypertext Markup Language) – язык, используемый для создания веб-страниц. Это не язык программирования, а язык разметки, он определяет синтаксис и размещение специальных инструкций (тегов), которые не выводятся на экран, но указывают браузеру, как отображать содержимое документа. Он также используется для создания ссылок на другие документы, локальные или сетевые, например, находящиеся в сети Интернет.

Документы HTML являются обычными текстовыми файлами. Это

означает, что для их создания можно использовать любой текстовый редактор, даже с минимальными возможностями.

Так же существуют специальные редакторы, которые делают работу с html-кодом более удобной. Например, в Notepad++ существует возможность работы сразу с несколькими документами, подсветка синтаксиса html и других языков разметки и программирования, программа предлагает варианты авто-завершения набираемого слова и т. д.

Документ HTML содержит текст (видимое содержимое страницы, контент) и встроенные теги. Тег – это конструкция языка, которая указывает браузеру на то, что он должен сделать. Любой документ HTML начинается с тега <HTML> и заканчивается тегом </HTML>. Это и дает браузеру возможность понять, что это и есть HTML-документ. Документ HTML разделяется на две основные части: заголовок – head(голова) и содержимое – body(тело). Заголовок содержит специальные теги (служебную информацию), которые не отображаются на веб-странице, но, по сути, организуют ее работу (например, они указывают браузеру на название страницы или на то, в какой кодировке она написана, а также на подключение к странице дополнительных файлов, таких как таблицы стилей или скриптов) [7]. В теле находятся «форматирующие» теги, которые отвечают за видимую часть страницы, то, что уже будет видимо для пользователя. На рисунке 7 приведены все самые основные теги html-документа и приведен пример простейшей веб-страницы.

Рисунок 7 – Основные теги простейшей веб-страницы

Текст, записанный в html документе, редактируется только с помощью тегов. Разделение на абзацы и прочее редактирование самого текста ни как не влияют на отображение его браузером.

Каждый тег состоит из имени, которое обычно отражает его действие, за которым может следовать список необязательных атрибутов, все они находятся внутри угловых скобок <>. Любая информация, записанная в угловые скобки, не выводится браузером на экран, так как считывается им как тег. Атрибут – это свойство тега, которое содержит определенное значение и влияет на функцию тега. Есть «общие» атрибуты, которые можно назначить любым тегам, а есть специальные атрибуты, которые могут принадлежать только одному или нескольким типам тегов. Обычно в html-документе имя тега и его атрибуты лояльно относятся к изменению регистра. Так, например, <BODY BGCOLOR=white> будет равнозначно понято браузером, как и <body bgcolor=white>. Но значения атрибутов чувствительны к регистру, поэтому их следует вводить только прописными буквами английского алфавита.

Большинство тегов являются парными. Это означает, что у них имеется начальный (открывающий или стартовый) и конечный (закрывающий) теги. Текст, вложенный между открывающим и закрывающим тегами, будет

выполнять действие, соответствующее функции данного тега. Например, тег <I> позволяет сделать текст курсивным:

«Этот сайт нужен для хранения <I>важной информации</I>.»

Результат: Этот сайт нужен для хранения *важной информации*.

Закрывающий тег имеет то же имя, что и открывающий, но перед ним всегда нужно ставить слеш (/). По-простому можно назвать его «выключателем тега». В закрывающем теге никогда не вставляют атрибуты.

Иногда можно упустить закрывающий тег, а браузер определит конец тега из контекста. Допустим тег (абзац), можно записать без конечного тега и большинство браузеров все равно правильно отобразит текст и абзац. Но главное помнить, что это разрешено не всем тегам, и не все браузеры поддерживают такие упущения. Поэтому, лучше для всех парных тегов, не забывать писать конечный тег. Это особенно важно, когда в документе присутствуют каскадные таблицы стилей.

Некоторые теги не нуждаются в конечных тегах в принципе, так как они используются для вставки отдельных (автономных) элементов на страницу. Например, тег изображения <img> – помещает изображение на веб-страницу. Другие автономные теги – это разрыв строки (
), горизонтальная линия (<hr>) и специальные теги, блока head, такие как <meta> и <base>.

Атрибут – это свойство тега, которое нужно для расширения или изменения его действий. К одному тегу можно добавить несколько атрибутов. Атрибуты тега следуют после его имени и разделяются пробелами. Нет какой-то определенной последовательности или порядка следования атрибутов. Большинство атрибутов имеют значения, которые прописываются после знака равенства (=), находящегося после имени атрибута. Значения ограничены 1024 символами. Значения атрибутов обычно чувствительны к регистру. Иногда значения должны находиться в кавычках (двойных или одинарных). Правила записи значений, следующие:

– если значение атрибута состоит из одного слова или числа и содержит только символы английского алфавита (a-z), цифры (0-9) и/или специальные

символы (точка <.> или дефис<->), то можно не использовать кавычки после знака равенства;

– если значение атрибута состоит из нескольких слов, разделенных запятыми или пробелами, или содержит специальные символы (кроме точки или дефиса), значит нужно использовать при написании такого значения кавычки после (=). Например, URL нуждаются в кавычках, потому что они содержат символы "://". Естественно кавычки нужны для задания значений цветов с использованием формата "#rrggbb" [8].

Во избежание ошибок стоит использовать кавычки для всех значений атрибутов. Для осуществления воздействия нескольких тегов на один элемент в тег HTML может быть помещен другой HTML-тег. Такую комбинацию называют вложением и для правильной ее работы открывающий и закрывающий теги вложенного тега должны обязательно находиться между открывающий и закрывающий тегами внешнего тега, например:

«Это сайт нужен для хранения <B><I>важной информации</I></B>.»

Результат: «Этот сайт нужен для хранения ***важной информации***.»

Распространенная ошибка – перекрытие тегов, когда сначала закрывают внешний тег, а уже потом – вложенный. Конечно, некоторые браузеры все равно правильно отобразят содержимое, написанное таким образом, но большинство запретят нарушать правила, а это значит, что размешать теги правильно – очень важно. На данном примере можно увидеть неправильное вложение тегов (тег<B> закрывается перед закрытием <I>):

«Это сайт нужен для хранения <B><I>важной информации</B></I>.»

Такой код браузер не отобразит.

Виды игнорируемых тегов:

– разрывы строк. Переносы абзаца в документе html не будут восприниматься браузером, если не будет введен тег <p> или
. Исключением является записанный тег <pre>: весь текст, записанный между открывающим <pre> и закрывающим </pre>, автоматически расставляется так же как записано в html коде, то есть расставляются все переносы абзацев и

пробелы;

– символы табуляции и множественные пробелы. Если в документе html записать несколько пробелов или другие символы табуляции, браузер отобразит только один пробел. Есть возможность добавить на страницу дополнительные пробелы с помощью специального симола (Snbsp – символ неразрывного пробела). Тег <pre> позволит отобразить все пробелы так же как они записаны в html-файле;

– множественные <p>-теги. При написании нескольких тегов <p> подряд, без текста внутри, не будет отображено браузером в виде нескольких переносов абзаца, а отразится как один абзац. Для нескольких переносов строки можно использовать несколько тегов
, такой перенос правильно воспринимается браузером;

– нераспознаваемые теги. Если тег указан некорректно или какой-то определенный браузер не понимает введенный тег, то он будет его просто игнорировать. В различных браузерах такая ситуация приводит к разным результатам, браузер может вообще не вывести результат, а может отразить содержимое неопознанного тега как просто текст;

– текст в комментариях. Специальные элементы (<!-- открывающий) и (--> закрывающий) используются для выделения комментариев, браузер не показывает текст, введенный между ними. Между этими символами и текстом всегда нужно ставить пробел, иначе браузер может их «не заметить». В комментарий можно вносить различные символы, текст и цифры. Главное, что комментарий нельзя вкладывать внутрь тегов, других ограничений нет. Комментарии в основном используются для пометок верстальщиком, чтобы удобнее было разбираться в коде [9].

Доступность веб-страницы. При проектировании сайта с фиксированным размером страницы, скорее всего, придется задавать ей значения размера экрана. Нужно, что бы веб-страница корректно выводила данные для как можно большего числа пользователей. Необходимо выбрать наиболее популярное разрешение экрана и разрабатывать сайт таким образом, чтобы страница всегда

заполняла все рабочее пространство.

Есть мнение, что страница в формате 640x480 удобна, так как посетителю не нужно использовать горизонтальную прокрутку. Она может затруднить восприятие страницы и отпугнуть пользователей от веб-сайта. Противоположное мнение, которого придерживаются многие разработчики, что стандартной можно считать страницу в формате 800x600. Немногие используют для веб-страниц еще более высокие разрешения. Естественно, окончательное решение должно, в первую очередь, зависеть от аудитории. Допустим, если веб-сайт разрабатывается для дизайнеров графики, то можно предположить, что они имеют мониторы с разрешением 800x600 и выше, и, отталкиваясь от этого, уже и разрабатывается страница [10].

Сегодня все больше пользователей применяют мобильные устройства и планшеты для выхода в интернет, поэтому на первый план выходит разработка веб-сайтов, которые будут удобны именно для таких пользователей, либо дополнительно разрабатывается мобильная версия.

Основная проблема в том, что раньше веб-сайты проектировались с ориентацией на разрешение мониторов персональных компьютеров, но ситуация сложилась таким образом, что стандартная верстка для мобильных устройств не годится, так как размеры экранов мобильных устройств значительно отличаются от диагоналей экранов компьютеров, из-за чего часто качественный и красивый дизайн веб-сайтов на экранах смартфонов и планшетов отражается некорректно. Это создает определенные проблемы, так как такие сайты долго прогружаются, текст нечитабельный, неудобная навигация, не работают flash и java скрипты. Большая часть пользователей долго не задерживаются на таких сайтах, и, столкнувшись с неудобствами, воспользуются услугами конкурентов, которые более продуманно отнеслись к мобильной версии сайта.

Есть еще одна важная причина, по которой стоит задуматься о создании мобильной версии веб-сайта – это появление мобильного поиска. Сайт просто не попадет в этот поиск, если у него не будет мобильной версии, так как в

десктопном поиске ситуация вполне стабильная, поисковики все большее внимание уделяют мобильному поиску, так как этот рынок растет в геометрической прогрессии и занимает все большие ниши [11].

Выделяют 3 варианта создания «мобильной версии сайта»:

1 Мобильная версия, как отдельный сайт для мобильных устройств на поддомене.

2 Адаптивный дизайн.

3 Мобильные приложения для IOS, Android, Windows Phone.

Мобильные приложения разрабатываются для мобильных операционных систем. При входе на сайт с мобильного устройства, посетителю предлагают скачать приложение. Положительные стороны подобного решения проблемы: удобный интерфейс, и разнообразные «фишки» которые не «впихнуть» в веб-сайт, личный кабинет, и т. п. Отрицательное заключается в том, что пользователя нужно как-то убедить скачать и установить приложение, а это уже отдельный пункт маркетинговой стратегии. Такие приложения нужно обслуживать и обновлять, защищать от взлома и т. д.

Адаптивный дизайн – это технология, позволяющая за счет определенных стилей кода автоматически адаптировать веб-сайт по ширине под разрешение устройства, с которого он открыт, неважно, что это: экран ПК, телефон или планшет. При разработке сайта верстальщики пишут его не под определенный масштаб, а проектируют страницы сайта так, что бы элементы автоматически подстраивались под разрешение экрана. Блоки либо меняют свое расположение, либо вообще не отображаются на мобильных устройствах. Такой вариант скорее подходит для обычных сайтов-визиток или блогов [12].

Плюсы адаптивного дизайна:

- правильный вывод всех элементов веб-сайта на различных устройствах за счет адаптации стилей к мобильным браузерам;

- соответствие требованиям поисковиков по удобству просмотра на мобильных устройствах;

- удобство в разработке – проще, чем создание мобильной версии;

- наличие одного адреса у стационарной версии сайта с адаптивной версткой, в связи с чем не нужно перенаправление посетителей, а также отсутствие необходимости для пользователей сайта запоминать дополнительный адрес поддомена;

- сохранение изначальной красоты веб-сайта.

Минусы адаптивного дизайна:

- страницы веб-сайта имеют большой вес из-за того, что адаптивный дизайн не позволяет заменить тяжелые декстопные элементы облегченными, поэтому сайт медленно загружается на мобильных устройствах. Это критично, поэтому скорость загрузки необходимо увеличивать;

- столкнувшись с какими-либо проблемами на адаптивной верстке, пользователь не сможет перейти на стандартную версию сайта;

- адаптивный дизайн предполагает перестройку каждой страницы веб-сайта.

Мобильная версия – специальная, отдельно разрабатываемая версия сайта, которая находится на поддомене. Поддомен сайта – это отдельный сайт, доменное имя которого включает доменное имя основного сайта и какую-то дополнительную приставку. То есть, доменное имя основного сайта будет являться доменной зоной его поддомена. Допустим, сайт с доменным именем m.domohozyain.ru будет поддоменом сайта domohozyain.ru. Обычно, мобильная версия создается под мобильные экраны с шириной менее 620px. Суть технологии в том, чтобы реализовать микро-версию сайта, содержащую только самый основной контент, не перегруженную информацией и разнообразными графическими элементами, а главное слишком крупными элементами навигации. Подойдет как для простых сайтов, так и для интернет-магазинов, различных сервисов: почта, новости, социальная сеть [13].

Плюсы мобильной версии:

- удобна для посетителей, так как значительно упрощена по сравнению со стандартной версией. Мобильная версия оснащена наиболее важной информацией и дает возможность совершить заказ/покупку с минимумом

действий;

- достаточно просто кардинально изменять код, так как это отдельная версия и не нужно вносить изменения в основной сайт;

- быстрая загрузка страниц, так как сайт сильно урезан и большенство элементов имеют меньший вес;

- есть возможность перейти на стандартную версию сайта при возникновении проблем на мобильной версии;

- соответствует требованиям поисковиков по удобству просмотра на мобильных устройствах;

Недостатки мобильной версии:

- сложность проекта, разработка мобильной версии сравнима с созданием приложения;

- сложность обслуживания: необходимо поддерживать работу версий для разных устройств;

- от части контента, графики и возможностей веб-сайта придется отказаться.

Основы CSS.

HTML дает возможность задавать параметры текста (цвет, размер и т. д.) при помощи тегов форматирования. При изменении одного конкретного тега это конечно удобно, но, если необходимо поменять характеристики множества однотипных элементов на веб-сайте, придется просматривать все страницы, чтобы найти и поменять теги, что займет много времени.

Каскадные таблицы стилей (Cascading Style Sheets, CSS) позволяют хранить цвет, размер и другие параметры текста в стилях. Стилем называется набор правил форматирования, который применяется к элементу страницы, чтобы быстро изменить его внешний вид [14].

Стили дают возможность одним действием применить сразу всю группу атрибутов форматирования. С их помощью можно, например, изменить вид всех абзацев. Вместо форматирования каждого абзаца в три приема, когда сначала прописываются его размер, затем шрифт и, наконец, выравнивание по

центру, то же самое можно сделать одновременно, применив стиль к тегу <p>.

Второе преимущество CSS состоит в том, что стили предлагают гораздо больше возможностей для форматирования, чем обычный HTML-код. Они могут храниться во внешнем файле, браузер кэширует такие документы, от чего загрузка веб-сайта будет происходить чуть быстрее.

CSS – это мощная система для разработчиков веб-сайтов, расширяющая возможности по дизайну и верстке веб-страниц [15]. Изначально, когда технология WWW только начинала свой путь, программисты и верстальщики были заняты в основном содержанием документов, а не их оформлением, но для простых посетителей представление сайта – то, как он выглядит, играет более важную роль. Ограничения HTML породили множество техник создания веб-страниц, таких как:

- использование различных расширений HTML;
- применение графики вместо текста;
- использование изображений для контроля пустого пространства, так называемые распорки;
- использование таблиц для верстки веб-страниц, в качестве блоков [16];
- написание программных скриптов вместо использования HTML.

Эти техники значительно усложняют разработку веб-страниц, ограничивают гибкость в их создании и обслуживании и создают трудности для людей ими не владеющих. Стили решают эти проблемы, в то же время заменяя лишь ограниченную область механизмов представления HTML.

С помощью CSS определяются стиль и вид текста. Аналогично тому, что использует тег <font>, задающий свойства шрифта, стили обладают большими возможностями и позволяют сократить код HTML.

2.3 Соверменные способы адаптивной верстки веб-сайтов

Современные тенденции в веб-разработке толкают разработчиков на активное применение на практике различных инструментов, позволяющих облегчить процесс верстки макетов и программирования клиентской

составляющей проекта. Для этих целей существует множество фреймворков, систем сборки, пакетных менеджеров, пакетов для задач любого уровня, препроцессоров, шаблонизаторов и прочих «помощников», которые придуманы что бы упростить и увеличить продуктивность работы эксперта в данной области.

Но для начинающих разработчиков слишком трудно освоить весь спектр этих инструментов, ведь для этого нужно стабильно пополнять свои знания и иметь представление об основах верстки. По этой причине некоторые начинающие веб-разработчики начинают верстать «по-отцовски», используя обычные html и css, с множеством повторяющихся селекторов, верстка без вникания в принципы различных сеток, но создающая множество «костылей» с различного рода позицианированием и т. д.

В новых версиях Html5 и Css3 можно найти ряд возможностей, которые помогут решить проблемы с позиционированием и адаптивным дизайном и упростят верстку веб-страниц.

1 @media-запросы

2 FlexBox

3 Css grid

Медиа-запросы применяются, тогда, когда нужно добавить различные css-стили для разных по типу отображения устройств (например, для принтера, монитора или смартфона), а также конкретных характеристик устройства (ширины окна просмотра браузера), или внешней среды (например, внешнее освещение). Учитывая немалое число подключаемых к интернету носителей, медиа-запросы становятся очень нужным инструментом для создания веб-страниц и приложений, которые будут правильно работать на всех доступных носителях.

Вместе с типами устройств в CSS3 существует поддержка разнообразных технических параметров носителей, на основании которых требуется подключать те или иные стили. Например, можно установить смартфон с максимальным разрешением 640 пикселей и для него ввести определенные

свойства Css-стилей, а для других устройств – другие. Еще можно обнаружить разные свойства, например, наличия монохромного экрана, ориентации (портретная или альбомная) и другие. Все свойства просто комбинируются, поэтому можно задать стиль только для устройств в альбомной ориентации с определенным разрешением экрана.

Возможности медиа-запросов не ограничиваются распознанием мобильных устройств, с их помощью вполне возможно создавать адаптивный макет, который будет подстраиваться под разрешение монитора и окна браузера, изменяя при необходимости ширину макета, число колонок, размеры изображений и текста. Медиа-запросы ограничивают ширину макета и при достижении определенного значения (к примеру, за счёт уменьшения окна браузера или при просмотре на устройстве с указанным размером) будет применяться иной стиль.

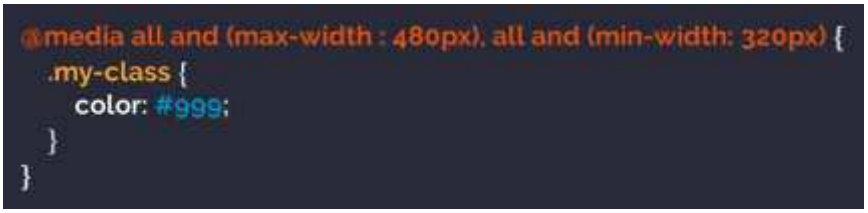

Рисунок 8 – Пример медиа запроса

На рисунке 8 приведен пример простейшего медиа запроса, который должен показать браузеру, что при разрешении экрана меньше 480 пикселей и больше 320 пикселей, нужно изменить цвет элементов под классом «.my-class» на цвет который обозначается как «#999».

Не менее важным, чем медиа запросы, является модуль Flexbox Layout (Flexible Box) направленный на предоставлении более эффективных способов позицтонирования: расположения, выравнивания и разделения свободного пространства между элементами в контейнере, даже если размер этих элементов изначально неизвестен и/или динамичен (отсюда и название «flex» гибкий).

Главная функция гибкой (flex) разметки в том, чтобы дать контейнеру

возможность поменять ширину/высоту (или последовательность) своих элементов, и оптимальным образом заполнить свободное пространство (в основном для корректного отображения на всех типах и размерах экранов). Flexbox контейнер расширяет элементы, чтобы заполнить доступное пространство или сжимает их, чтобы избежать переполнения, выхода контента за границы веб-страниц и исключить горизонтальную полосу прокрутки.

Особенно важно, что Flexbox «не зациклен» на определенном направлении в отличии от обычных блоков, сформированных на вертикальном позиционировании или строчных элементов, в основе которых лежит горизонтальное позиционирование. С их помощью можно работать, но им не хватает гибкости для использования в больших или сложных веб-документах, особенно если требуется изменение размеров или ориентации, растягивание или сжатие.

Flexbox предназначен для компонентов приложения и маленьких макетов, в отличии от CSS Grid подходящего для более масштабных верстоk.

Flexbox – это целый модуль, а не отдельное свойство, он включает целый комплект свойств. Часть из них предназначена для контейнера (родительский элемент, называемый «flex-контейнер»), а остальные устанавливаются дочерним элементам (называемым «flex-элементами»).

Если раньше система компоновки была основана на блочных и строчных направлениях, то Flexbox добавил так называемые «flex-flow направления». На рисунке 9, показана основная спецификация flex-box позиционирования. Изначально элементы будут располагаться вдоль основной оси (от main-start к main-end) или перпендикулярной оси (от cross-start к cross-end).

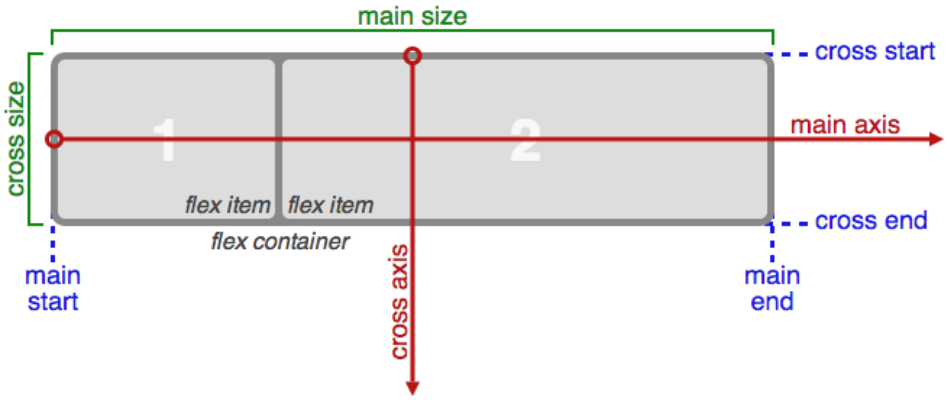

Рисунок 9 – Структура FlexBox

- main axis – это основная ось flex-контейнера, на которой расположены flex-элементы. Это не обязательно горизонтальная ось, направление зависит от свойства flex-direction;
- main-start | main-end – flex-элементы размещаются внутри контейнера, начиная с main-start и заканчивая main-end;
- main size – ширина или высота flex-элемента, в зависимости направления основной оси;
- cross axis – «поперечная» ось, она перпендикулярна основной оси;
- cross-start | cross-end – flex-элементы размещаются внутри контейнера, начиная с cross-start и заканчивая cross-end;
- cross size – ширина или высота flex-элемента, в зависимости направления поперечной оси;

Свойства для родительского элемента (Flex-контейнера):

1 display – Назначает flex-контейнер; строковый или блочный зависит от прописанного значения. Запускает flex-контекст для всех своих прямых, дочерних элементов.

```
.kont {
display: flex; или display: inline-flex;
}
```

2 flex-direction – определяет основную ось и таким образом устанавливает направление элементов, расположенных в контейнере. Flexbox (помимо

опциональной обёртки) представляет собой концепцию однонаправленного макета. Flex-элементы, прежде всего, это горизонтальные строки или вертикальные колонки. На рисунке 10 представлены все варианты направлений, заданных значениями свойства flex-direction.

Рисунок 10 – Варианты направлений, заданных значениями flex-direction

.kont {

flex-direction: row; (по умолчанию – слева направо) | row-reverse; (справа налево) | column; (сверху вниз) | column-reverse; (снизу вверх)

}

3 flex-wrap – по умолчанию, элементы будут заполнять только одну строку, но можно поменять их поведение и разрешить элементам выноситься с переполненной строки на следующую.

.kont {

flex-wrap: nowrap; (в одну строку)| wrap; (в несколько строк, сверху вниз)| wrap-reverse (в несколько строк снизу вверх);

}

4 flex-flow – сокращение для свойств flex-direction и flex-wrap, которые вместе выражают основную и поперечную оси контейнера. По умолчанию row nowrap.

.kont {

flex-flow: сперва пишем значение <‘flex-direction’> || затем через пробел <‘flex-wrap’>;

}

5 justify-content – определяет выравнивание по основной оси. Это помогает разделить свободное пространство, оставшееся после того как все фиксированные и «свободные» по ширине flex-элементы, достигли максимального размера. Дополнительно justify-content способствует осуществлению контроля над выравниванием элементов, когда они переполняют строку. Обеспечивает горизонтальное выравнивание, если основная ось горизонтальная. На рисунке 11 представлены все варианты расстановки элементов, заданные значениями свойства justify-content.

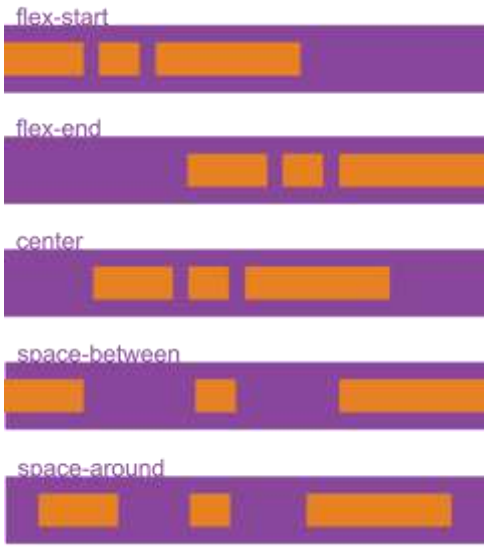

Рисунок 11 – Варианты расстановки элементов, заданные значениями свойства justify-content

justify-content: flex-start (по умолчанию) – элементы прижимаются к началу строки;

justify-content: flex-end – элементы прижимаются к концу строки;

justify-content: center – элементы группируются по центру строки;

justify-content: space-between – элементы размещаются равномерно по строке; крайние элементы прижимаются у началу и соответственно к концу строки;

justify-content: space-around – элементы размещаются равномерно по строке с одинаковыми отступами от друг друга и от начали и конца строки (по краям это расстояние в 2 раза меньше, за счет того что отступ считается от

каждого элемента, равный с обеих сторон).

6 align-items – это свойство формирует поведение flex-элементов вдоль поперечной оси на текущей строке. Действует подобно justify-content, обеспечивая вертикальное выравнивание если основная ось горизонтальная. На рисунке 12 представлены все варианты расстановки элементов, заданные значениями свойства align-items.

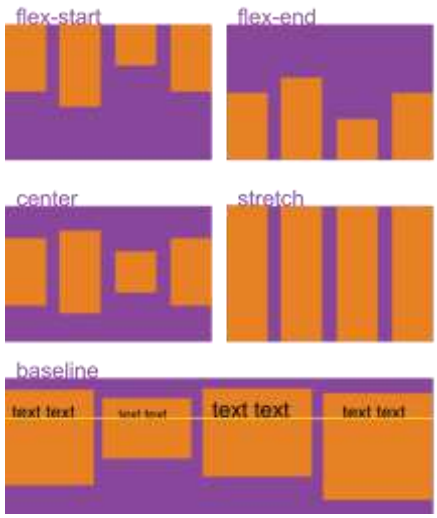

Рисунок 12 – Варианты выравнивания элементов, заданные значениями свойства align-items

.kont {

align-items: flex-start; (в начале поперечной оси) | flex-end; (в конце поперечной оси) | center; (по центру поперечной оси) | baseline; (выравнивание по базовой линии) | stretch; (растягивание и заполнение контейнер целиком соблюдая min/max-width | baseline; (выравнивание по базовой линии)

}

7 Flexbox Align Content – свойство действует, когда есть несколько строка с flex-элементами, определенно выравнивая строки относительно flex-контейнера. Варианты выравнивания представлены на рисунке 13.

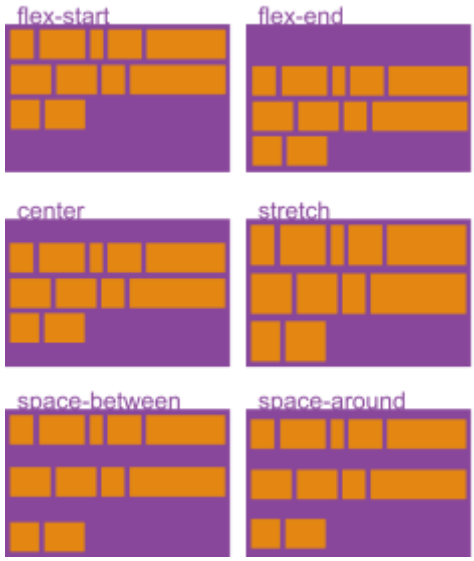

Рисунок 13 – Варианты выравнивания строк относительно flex-контейнера, заданные значениями свойства align-content

. kont{

align-content: flex-start | flex-end | center | space-between | space-around | stretch;

}

Свойства для дочерних элементов (Flex элементов):

1 Order. Изначально, все элементы помещаются в том же порядке, как они записаны в html документе. С помощью свойства order можно управлять порядком, в котором будут располагаться элементы внутри контейнера. Чем меньше значение, тем ближе к началу будет элемент.

. element {

order: -1; или order: 5;

}

2 flex-grow – свойство задающее возможность элемента увеличиваться в размере, при необходимости. Оно получает безразмерное значение в качестве пропорции, определяющее сколько свободного пространства внутри контейнера должен занять элемент. В том случае, когда у всех элементов значением свойства flex-grow является 1, незанятое пространство внутри контейнера будет равномерно распределено между всеми элементами. Если у

элемента значение больше чем у остальных, то он будет пытаться занять больше пространства, на разницу между значениями свойства flex-grow. На рисунке 14 приведен пример результата применения свойства flex-grow.

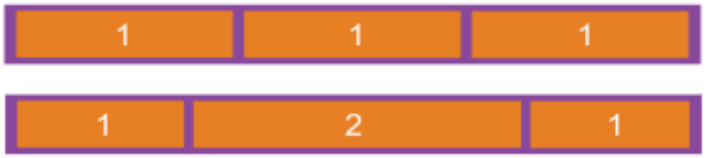

Рисунок 14 – Пример применения свойства flex-grow

```
.element {
  flex-grow: <number>; /* по умолчанию 0 */
}
```

В качестве <number> можно применять только натуральные положительные целые числа.

3 flex-shrink – свойство определяющее возможность элемента уменьшать свой размер, при необходимости.

```
.element {
  flex-shrink: <number>; /* по умолчанию 1 */
}
```

Значение <number> такие же как и у свойства flex-grow.

4 flex-basis – назначает размер элемента по умолчанию, до распределения оставшегося пространства. В качестве значения может быть размер в процентах, rem, px и т.д. Так же в качестве значения можно использовать ключевое слово:

- auto - элемент принимает размер своих свойств width или height;
- Content – размер основан на содержимом элемента
- min-content
- max-content
- fit-content.

```
.element {
```

flex-basis: <length> | auto; /* по умолчанию auto */

}

При установленном значении 0, дополнительное пространство вокруг содержимого не будет учитываться. Если установить auto, дополнительное пространство будет распределяться на основе значения flex-grow.

5 flex – сокращение для flex-grow, flex-shrink и flex-basis. Второй и третий параметры (flex-shrink и flex-basis) не обязательны. Значение по умолчанию выставлены как: 0 1 auto.

.element {

flex: none | [<'flex-grow'> <'flex-shrink'>? || <'flex-basis'>]

}

6 align-self – свойство, позволяющее выровнять отдельный flex-элемент, отменяя выравнивание, заданное с помощью свойства align-items. На рисунке 15 приведен пример применения align-self: flex-end, когда у контейнера прописано: align-items: flex-start.

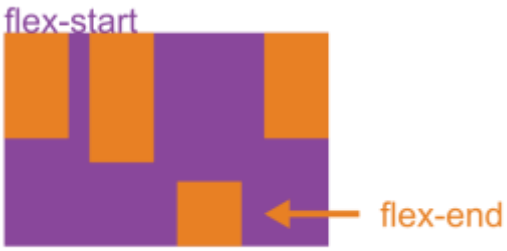

Рисунок 15 – Пример применения значения flex-end свойства align-self

.element {

align-self: auto | flex-start | flex-end | center | baseline | stretch;

}

На flex-элементы не действуют свойства float, clear и vertical-align. В основном свойства FlexBox направлены на помощь в позиционировании [17].

CSS Grid Layout («Grid») – это двумерная система компоновки, основанная на сетке, цель которой заключается в том чтобы полностью

изменить способ проектирования пользовательских интерфейсов основанных на сетке. CSS всегда использовался для разметки веб-страниц, но раньше были большие проблемы с позиционированием элементов. Раньше разработчики использовали таблицы, потом обтекания (floats), позиционирование и линейные блоки (inline-block), это было сложно, долго, не адаптивно и не давало много нужных функциональных возможностей, таких как вертикальное выравнивание и так далее. Flexbox хороший инструмент, но он нужен для простых одномерных макетов, а не для сложных двумерных. CSS Grid'ы – это первый модуль, созданный специально для решения проблем позиционирования, которые до этого момента решались при помощи не подходящих для этого инструментов при проектировании веб-сайтов. Нужно подчеркнуть, что Grid, это не замена Flexbox'у, а скорее наоборот «дополнение». Использование двух этих модулей вместе облегчает верстку в десятки раз.

Так же, как и во Flexbox, последовательность элементов в Html-документе, не важна, так как с помощью «гридов» можно расставлять их в любом порядке, а это еще и упрощает перегруппировку сетки при использовании медиа запросов. Можно распределить разметку всей страницы, а затем полностью перестроить её, для корректного отображения на устройстве с другой шириной экрана используя лишь несколько строчек CSS-кода. CSS Grid – один из самых мощных CSS модулей, представленных за все время существования html и css. На сегодняшний день процент поддержки CSS Grid составляет 88.11% с префиксами. Многие браузеры уже поддерживают CSS Grid, без префиксов: Chrome (включая Android), Firefox (включая Android), Safari (включая iOS), и Opera (включая Android). Internet Explorer 10 и 11 поддерживали его, но с использованием более старого синтаксиса [18].

Первым делом назначается элемент-контейнер с помощью display: grid, это прямой родитель для всех элементов сетки. Дочерние элементы (прямые потомки) контейнера.

Разделительные линии, составляют структуру для сетки. Они могут быть вертикальными («линии колонок») или горизонтальными («линии строк») и

располагаются с двух сторон от строки или столбца. На рисунке 16 приведен пример вертикальной линии или линии колонки (желтая).

Рисунок 16 – Линия колонки (CSS Grid Line)

Пространство между двумя соседними линиями называется трек сетки. Можно назвать их столбцами или строками сетки. На рисунке 17 изображён трек между второй и третьей линией строк.

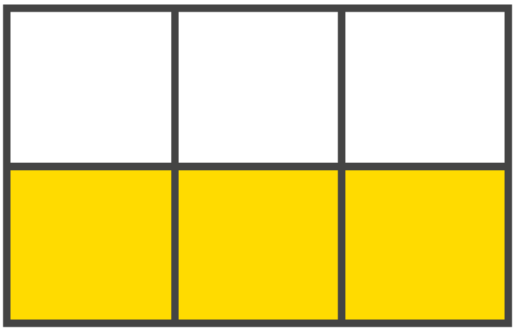

Рисунок 17 – Трек сетки (CSS Grid Track)

Ячейка сетки – пространство между линиями двух соседних строк и двух соседних столбцов. Ячейка является «единицой измерения» сетки. На рисунке 18 представлен пример ячейки между линиями строк 1 и 2, линиями колонок 2 и 3.

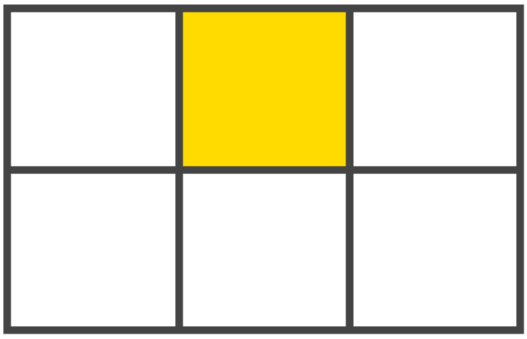

Рисунок 18 – Ячейка сетки (CSS Grid Cell)

Пространство, окружённое линиями с четырех сторон, называют областью сетки. Область может содержать любое количество ячеек. На рисунке 19 представлен пример области между строками 1 и 3, и колонками 1 и 3.

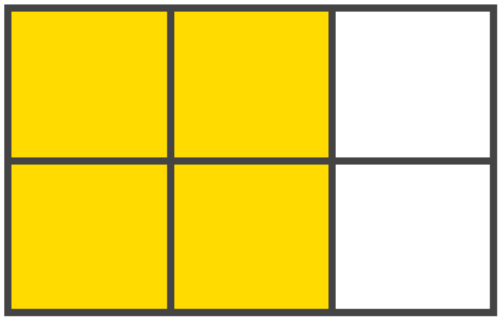

Рисунок 19 – Область сетки (CSS Grid Area)

Свойства для родительского элемента (Контейнера сетки)

1. display – назначет элемент контейнером и вводит для его содержимого новый контекст создания сетки.

Значения:

- grid – формирует сетку как блок;
- inline-grid – формирует сетку как линейный блок;
- subgrid – если ваш контейнер сам является элементом другого контейнера (при использовании вложенной сетки), то можно использовать это значение того, чтобы размеры строк/колонок были переданы от родительского элемента, а не задавали свой;

При использовании свойства display с вышеперечисленными значениями,

свойства column, float, clear и vertical-align не смогут воздействовать на данный контейнер.

2. grid-template-columns и grid-tempate-rows – свойства формирующие колонки (columns) и строки (rows) сетки используя перечисление значений, разделённых пробелами. Значения обозначают размер трека, а пробелы между ними представляют линии сетки.

Значение может выражаться фиксированным размером, процентами или частью свободного пространства в сетке, с помощью единицы fr (fraction), специальные еденицы измерения для «гридов». Так же можно задать в качестве значения grid-template-columns или grid-tempate-rows уникальное имя.

Например:

```
.kont{
  grid-template-columns: 25px 30px auto 30px 25px;
  grid-template-rows: 40% 1fr auto 10px;
}
```

Когда остаются пробелы между значениями треков, линиям сетки автоматически присваиваются номера как показано на рисунке 20.

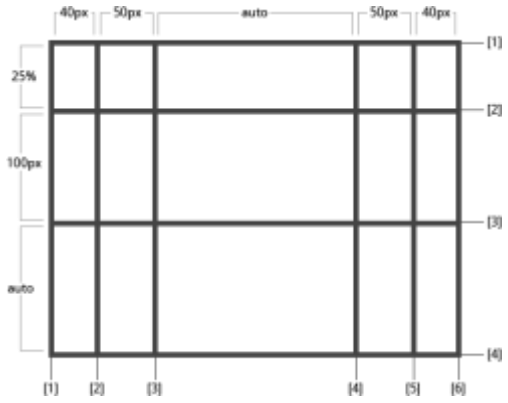

Рисунок 20 – Пример построение grid-сетки и присвоения номеров линиям сетки (CSS Grid Line Numbers)

Можно назвать линии самому, тогда между значениями треков нужно вводить уникальные имена в квадратных скобках. Для линии можно задать несколько названий через пробел.

Если значение содержит одинаковые элементы, то можно использовать

нотацию repeat(..):

```
.kont {
  grid-template-columns: repeat(3, 35px [line1]) 10%;
}
```

Тоже самое что и:

```
.kont {
  grid-template-columns: 35px [line1] 35px [line1] 35px [line1] 10%;
}
```

Единица измерения fr позволяет установить размер треков как часть свободного пространства в контейнере. На примере ниже, где каждому элементу выделяется одна третья часть ширины контейнера.

```
.kont {
  grid-template-columns: 1fr 1fr 1fr;
}
```

Свободное пространство вычисляется уже после того, как будет отведено место для всех фиксированных элементов.

3 grid-template-areas – свойство формирующее шаблон сетки ссылаясь на имена областей, которые заданы дочерним элементам контейнера используя свойство grid-area. Множественное написание имен областей приведет к тому, что область охватит большее число ячеек. Точкой обозначают пустую ячейку. Синтаксис представляет из себя визуализацию структуры сетки, как на примере ниже:

```
#element-a {
  grid-area: h1;
}
#element-b {
  grid-area: m1;
}
#element-c {
  grid-area: s1;
```

```
}
#element-d {
  grid-area: f1;
}
.kont{
  grid-template-columns: 50px 50px 50px 50px;
  grid-template-rows: auto;
  grid-template-areas:
    "h1 h1 h1 h1"
    "m1 m1 . s1"
    "f1 f1 f1 f1";
}
```

На рисунке 21 можно увидеть примерный результат данного когда: формируется сетка из 4 колонок и 3 строк. Вся верхняя строка будет занята областью h1 (header). В строке по середине будут помещены область m1 (main), занимающая 2 колонки, пустая ячейка и область s1 (sidebar), занимающая 1 колонку. Последняя строка будет содержать только область f1 (footer).

Рисунок 21 – Пример создания структуры веб-сайта с grid-template-areas

У каждой строки должно быть равное количество ячеек. Можно использовать разное количество не разделенных точек для обозначения пустых ячеек, они будут обозначать одну ячейку, пока между ними не появится пробел.

4 grid-template – сокращение для grid-template-rows, grid-template-columns,

и grid-template-areas.

Значения:

- none – устанавливает все три свойства в их начальное значение;
- subgrid – устанавливает grid-template-rows и grid-template-columns в subgrid, и grid-template-areas в его начальное значение;
- еще можно устанавливать определённые значения для всех трех свойств, получется более сложный, но довольно удобный синтаксис, для указания всех трёх свойств, как в примере:

```
.kont {
  grid-template:
    [row1-start] 25px "h1 h1 h1" [row1-end]
    [row2-start] "f1 f1 f1" 25px [row2-end]
    / auto 50px auto;
}
```

Если записать все это отдельно:

```
.container {
  grid-template-rows: [row1-start] 25px [row1-end row2-start] 25px [row2-end];
  grid-template-columns: auto 50px auto;
  grid-template-areas:
    "h1 h1 h1"
    "f1 f1 f1";
}
```

5 grid-column-gap и grid-row-gap – свойства позволяющие задать ширину линий. Визуально это похоже на ширину отступов между столбцами и строками. Ниже приведен пример кода с использованием свойств grid-column-gap и grid-row-gap, а на рисунке 22 приведен результат этого кода:

```
.kont {
  grid-column-gap: 10px;
  grid-row-gap: 15px;
```

}

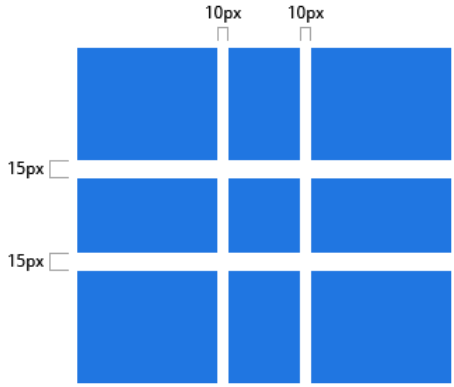

Рисунок 22 – Пример заданной ширины линий

Отступы создаются только между колонками и строками, но не для внешних краев сетки.

6 grid-gap – сокращенная запись для свойств grid-row-gap и grid-column-gap.

```
.kont{
  grid-gap: 10px 15px;
}
```

Если задать толко одно значение, то отступы будут равными.

7 justify-items – свойство, выравнивающее элементы вдоль оси строки, обеспечивая тем самым, горизонтальное выравнивание. Значение этого свойства применяется ко всем элементам сетки внутри контейнера. На рисунке 23, можно увидеть что произойдет, если ввести justify-items: start. Другие значения: end, center, stretch.

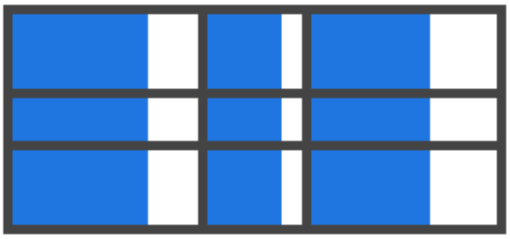

Рисунок 23 – Результат применения justify-items: start

8 align-items – свойство, выравнивающее элементы вдоль оси столбца, обеспечивая тем самым, вертикальное выравнивание. В остальном все характеристики и значения такие же как у свойства justify-items. На рисунке 24, можно увидеть, что будет если ввести align-items: end.

Рисунок 24 – Результат применения align-items: end

9 justify-content – применяется в тех случаях, когда общий размер сетки меньше размера контейнера. Это происходит если у всех элементов сетки даны фиксированные единицы измерения, например, пиксели. justify-content устанавливает выравнивание сетки внутри контейнера, вдоль оси строки обеспечивая, горизонтальное выравнивание. Кроме стандартных значений, таких как используют align-items и justify-items, у justify-content и align-content есть дополнительные значения:

- space-around – одинаковое пространство между элементами, и полуразмерные отступы по краям;
- space-between – одинаковое пространство между элементами, без отступов по краям;
- space-evenly – одинаковое пространство между элементами, и полноразмерные отступы по краям.

На рисунке 25 показан результат применения justify-content: end.

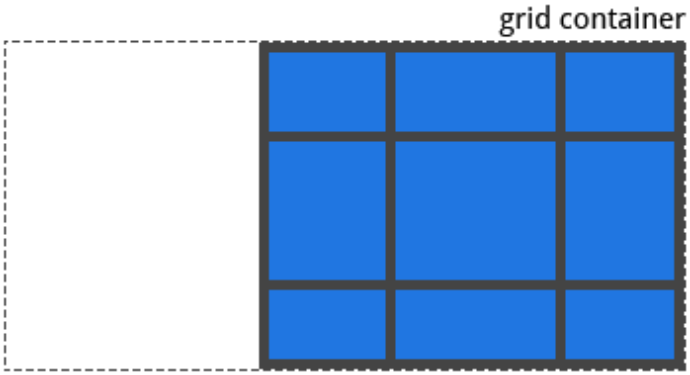

Рисунок 25 – Результат применения justify-content: end

10 align-content – свойство подобное по всем характеристикам justify-content, только выравнивает сетку вдоль оси колонки (вертикально). На рисунке 26 приведен результат применения align-content: start.

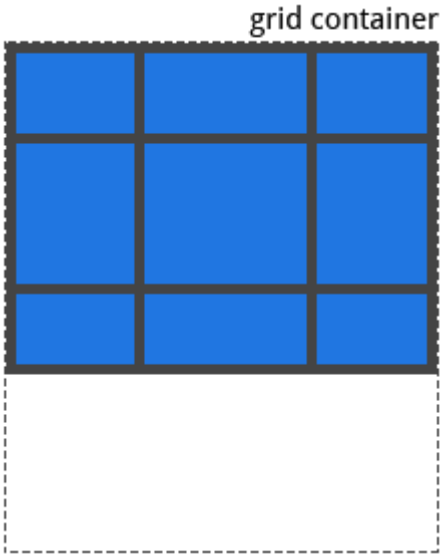

Рисунок 26 – Результат применения justify-content: end

11 grid-auto-columns и grid-auto-rows – свойства определяющее размер любых автоматически созданных треков (неявных треков). Неявные треки создаются при явном позиционировании столбцов и строк (через grid-template-rows/grid-template-columns), которые находятся за пределами заданной сетки.

12 grid-auto-flow – свойство, которое использующееся в тех случаях, когда есть элементы, которые явно не позиционируете в сетке, тогда запускается алгоритм авто-размещения, а grid-auto-flow контролирует то, как он

работает. Значения, которые использует grid-auto-flow:

row –алгоритм авто-размещения заполнит каждую строку поочереди, добавляя новые строки при необходимости;

column – алгоритм авто-размещения заполнит каждую колонку поочерёдно, добавляя новые колонки при необходимости;

dense – алгоритм авто-размещения попытаться заполнить свободное пространство в сетке, переставляя более мелкие элементы, расположенные дальше, если они подходят для этих мест. Это приведет к тому, что элементы будут отображаться не по порядку. Минус этого подхода в том, что при использовании клавиатуры для перемещения по веб-сайту, порядок будет нарушен, а это неудобно.

13 grid – свойство для сокрощения записи кода, содержит в себе такие свойства как: grid-template-rows, grid-template-columns, grid-template-areas, grid-auto-rows, grid-auto-columns, и grid-auto-flow. Еще он присваивает свойтвам grid-column-gap и grid-row-gap их начальные значения, хотя с его помощью и нельзя задать этим свойствам явные значения. Примеры, приведенные ниже, дадут один и тот же эффект:

```
.kont {
  grid: 200px auto / 1fr auto 1fr;
}
.kont {
  grid-template-rows: 200px auto;
  grid-template-columns: 1fr auto 1fr;
  grid-template-areas: none;
}
```

И следующие два примера также эквивалентны:

```
.kont {
  grid: column 1fr / auto;
}
.kont {
```

```
  grid-auto-flow: column;
  grid-auto-rows: 1fr;
  grid-auto-columns: auto;
}
```

Свойства для дочерних элементов (Grid элементы)

1 grid-column-start, grid-column-end, grid-row-start, grid-row-end

Определяют местоположение в сетке ссылаясь на конкретные линии. grid-column-start/grid-row-start - это линия с которой начинается элемент, а grid-column-end/grid-row-end - это линия на которой элемент заканчивается. В качестве значений может выступать название линии, число линий(span <number>), или достижение до какой-то конкретной линии(span <name>) или auto, означающее автоматическое размещения, автоматическое охват, или охват по умолчанию;

2 grid-column и grid-row – сокращенная для grid-column-start + grid-column-end, и grid-row-start + grid-row-end, соответственно.

3 grid-area – свойства нужное что бы задать название элементу, для использования этого названия в свойстве контейнера grid-template-areas. В качестве альтернативы, это свойство может быть использовано в качестве сокращения для grid-row-start + grid-column-start + grid-row-end + grid-column-end.

```
#elemetnt1 {
  grid-area: h1
}
```

```
#elemetnt2 {
   grid-area: 1 / col4-start / last-line / 6
}
```

4 justify-self – свойство равное по значению с justify-items, но для конкретного элемента, так же выравнивает содержимое элемента вдоль оси строки.

5 align-self – свойство равное по значению с align-items, но для конкретного элемента, так же выравнивает содержимое элемента вдоль оси столбца [19].

3 Разработка веб-сайта для магазина строительных материалов «ДомоХозяин»

3.1 Программное обеспечение, используемое при разработке веб-сайта

Для проектирования веб-сайта для магазина строительных материалов «ДомоХозяин», понадобятся следующие программы: Denwer – локальный сервер (Apache, PHP, MySQL), Adobe Photoshop CS4, Notepad++.

1. Денвер – это не город, а Де эН Вэ еР (Д.Н.В.Р.) аббревиатура, которая имеет следующий смысл – Джентельменский Набор Веб-разработчика [10]. Денвер это самоинсталирующийся дистрибутив, в состав которого входят сервер APACHE с PHP + MySql, miniPerl и заглушка SendMail которая позволяет тестировать отправку электронной почты [12].

2. Adobe Photoshop CS4 Portable – многофункциональный графический редактор, разработанный и распространяемый фирмой Adobe Systems. Основное назначение программы Adobe Photoshop создание фото, реалистических изображений, работа с цветными сканированными изображениями, ретуширование, цветокоррекция, коллажирование, трансформации, цветоделение и другое. Adobe Photoshop располагает всеми методами работы с точечными изображениями, при этом имеет возможность работы со слоями и использует контуры.

Программа позволяет легко изменять цветовое представление документов (битовое, в градациях серого, дуплекс, индексированные цвета, RGB или CMYK). Photoshop это программа растровой графики, то есть любой элемент изображения строится по точкам. Adobe Photoshop это профессиональный графический редактор, который при этом достаточно прост в освоении и использовании.

3. Notepad++ – свободный текстовый редактор с открытым исходным кодом для Windows с подсветкой синтаксиса большого количества языков разметки и программирования.

Программа предназначена для работы с текстом с большим количеством функций, которые помогают не запутаться в кодах и значительно упрощают их написание. В Notepad++ есть удобная функция замены спецсимволов на их коды. Часто такие символы не заметны в огромном количестве знаков на языках программирования, их можно заменить на коды, использовав только эту функцию и не перебирая весь текст вновь и вновь при помощи XML Notepad.

Основное отличие Notepad++ от других подобных программ в том, что в нем существует возможность открыть сразу несколько документов. Также для любого верстальщика или веб-мастера будут полезны функции: ликвидация лишних пробелов и перевод строки, нужные для того что бы сделать код чище и удобней для чтения. К программе можно подключить различные плагины, которые добавят еще несколько полезных функций этот текстовый редактор [13].

4. ColorMania 5.1 – программа для определения цвета. С помощью инструмента «пипетка» можно кликнуть по цвету и узнать его кодировку в различных представлениях: RGB, HEX, HSB, CMYK и других. Также можно выбирать цвет из различных цветовых палитр.

5. Комплект различных браузеров, таких как Opera, Firefox, Yandex, Сафари и IE5-IE8. Они нужны для проверки веб-сайта, чтобы определить, как он будет отображаться в разных браузерах.

6. Браузер Firefox Quantum: Developer Edition – единственный браузер с инструментами, созданными специально для построения и дизайна с помощью CSS-сетки. Эти инструменты позволяют визуализировать сетку, отображать связанные имена областей, предпросматривать трансформации на сетке и многое другое.

Новые инструменты разработчика Firefox Quantum – мощны, гибки и хорошо настраиваемы. В них есть лучший в своём классе отладчик JavaScript, который может эмулировать поведение других браузеров, и построен на React и Redux.

3.2 Описание веб-сайта для магазина строительных материалов «ДомоХозяин»

В процессе бакалаврской работы разработан веб-сайт, удовлетворяющий требованиям организации. Он совмещает в себе сайт-презентацию и сайт-каталог. Сайт дает полную информацию о деятельности магазина, а также содержит текстовый контент в виде статей и видеоматериал, которые могут быть полезны для посетителей. На разработанном сайте есть каталог с фотографиями товаров, удобный поиск по каталогу и возможность скачать прайс лист. На рисунке 27 представлена индексная страница сайта, на которой в левом верхнем углу расположен логотип с названием магазина. Поиск по каталогу можно использовать как на главной, так и на остальных страницах веб-сайта, так же, как и сам каталог, который реализован как выдвижное меню «гамбургер». По центру индексной страницы расположен баннер с акциями, проходящими в магазине, который автоматически сменяется в определенный временной промежуток. В качестве контента для индексной страницы использован сезонный товар.

Рисунок 27 – Индексная страница разработанного веб-сайта

На сайте можно зарегистрироваться, заполнив 6 полей: с именем и фамилией, номером телефона и адресом электронной почты, и два поля для ввода пароля. Страница регистрации представлена на рисунке 28. Для регистрации использован обработчик php.

Рисунок 28 – Страница для регистрации разработанного веб-сайта

Для привлечения клиентов на веб-сайт, добавлены страницы со статьями, содержащие полезную информацию о строительстве, ремонте и прочих сферах, связанных с нашим магазином. В некоторых статьях добавлен видео материал с youtube-канала магазина ДомоХозяин. Одну из статей с видео материалом, можно увидеть на рисунке 29.

Рисунок 29 – Страница статьи с видеоматериалом на разработанном веб-сайте

Если выбрать определенный раздел каталога товаров, попадаем на странице с карточками товара. С помощью css grid и flexbox, удалось сделать все карточки одного размера (не зависимо от того сколько контента они содержат), выровненные как по горизонтали, так и по вертикали и подстраивающиеся под ширину экрана, без медиа запросов. При нажатии на карточку товара, можно получить о нем более подробную инормацию. На рисунке 30 представлена страница одного из разделов каталога – сантехника.

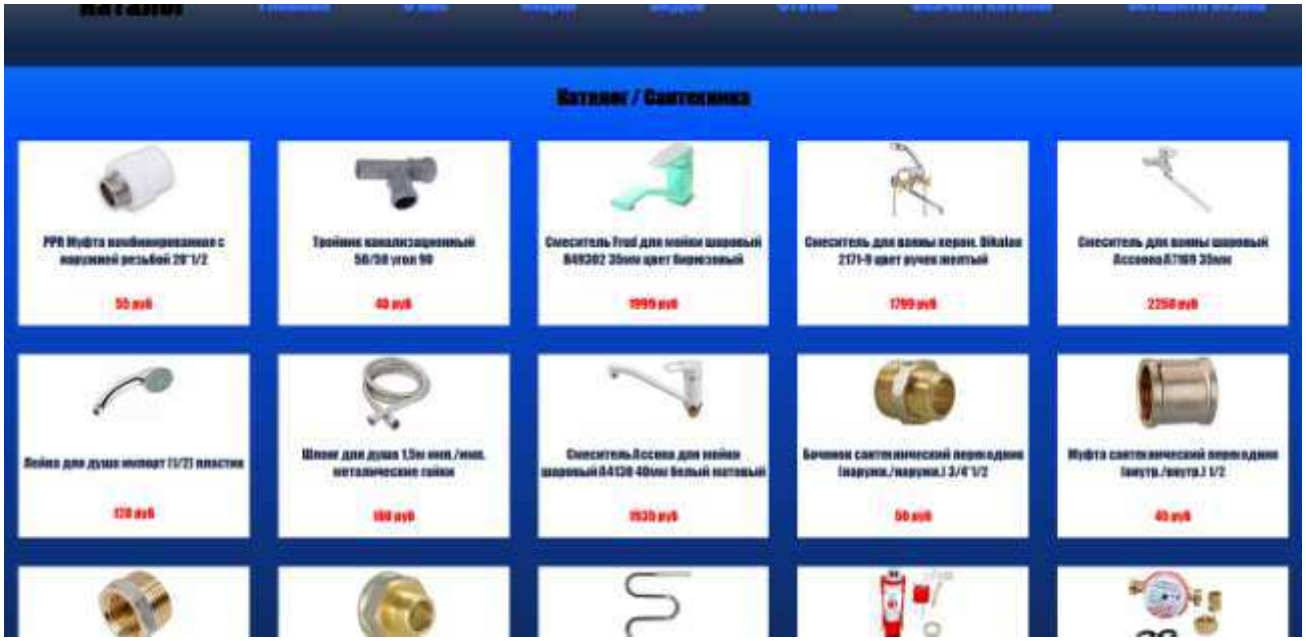

Рисунок 30 – Страница одного из разделов каталога

Любая страница разработанного веб-сайта адаптивно подстраивается под любую ширину экрана, что бы посетители могли комфортно пользоваться сайтом с мобильных, планшетов и других устройств. На рисунках 31 и 32, представлен скриншот индексной страницы сайта, с маленького смартфона с разрешением экрана 480*320.

Рисунок 31 – Индексная страница сайта на разрешении 480*320

Рисунок 32 – Индексная страница сайта на разрешении 320*480

На рисунках 33 и 34 представлена индексная страница сайта на экране большого смартфона с разрешением экрана 800*480.

Рисунок 33 – Индексная страница сайта на разрешении 800*480

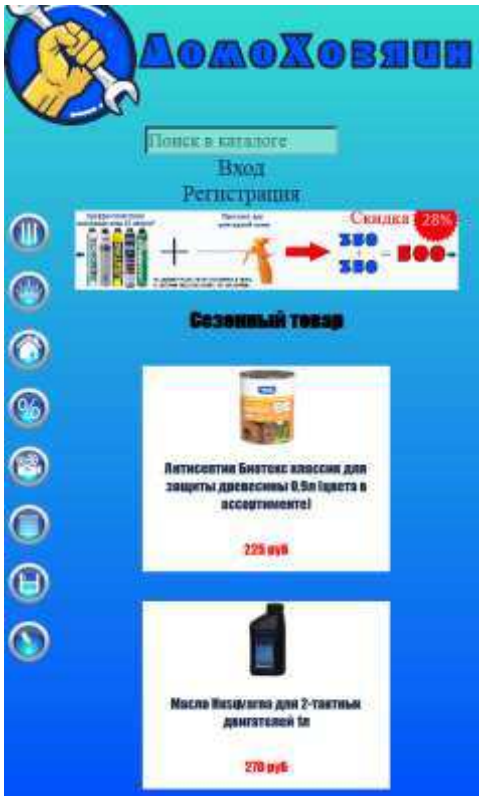

Рисунок 34 – Индексная страница сайта на разрешении 480*800

На рисунках 35 и 36 представлены скриншоты индексной страницы сайта

с планшета с разрешением экрана 1024*768 пикселей в горизонтальном и вертикальном положениях.

Рисунок 35 – Индексная страница сайта на разрешении 1024*768

Рисунок 36 – Индексная страница сайта на разрешении 768*1024

ЗАКЛЮЧЕНИЕ

В процессе выполнения бакалаврской работы были охарактеризованы веб-сайты магазинов строительных материалов. Было рассмотрено три сайта разных магазинов строительных материалов, они были охарактеризованы, у каждого из них есть свои плюсы и минусы, были сделаны определенные выводы, что стоит вносить в свой веб-сайт, а чего лучше избежать.

Описана работа магазина строительных материалов «ДомоХозяин» и сформированы требования к проекту веб-сайта. Описание работы магазина было нужно для того, чтобы понять, каким должен быть веб-сайт, какие функции он должен выполнять и какие страницы иметь. Затем были сформированы требования к проекту веб-сайта.

Описаны основные этапы и методы проектирования веб-сайтов. В процессе сбора информации на данную тему, были окончательно определены «инструменты» для проектировки веб-сайта, html, css и для некоторых функций php и java script. Был составлен план, по которому и производилась работа по проектировки сайта.

Было проведено исследование технологий и модулей, использующихся для современной верстки веб-сайтов, и выбраны оптимальные пути адаптации сайта под большинство устройств.

В результате бакалаврской работы был спроектирован и разработан веб-сайт, который отвечает всем требованиям организации и удовлетворяет все их пожелания.

Товар выставлен на сайте в виде карточек, содержащих информацию о товаре, его изображение и цену. Он рассортирован на определенные категории. На главной странице сайта выставляется сезонный товар, который постоянно будет обновляться.

Сайт работает быстро и без сбоев. Дизайн удобен и привлекателен. Простая поисковая система, поможет любому посетителю найти нужный ему товар. Удобная навигация не даст потенциальным клиентам заблудиться на

сайте. Контент, содержащий как текстовую информацию, так и видеоматериал, создан для привлечения внимания новых посетителей сначала на сайт, а затем и в магазин. Веб-сайт работает автономно и не требует установки дополнительного софта.

В проектировании веб-сайта были использованы современные модули и технологии адаптивного дизайна, такие как flexbox и css grid. Страницы веб-сайта адаптируются под любые устройства и гаджеты, поэтому пользователи получат одну и ту же информацию в удобном для каждого из них формате как с мобильных, так и со стационарных устройств.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Богданов-Катьков, Н.В. Интернет: Новейший справочник: учебное пособие / Н.В. Богданов-Катьков, А.А. Орлов. – Москва: Эксмо, 2012. – 928 с.

2. Евдокимов, Н.В. Основы контентной оптимизации. Эффективная Интернет-коммерция и продвижение сайтов в Интернет: учебное пособие / Н. В. Евдокимов. – Москва: Вильямс, 2010. – 160 с.

3. Белявский, О.В. Эффективная работа в сети Интернет: учебное пособие / О.В. Белявский, О.Л. Капилевич. – Москва: Триумф, 2008. – 176 с.

4. Создание Web-сайтов: Учебно-методическое пособие / Н.А Инькова, Е.А. Зайцева, Н.А. Кузьмина, С.Г. Толстых. – Тамбов: Изд-во Тамб. гос. техн. ун-та, 2005. – 56 с.

5. Орлов, Л. В. Web-сайт без секретов: учебное пособие / Л. В. Орлов. – Москва: Бук-пресс, 2006. – 512 с.

6. Борисенко, А.А. Web-дизайн. Просто как дважды два: учебное пособие / А.А. Борисенко. – Москва: Эксмо, 2008. – 320 с.

7. Дакетт, Д. HTML и CSS. Разработка и дизайн веб-сайтов: учебное пособие / Д. Дакетт. – Москва: Эксмо, 2013. – 480 с.

8. Дакетт, Д. Основы веб-программирования с использованием HTML, XHTML и CSS: учебное пособие / Д. Дакетт. – Москва: Эксмо, 2015. – 768 с.

9. Дронов, В. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов: учебное пособие / В. Дронов. – Санкт-Петербург: БХВ-Петербург, 2014. – 351 с.

10. Квинт, И. Создаем сайты с помощью HTML, XHTML и CSS: учебное пособие / И. Квинт. – Санкт-Петербург: Питер, 2014. – 448 с.

11. Мержевич, В.А. HTML и CSS на примерах: учебное пособие / В. А. Мержевич. – Санкт-Петербург: «БХВ-Петербург», 2016. – 448 с.

12. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5: учебное пособие / Р. Никсон. – Москва: Мир, 2016. – 688 с.

13. Пауэрс, Д. Adobe Dreamweaver, CSS, Ajax и PHP: учебное пособие / Д. Пауэрс. – Санкт-Петербург: БХВ-Петербург, 2014. – 982 с.

14. Прохоренок, Н. А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера: учебное пособие / Н.А. Прохоренок, В.А. Дронов. – Москва: РГГУ, 2015. – 768 с.

15. Ташков, П.А. Веб-мастеринг HTML, CSS, JavaScript, PHP, CMS, AJAX, раскрутка: учебное пособие / П.А. Ташков. – Москва: Книга по Требованию, 2012. – 512 с.

16. Бабаев, А. Создание сайтов: учебное пособие / А. Бабаев, М.М. Боде, Н.С. Евдокимов. – Санкт-Петербург: Питер, 2014. – 397с.

17. Полное руководство по Flexbox [Электронный ресурс] : справочник для разработчиков от 02.07.2017 / Chris Coyier – Электрон. журн. – Режим доступа: https://css-tricks.com/snippets/css/a-guide-to-flexbox/

18. Электронный ресурс по сбору статистики о веб-технологиях [Электронный ресурс] : база данных содержит сведения о всех видах плагинов поддерживаемых конкретными браузерами. – Режим доступа: https://caniuse.com/

19. Полное руководство по CSS Grid [Электронный ресурс] : справочник для разработчиков от 12.06.2017 / Chris House – Электрон. журн. – Режим доступа: https://css-tricks.com/snippets/css/complete-guide-grid/

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт педагогики, психологии и социологии
Кафедра современных образовательных технологий

УТВЕРЖДАЮ
Заведующий кафедрой
_____ И. А. Ковалевич
«_19_» _06_____ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 – Прикладная информатика

Разработка веб-сайта для магазина розничной торговли

Руководитель старший преподаватель _____ О.В. Шайдурова

Научный консультант доцент, канд.техн.наук _____ И.А. Ковалевич

Выпускник _____ А.И. Марьясов

Красноярск 2018