Содержание

Введение

1. Исследование сравнительных особенностей веб-сайтов

1.1 Оценка необходимости разработки проекта

1.2 Задачи, стоящие перед проектом

1.3 Сравнительные характеристики схожих продуктов

1.4 Выводы

2. Техническое задание

2.1 Основания для разработки

2.2 Цели разработки

2.3 Требования к программному продукту

2.3.1 Функциональные характеристики

2.3.2 Надежность

2.3.3 Защита информации от несанкционированного доступа

2.3.4 Антивирусная защита

2.3.5 Эргономика и техническая эстетика

2.3.6 Информационное наполнение

2.4 Условия выполнения проектных работ

2.5 Процедура контроля и приемки работ

3. Раздел проекта

3.1 Технология гипертекста — язык HTML

3.2 Проектирование структуры и архитектуры программного продукта

3.2.1 Декомпозиция задачи

3.2.2 Определение взаимосвязей между модулями и разделами

3.3 Инструменты для создания веб-страниц

3.4 Проектирование пользовательского интерфейса

3.5 Разработка визуального дизайна

4. Специальный раздел

4.1 Технический раздел

4.1.1 Характеристика проекта

4.1.2 Оценка необходимых вычислительных ресурсов

4.1.3 Гостевая книга для веб-сайта

4.1.4 Счетчик посещаемости

4.2 Рабочая версия проекта

4.3 Оптимизация и продвижение веб-сайта

5. Организационно-экономический раздел

5.1 Затраты на разработку и размещение сайта в Интернете

5.2 Выводы

6. Безопасность жизнедеятельности

6.1 Описание условий труда программиста

6.2 Требования к производственным помещениям

6.2.1 Цветовое оформление и коэффициенты отражения

6.2.2 Освещение

6.2.3 Параметры микроклимата

6.2.4 Уровень шума и вибрации

6.2.5 Электромагнитные и ионизирующие излучения

6.2.6 Эргономические требования к рабочему месту

6.3 Пожарная безопасность

6.4 Расчет уровня освещенности

6.5 Оценка уровня шума

Выводы

Заключение

Список тспользуемых источников

Приложения

# 

# Введение

Темой дипломного проекта является «Разработка веб-сайта по предоставлению услуг по дизайнерскому остеклению для компании GlassArt».

Причины выбора этой темы заключаются в следующем:

1. Углубленное понимание предметной области.

2. Опыт работы инженером в производственном отделе завода оконных систем.

3. Возможность продемонстрировать усвоение знаний по ключевым учебным дисциплинам и навыков в программировании.

Разработка функционального программного продукта, который будет иметь практическое применение.

Целью дипломного проекта является разработка программного продукта для производственной компании по дизайнерскому остеклению "GlassArt". Цель дипломного проекта заключается в создании программного продукта для компании "GlassArt", специализирующейся на дизайнерском остеклении. Данный программный продукт (далее ПП) представляет собой веб-сайт, который нацелен на эффективное продвижение продукции компании в интернете, улучшение имиджа и узнаваемости бренда, а также на увеличение объёмов продаж. Разработка данного проекта обусловлена необходимостью расширения функциональных возможностей как для пользователей, так и для администраторов, а также поддержания сайта компании на современном уровне.

Основной задачей дипломного проекта является разработка сайта для производственной компании, который будет удобным в использовании и информативным. Сайт не должен содержать избыточной или ненужной информации, а его структура должна обеспечивать легкий доступ к интересующим посетителя данным, при этом ненавязчиво привлекая его внимание.

# 1. Исследование сравнительных особенностей веб-сайтов

# 1.1 Оценка необходимости разработки проекта

На этапе разработки любого веб-ресурса осуществляется тщательный анализ целей его создания, а также прогнозируются возможные действия пользователей на сайте. Ресурс должен быть полезен как для посетителей, так и для его создателей, предоставляя актуальную, интересную и необходимую информацию, а также способствуя достижению поставленных задач.

Основная цель создания сайта для компании, занимающейся остеклением окон, дверей и лестниц, заключается в увеличении объема продаж. Важно заинтересовать потенциального покупателя и привлечь его внимание, чтобы он выбрал именно эту компанию для приобретения нужной продукции.

Посетитель сайта подобного рода обычно стремится выбрать наилучший вариант, учитывая такие факторы, как:

• соотношение цена-качество

• гарантии

• доставка

• и другие аспекты.

Во время своего пребывания на сайте у пользователя могут возникнуть дополнительные "второстепенные цели": ознакомиться с информацией о различных смежных предложениях, прочитать отзывы других клиентов, узнать о действующих скидках и акциях. После того как посетитель сделает свой выбор, ему понадобятся данные о способах оплаты, условиях доставки и сроках выполнения заказа. Поэтому при разработке веб-ресурса важно учитывать не только основные "стратегические" цели, но и все возможные "второстепенные" цели, которые могут возникнуть на разных этапах.

# 1.2 Задачи, стоящие перед проектом

• Продвижение компании в интернете

• Реклама продукции и услуг

• Информационная поддержка проекта

Исходя из поставленных задач, можно сформулировать общую концепцию создания веб-ресурса, а именно его тип:

Промо-сайт — это интернет-проект, предназначенный для привлечения внимания к определенному продукту или услуге. Его главная цель — донести информацию о новой акции, услуге или товаре до максимально широкой аудитории, подчеркнуть преимущества продукта и обеспечить его запоминаемость.

# 1.3 Сравнительные характеристики схожих продуктов

Существуют определенные рекомендации, созданные экспертами, для повышения удобства использования сайта. Если интерфейс сайта неясен и неудобен, маловероятно, что пользователь продолжит взаимодействие с таким продуктом.

Основные факторы, которые следует учитывать при оценке качества дизайна сайта и составлении технического задания:

1. Технические:

- скорость передачи данных (качество интернет-соединения);

- характеристики монитора (размер экрана, разрешение, количество цветов и т.д.), а также его качество;

- настройки просмотра, включая возможность отключения графики при посещении сайта;

- версия используемого браузера;

- параметры системы безопасности на компьютере пользователя (например, если он подключен к локальной сети организации);

- настройки браузера (размер и тип шрифта и т.д.);

- версия операционной системы на компьютере пользователя;

- географическое положение компьютера (город, страна, континент);

- особенности работы мыши (метод перемещения курсора, наличие колесика и т.д.

2. Социальные:

- уровень образования посетителя (начальное, среднее, высшее);

- должность посетителя (руководитель, менеджер, специалист по снабжению, бухгалтер, посредник и т.д.);

- финансовое положение посетителя;

- религиозная принадлежность посетителя.

3. Психологические:

- Мотивация (цели, уровень заинтересованности, ожидания):

- Насколько значима для пользователя информация, размещенная на сайте (уровень его интереса).

- Цели, которые ставит перед собой пользователь сайта.

- Чего ожидает пользователь от сайта.

- Личностные характеристики (характер, темперамент, предрасположенности);

- Психологическое состояние (мрачное, угрюмое, игривое, приподнятое и т.д.);

- Общее самочувствие пользователя;

- Степень усталости пользователя (усталый, энергичный, сонный и т.д.);

- Соответствует ли информация на сайте тому ключевому слову, по которому сайт был найден.

4. Физиологические:

- Зрение клиента (использует ли очки);

- Дальтонизм (цветовая слепота);

- Возраст клиента;

- Пол клиента (мужской или женский);

5. Дополнительные условия:

* Опыт пользователя и его знания о Интернете;
* Время суток (утро, день, обед, вечер, ночь);
* День недели, выходной или праздничный день;
* Частота посещения сайта (первый раз, второй раз, постоянный посетитель) и количество ранее просмотренных сайтов;
* Кто оплачивает доступ в Интернет - сам пользователь или компания;
* Индивидуальный или групповой просмотр (просматривает ли сайт один человек или несколько).

Анализ путей, по которым пользователи попадают на сайт, показывает, что основным источником трафика являются поисковые системы. После индексации сайта в этих системах, посетители могут сразу перейти не на главную страницу, а на внутренние разделы. Практика демонстрирует, что на крупных сайтах (с более чем 100 отдельными HTML-страницами) доля посетителей, попадающих на главную страницу, составляет 10% и менее. Остальные 90% пользователей начинают свой просмотр с внутренних страниц. Поэтому каждая страница сайта должна обладать удобной и интуитивно понятной навигацией. Важно, чтобы посетитель мог сразу определить, в каком разделе и на какой странице он находится.

Результаты:

- По содержанию продукты практически не отличаются друг от друга: стандартный набор предлагаемых товаров (окна, лестницы, откосы), контактные данные и т.д.;

- Некоторые из них выполнены немного лучше других с точки зрения удобства использования — интуитивно понятная навигация по сайту, максимальное количество информации при минимальных перемещениях;

- На большинстве сайтов используются всего три типа расположения навигационных ссылок: вертикальное, горизонтальное и комбинированное. Применение других форматов (например, кругового и т.п.) всегда приводит к неудобствам для пользователей, что негативно сказывается на посещаемости сайта.

# 1.4 Выводы

Таким образом, сайт должен быть удобным в использовании и информативным, не перегруженным лишней или ненужной информацией. Он должен представлять информацию таким образом, чтобы посетитель мог легко найти интересующие его данные, а также ненавязчиво привлечь его внимание к другим аспектам.

# 2. Техническое задание

# 2.1 Основания для разработки

Необходимо создать программный продукт (ПП) под названием «Разработка веб-сайта для предоставления услуг дизайнерского остекления компании GlassArt». Этот продукт будет направлен на эффективное продвижение продукции компании в интернете, улучшение имиджа и узнаваемости бренда, а также на увеличение объема продаж. Разработка обусловлена потребностью в расширении функциональных возможностей как для пользователей, так и для администраторов, а также в поддержании сайта компании на современном уровне.

Данный ПП будет полезен для людей, заинтересованных в продукции, предлагаемой компанией, включая:

- Остекление окон;

- Остекление дверей;

- Остекление лестниц;

- Изготовление панно и зеркал и другие услуги.

**2.2 Цели разработки**

Обеспечение пользователя исчерпывающей информацией о компании и её товарах.

# 2.3 Требования к программному продукту

# 2.3.1 Функциональные характеристики

ПП должен выполнять следующие функции:

1. Предоставлять информацию о компании:

• Общие сведения о компании;

• Опыт работы, награды и достижения;

• Данные о сотрудничестве;

• Партнёры компании.

2. Обеспечить пользователя каталогом товаров:

• Каталог товаров, организованный по категориям и имеющий древовидную структуру.

3. Содержать форум для возможности задать интересующие вопросы.

4. Предоставить полную контактную информацию о компании.

• Контактные адреса и номера телефонов;

• Данные о графике работы;

5. Содержать данные о вакансиях в компании:

• Перечень текущих вакансий;

Детальное описание каждой из вакансий.

# 2.3.2 Надежность

Одним из ключевых аспектов качества системы является ее надежность. Под надежностью подразумевается способность системы или ее отдельных элементов соответствовать заранее установленным требованиям в течение определенного времени при соблюдении заданных режимов и условий эксплуатации. Система должна обеспечивать восстановление информации в случае программно-аппаратных сбоев (например, отключения электропитания, отказов носителей информации, вирусных атак и т.д.), а также сохранять работоспособность при выходе из строя отдельных компонентов.

В системе необходимо предусмотреть обработку следующих аварийных ситуаций:

1) Неисправность общего или специализированного программного обеспечения комплекса автоматизации (отдельного автоматизированного рабочего места или сервера).

После сбоя серверной операционной системы или системы управления базами данных в процессе выполнения пользовательских задач должно быть обеспечено восстановление данных в базе данных до состояния, соответствующего последней успешно завершенной транзакции перед сбоем.

Время восстановления работоспособности после сбоев и отказов не должно превышать трех часов. При этом в указанный период не включается время, затраченное на развертывание и настройку специального программного обеспечения на серверах, а также время, необходимое для решения проблем с техническим обеспечением и установкой операционной системы.

2) Выход из строя отдельных компонентов технических средств комплекса автоматизации.

Неисправность одного из автоматизированных рабочих мест или сбой в канале связи локальной вычислительной сети между автоматизированным рабочим местом и сервером не должны приводить к остановке работы комплекса средств автоматизации.

3) Сбои или неисправности активного накопителя на жестком магнитном диске.

Система автоматизации должна обеспечивать возможность "горячей" замены неисправного или вышедшего из строя активного накопителя на жестком магнитном диске без приостановки работы системы и потери данных.

Также в системе автоматизации должна быть предусмотрена возможность восстановления данных с внешнего накопителя после восстановления активного накопителя.

4) Импульсные помехи, сбои или отключение электропитания.

Импульсные помехи, сбои или отключение электропитания не должны вызывать неисправности в технических средствах комплекса автоматизации и/или нарушать целостность данных. Отключение электропитания на срок до 15 минут не должно приводить к остановке работы комплекса автоматизации. Необходимо предусмотреть средства для уведомления пользователей о прекращении электропитания.

# 2.3.3 Защита информации от несанкционированного доступа

Меры по защите информации от несанкционированного доступа должны включать в себя:

• Защиту ресурсов системы от несанкционированного доступа из внешних телекоммуникационных сетей, таких как Интернет;

• Регистрацию системных событий и попыток несанкционированного доступа к защищаемым ресурсам с использованием как стандартных, так и дополнительных средств.

Средства защиты информации от несанкционированного доступа должны включать в себя:

• Встроенные механизмы защиты от несанкционированного доступа в сетевых операционных системах;

• Встроенные средства защиты в системах управления базами данных;

• Встроенные средства защиты в используемых приложениях;

• Аппаратно-программные комплексы для защиты автоматизированных рабочих мест авторизованных пользователей (администраторов) от несанкционированного доступа;

• Средства защиты серверов от несанкционированного доступа;

• Средства защиты межсетевых экранов, маршрутизаторов и другого коммуникационного оборудования от несанкционированного доступа;

* Системы контроля, управления и идентификации, которые должны обеспечивать процедуры идентификации пользователей, мониторинга подключения и управления их доступом к ресурсам системы;
* Средства экранирования, которые должны обеспечивать безопасное подключение к открытым сетям (например, Интернет).

# 2.3.4 Антивирусная защита

Антивирусные средства должны гарантировать эффективный контроль над возможными источниками внедрения компьютерных вирусов в систему:

• Сервер и рабочие станции пользователей (редакторов, администраторов) должны быть защищены антивирусным ПО;

• Установка антивирусного программного обеспечения и автоматическое обновление баз вирусных сигнатур на сервере должны осуществляться централизованно.

# 2.3.5 Эргономика и техническая эстетика

Дизайн системы должен соответствовать следующим требованиям в области эргономики и технической эстетики:

• Корректно отображаться в зависимости от типа подключения пользователя (например, пользователи, работающие через модем, и пользователи с высокоскоростным доступом), а также их личных предпочтений;

• Иметь оптимизированный объем графических элементов для обеспечения максимальной скорости загрузки страниц сайта;

• Обеспечивать простую и понятную идентификацию раздела портала, в котором находится пользователь;

• Предоставлять интуитивно понятную навигацию, позволяющую пользователю тратить минимальное время и усилия на перемещение по страницам сайта;

• Включать развитую систему поиска информации как через контекстный поиск, так и через древовидные списки документов;

• Корректно отображаться на всех возможных разрешениях и с любым количеством одновременно отображаемых цветов на мониторе, при этом минимальное разрешение браузера должно составлять 800x600 пикселей.

• Сохранять единообразие отображения при использовании веб-браузеров Microsoft Internet Explorer версии 5.0 и выше, Mozilla Firefox 2.0 и выше, Opera 5.0 и выше;

• Иметь систему контекстных подсказок на страницах, где пользователи могут столкнуться с трудностями;

• Гарантировать приемлемое качество печати страниц портала на принтере;

• Обеспечивать полную обратную связь с пользователем через веб-браузер без необходимости использования почтовых клиентов или других программ;

• Корректно отображать информацию на компьютерах без предустановленных флэш-модулей и с отключенной поддержкой скриптов и других технологий;

• Предусматривать возможность поддержки многоязычного контента;

Содержать полный набор метаданных для эффективного индексирования поисковыми системами и правильного автоматического выбора кодировки.

# 2.3.6 Информационное наполнение

Изменение содержимого разделов должно выполняться через администраторский веб-интерфейс (систему управления контентом), который не требует специальных навыков программирования. Интерфейс должен обеспечивать возможность редактирования иерархии страниц сайта, управления правами доступа к ним с учетом групп пользователей, а также редактирования информационного содержания страниц. Наполнение страниц должно осуществляться с использованием заранее подготовленных шаблонов. Также необходимо предусмотреть возможность изменения этих шаблонов, добавления ссылок и прикрепления файлов различных форматов, а также форматирования текста, графики, таблиц и мультимедийных материалов. Должна быть возможность экспорта данных из файлов MS Word и Excel, а также вставки HTML-кода. Кроме того, необходимо обеспечить возможность редактирования дизайна страниц через административный интерфейс.

# 2.4 Условия выполнения проектных работ

Заказчик должен предоставить исполнителю всю необходимую входную информацию на электронном носителе (USB Flash Drive) или через электронную почту с подтверждением получения.

Исполнитель обязан представить готовый проект к согласованной обеими сторонами дате, указанной в договоре на разработку интернет-ресурса.

# 2.5 Процедура контроля и приемки работ

ПП принимается после завершения всех работ исполнителем. Он должен пройти следующие ключевые этапы испытаний:

1. Предварительные испытания;

2. Опытная эксплуатация;

3. Приемочные испытания.

Испытание программного продукта должно проводиться через тестирование с целью обнаружения потенциальных ошибок и недостатков. Оцениваются функциональность программного обеспечения и его соответствие требованиям, изложенным в документе "Техническое задание". Предварительные испытания завершаются составлением акта приемки для опытной эксплуатации.

Опытная эксплуатация осуществляется для оценки технологичности внедренного процесса, проверки работы системы в целом, а также корректности выполнения поставленных задач и реализованных функций.

Данная эксплуатация должна проводиться с использованием полного объема реальных данных, которые могут быть загружены из существующей системы или введены вручную через разработанный интерфейс. В ходе опытной эксплуатации необходимо вести журнал, в котором будут фиксироваться результаты выполненных работ, замечания по функционированию программного обеспечения и предложения по его улучшению.

В ходе опытной эксплуатации разработчики вносят изменения в программное обеспечение на основе полученных замечаний и предложений, а также утверждают техническую документацию. Результаты опытной эксплуатации фиксируются в акте о её завершении.

Решение о переводе системы в постоянную эксплуатацию принимает комиссия, назначаемая распоряжением о проведении приемочных испытаний. Комиссии предоставляется доработанное программное обеспечение (включая программы и документацию), скорректированное по итогам опытной эксплуатации. Комиссия проверяет полноту выполненных доработок, а также комплектность и качество технической документации. На основании анализа выполненных работ и журнала опытной эксплуатации комиссия принимает решение о допуске системы к постоянной эксплуатации.

На заключительном этапе стороны подписывают акт сдачи-приёмки работ по договору, и исполнитель передаёт заказчику всю необходимую электронную информацию и техническую документацию.

# 3. Раздел проекта

# 3.1 Технология гипертекста — язык HTML

Язык разметки HTML является основным стандартом для создания электронных документов, размещаемых в Интернете. HTML не является языком программирования; его основная функция заключается в разметке текстовых документов, то есть в форматировании текста.

HTML-документ представляет собой текстовый файл с расширением .html. Он содержит текстовую информацию, которая отображается в окне браузера, а также стандартные команды, называемые тегами.

Некоторые теги предназначены для форматирования текста и указывают браузеру, как должен выглядеть тот или иной элемент, например, заголовок, подзаголовок или основной текст. Другие теги используются для вставки графики, аудио и видео. Однако возможности HTML довольно ограничены.

Для обеспечения более гибких и разнообразных способов оформления элементов веб-страницы была создана технология CSS (Cascading Style Sheets — каскадные таблицы стилей). CSS расширяет возможности HTML в управлении отображением элементов веб-страниц на любых устройствах, которые могут использоваться посетителями для взаимодействия с информацией в сети WWW. Кроме того, эта технология значительно упрощает процесс редактирования как отдельных документов, так и всего ресурса в целом.

# 3.2 Проектирование структуры и архитектуры программного продукта

# 3.2.1 Декомпозиция задачи

Цель декомпозиции заключается в выявлении модулей и процедур низшего уровня, на основе которых можно создать функциональную систему, максимально использующую принцип сборки программ из готовых компонентов. Этот подход соответствует общей концепции современных технологий программирования.

Структурирование программ осуществляется прежде всего для упрощения разработки, программирования, отладки и внесения изменений в программный продукт.

# 3.2.2 Определение взаимосвязей между модулями и разделами

Связь между модулями будет осуществляться через гиперссылки. Гиперссылки представляют собой соединения, которые направляют от одной веб-страницы к другой. Они создаются с использованием специального парного тега <a> и имеют следующий формат: <A HREF=". /about.html»> Страница N125</A>. Тег <a> включает атрибут href, который указывает интернет-адрес страницы, на которую будет осуществлен переход при нажатии на гиперссылку. Этот атрибут обязательно должен присутствовать в каждом теге <a>, который определяет гиперссылку.

# 

# 3.3 Инструменты для создания веб-страниц

Web-страницу можно создать с помощью простого текстового редактора, такого как Блокнот (Notepad), используя язык разметки HTML. Этот метод называется ручным кодированием.

Существует и другой способ — использование специализированных программ, известных как веб-редакторы.

Веб-редакторы делятся на несколько типов:

1. Не визуальные редакторы — они работают непосредственно с кодом, обеспечивая быстрый ввод тегов, подсветку синтаксиса и проверку корректности ссылок.

2. Визуальные редакторы — позволяют работать с графическим представлением элементов страницы, такими как текст, изображения и таблицы.

Существуют программы, которые объединяют удобные функции HTML-редакторов с возможностью предварительного просмотра веб-страницы в визуальном редакторе. Такие редакторы называются гибридными. Одним из примеров такого редактора является программа Dreamweaver.

Программное обеспечение Adobe Dreamweaver предоставляет дизайнерам и разработчикам возможность создавать веб-сайты профессионального уровня, соответствующие отраслевым стандартам. Оно поддерживает визуальный дизайн и редактирование кода, разработку страниц для систем управления контентом, а также обеспечивает точное тестирование совместимости с различными браузерами.

# 3.4 Проектирование пользовательского интерфейса

Проектирование пользовательского интерфейса является ключевым этапом в создании интернет-приложений. Качество навигационной концепции влияет на множество критически важных аспектов, которые необходимо учесть. К числу основных можно отнести следующие:

* Удобная и интуитивно понятная навигация по сайту;
* Легкость в поиске нужной информации;
* Выделение важных рабочих областей для пользователя;
* Простота восприятия представленных данных.

# 3.5 Разработка визуального дизайна

Среди графических редакторов можно выделить мощные профессиональные программы, такие как Adobe Photoshop, Adobe Illustrator, 3D Studio и CorelDraw. Эти программы являются довольно объемными и предлагают множество вспомогательных инструментов и эффектов, что делает их графическими пакетами. Существуют также более простые и компактные редакторы, такие как Paintshop Pro, Photofinish и Paint, входящий в состав Windows. Графические редакторы делятся на два типа: растровые (например, Adobe Photoshop, Paintshop, Paint, GIMP), которые создают изображения, основываясь на пикселях, каждому из которых присвоены цвет и яркость, и векторные, которые рисуют целые линии — дуги и отрезки, представляя сложные формы как комбинацию таких элементов.

GIMP (GNU Image Manipulation Program) — это растровый графический редактор, предназначенный для создания и редактирования растровой графики, с частичной поддержкой векторной графики. Проект был основан в 1995 году Спенсером Кимбеллом и Питером Маттисом в рамках дипломной работы и в настоящее время поддерживается группой добровольцев.

С помощью GIMP можно выполнять различные задачи, такие как создание графики и логотипов, изменение размеров и обрезка фотографий, раскрашивание, комбинирование изображений с использованием слоёв, а также ретуширование и конвертация изображений в разные форматы.

Графический редактор CorelDRAW предназначен для работы с векторной графикой и занимает лидирующие позиции среди аналогичных программ. Его популярность обусловлена широким набором инструментов для создания и редактирования графических изображений, интуитивно понятным интерфейсом и высоким качеством получаемых результатов. CorelDRAW особенно удобен для создания иллюстраций, состоящих из множества рисунков, фотографий и текстов. В пакет CorelDRAW также входит редактор растровой графики Corel Photo-Paint. Интерфейсы обеих программ схожи, а их функциональные возможности прекрасно дополняют друг друга. С помощью этих двух программ можно выполнить практически любую, даже самую сложную, графическую задачу.

Одним из графических редакторов, предназначенных для создания растровых изображений, является Adobe Photoshop — программа, разработанная компанией Adobe Systems. Этот продукт занимает лидирующие позиции на рынке коммерческих средств редактирования растровой графики и является наиболее известным предложением компании Adobe. Важным преимуществом Adobe Photoshop по сравнению с конкурентами является автоматизация множества рутинных задач, что позволяет создавать сложные изображения в кратчайшие сроки.

# 4. Специальный раздел

# 4.1 Технический раздел

# 4.1.1 Характеристика проекта

Проект представляет собой систему управления базами данных и набор скриптов, которые позволяют выполнять выборку, анализ и вывод информации из базы данных, а также вводить необходимую информацию клиентом (пользователем) для оформления заказа.

В общем, проект включает в себя:

* источники данных — базы данных, таблицы и запросы;
* документы — формы и отчеты;
* программы (скрипты);
* вспомогательные HTML-файлы.

# 4.1.2 Оценка необходимых вычислительных ресурсов

Минимальные системные требования:

1. Для сервера:

• Процессор: Intel Pentium/Celeron, AMD K6, K7/Duron/Athlon и совместимые с x86, с тактовой частотой от 500 MHz;

• Оперативная память: минимум 512 Mb;

• Видеокарта и монитор: любые, совместимые с необходимой операционной системой; возможна работа без монитора и видеокарты (управление осуществляется удалённо);

• Клавиатура и мышь: любые, совместимые с необходимой операционной системой; возможна работа без клавиатуры и мыши (управление осуществляется удалённо);

• Жёсткий диск: минимум 200 Gb;

• Пропускная способность канала: от 2 Mbit;

• Операционная система: семейства UNIX (в редких случаях Windows на базе NT);

* Поддержка протоколов: TCP/IP (FTP, HTTP, POP3);
* Веб-сервер: Apache и PHP.

2. Для клиента:

• Процессор: Intel Pentium/Celeron, AMD K6, K7/Duron/Athlon и x86-совместимые с тактовой частотой от 300 MHz;

• Память: от 128 Mb;

• Видеоадаптер: от 8 Mb;

• Монитор: 1024x768 и выше;

• Клавиатура и мышь: любые, поддерживаемые требуемой ОС;

• Жёсткий диск: от 20 Gb;

• Пропускная способность канала: от 33,6 кбит/с;

• ОС: любая, поддерживаемая работу со стеком протоколов TCP/IP;

Поддержка протоколов: TCP/IP (FTP, HTTP, POP3).

Необходимое ПО:

1. Для клиента:

• сёрфинг в сети: браузер (поддерживаемый JavaScript);

1. Для администратора Интернет-магазина:

• сёрфинг в сети: браузер (поддерживаемый JavaScript);

• просмотр/редактирование документов: текстовый редактор (любой), браузер (любой);

• отправка электронных писем: e-mail клиент (любой);

доступ к серверу через FTP протокол: ftp-клиент (любой).

# 4.1.3 Гостевая книга для веб-сайта

Большинство серверов в сети предоставляют возможность использовать бесплатную гостевую книгу. Установить её очень легко и удобно, и для этого не нужно получать разрешение на запуск программ на сервере. Всё, что требуется, — это добавить несколько строк в HTML-код вашей страницы.

Обзор гостевых книг:

WR Lite Guest — это гостевая книга, разработанная на PHP, использующая текстовые файлы в качестве базы данных. Она предлагает минимальный набор функций, сосредоточенный на самых необходимых возможностях с низкой нагрузкой на систему.

Данная гостевая книга включает в себя все стандартные функции, характерные для подобных скриптов:

• Защита от спама;

• Возможность вставки смайлов;

• Система антифлуда;

• Удаление и редактирование сообщений администратором;

• Отправка уведомлений на e-mail администратору;

• И многое другое.

WR Lite Guest — это прекрасный вариант для небольших сайтов, установка и интеграция дизайна занимают всего несколько минут.

BlackBook — это бесплатный скрипт полноценной гостевой книги, который использует текстовые файлы для хранения данных вместо базы данных MySQL.

BlackBook не имеет значительных отличий от аналогичных скриптов, однако стоит выделить несколько особенностей, таких как:

• Легкость установки, настройки и управления;

• Возможность загрузки изображений пользователями;

• Поддержка смайлов;

• Удобная форма регистрации с множеством полей;

• Многоязычная поддержка;

BlackBook является отличным решением для самых требовательных вебмастеров.

Домашняя страница скрипта гостевой книги MBook mbook. mod-site.net

Однопользовательский скрипт гостевой книги имеет практически те же функциональные возможности, что и гостевые книги нашего сервиса. Отличие состоит Основные технические характеристики однопользовательского скрипта гостевой книги MBook:

• Язык: Perl

• СУБД: MySql

• Размер дистрибутива: ~1МБ в сжатом виде, ~2МБ в разжатом виде

Возможности гостевых книг MBook v3.2:

• Отсутствие баннеров (мы гарантируем, что они не появятся в наших гостевых книгах).

• Более 20 кодов для форматирования текста

• Режим форума: посетители гостевой книги могут комментировать сообщения.

• Возможность предварительного просмотра сообщения перед добавлением в гостевую книгу.

• Поиск по гостевой книге.

• Страница с правилами гостевой книги.

• Межбраузерная совместимость: гостевые книги протестированы в браузерах Internet Explorer 6, Netscape 7.0, Opera 7.11, Mozilla 1.0.

• Уведомления о новых сообщениях и комментариях для владельца книги и для посетителей.

• Возможность удалять, редактировать сообщения и отвечать на них прямо из письма-уведомления.

• Возможность создавать несколько гостевых книг на одном аккаунте. При этом одни книги могут использовать дизайн или сообщения других книг (можно создавать книги с одинаковым дизайном или одинаковым содержанием).

• Гибкая система защиты от хулиганов: отслеживание одинаковых сообщений, установка максимального количества сообщений для одного посетителя в день, установка минимального времени между двумя сообщениями, разрезание длинных слов для исключения "раздвигания" книги, запрет отдельных IP-адресов и групп IP-адресов, возможность запрета просмотра гостевой книги для отдельных личностей, возможность запрета оставлять сообщения с анонимных прокси-серверов и др.

• Защита email-адресов от спаммеров с помощью JavaScript.

• Возможность отправить из гостевой книги личное сообщение владельцу книги (такое сообщение не будет опубликовано).

• Расширенная статистика посещаемости, включая статистику по каждому посетителю гостевой книги.

• Возможность изменения порядка вывода сообщений и комментариев на экран

Сохранение содержимого полей оставленного сообщения. В следующий раз, когда посетитель захочет оставить сообщение в Вашей гостевой книге, необходимые поля (имя, email, город, icq) будут заполняться автоматически.в том, что однопользовательский скрипт гостевой книги позволяет создать только одну гостевую книгу для одного домена.

# 4.1.4 Счетчик посещаемости

Как показывает опыт, в продвижении сайтов хороший счетчик является самым необходимым и полезным инструментом для оценки ваших действий в ходе рекламной кампании. Несмотря на существование неких общих правил и принципов, подробно описывающих последовательность проведения рекламной кампании, реакция аудитории на то или иное действие всегда индивидуальна и может дать совершенно непредсказуемый результат. По этой причине необходимо постоянно отслеживать поток посетителей и просчитывать эффект от проводимых акций.

В интернете можно найти счетчики на любой вкус:

1) Rambler - известный российский сервер. Предоставляемая информация: общее количество посещений, среднее количество посещений в день, в час, сводные отчеты по хостам посетителей, количество возвратов на вашу страницу в течение часа, рейтинг в разных группах страниц за разные периоды.

2) SpyLOG - это уникальный счетчик, позволяющий проводить разнообразные и сложные опросы пользователей сайта. На данный момент он предлагает более 600 различных статистических данных, сгруппированных в более чем 50 динамических форм. Помимо того, что он позволяет отслеживать, откуда приходят посетители, через какой браузер и с каким разрешением экрана они просматривают страницы, он также может отслеживать их путь по сайту, время, проведенное на сайте, ключевые слова, которые ищут в поисковых системах, и многое другое.

Легкий счетчик - можно подсчитать до 200 страниц, а результаты отображаются в простой и удобной форме. Вы можете получать ежедневные, еженедельные и ежемесячные отчеты по электронной почте. Начальное значение счетчика можно установить самостоятельно. Есть дополнительный счетчик, показывающий количество посещений после сброса, который можно сбрасывать вручную или автоматически ежедневно, еженедельно или ежемесячно.

3) JCount - это счетчик на базе Java. Поскольку он предоставляется рекламодателем бесплатно, размер счетчика совпадает со стандартным размером рекламного баннера. Через регулярные промежутки времени отображается рекламная информация и данные вашего счетчика. Регистрация очень проста. После регистрации вы можете получить код счетчика на странице рекламодателя или получить его по электронной почте. Если вы измените адрес своей страницы в тексте кода счетчика, вам не придется ничего менять.

Чтобы разместить счетчик на своем сайте, нужно зарегистрироваться и указать адрес своего сайта.

# 4.2 Рабочая версия проекта

Модуль "Главная".

Модуль является статической страницей и содержит информацию о компании "GlassArt".

Модуль "Каталог".

Модуль отображает информацию о производимой продукции.

Модуль "Услуги".

Модуль отображает информацию о предоставляемых услугах.

Модуль "Работы"

Модуль отображает информацию о выполненных работах.

Модуль "FAQ".

Модуль отображает информацию о часто задаваемых вопросах.

Модуль "Контакты":

Модуль "Контакты" отображает контактные данные о компании.

# 4.3 Оптимизация и продвижение веб-сайта

Оптимизация сайта.

Оптимизация - в широком смысле - это комплекс мер, направленных на повышение популярности сайта, увеличение посещаемости и продаж. В более узком смысле под оптимизацией понимается, в принципе, работа внутренних факторов, то есть это внутренняя оптимизация страниц или всего сайта для дальнейшего продвижения. Внутренняя оптимизация сайта напрямую влияет на его продвижение и, при отсутствии конкуренции в поисковых системах, напрямую повышает его рейтинг.

Сайт должен быть максимально оптимизирован под требования поисковых систем. Сайт должен иметь правильный дизайн и правильное содержание, т. е. контент, наполнение.

Поисковые системы - это интернет-системы, созданные для того, чтобы помочь пользователям найти нужную им информацию в сети, например, yandex.ru.

При создании сайта были учтены требования поисковых систем, и сайт был оптимизирован:

Размер страницы. Поисковые системы, в отличие от пользователей, не распознают графику. Единственное, что для них важно, - это кодировка страницы. Средний размер страницы сайта составил 12-15 Кб.

Чистый код. Это означает, что страница загружается быстро, код легко читается, а общая структура легко читается. JavaScript-код большого размера увеличивает время загрузки страницы, что негативно сказывается на продвижении в поисковых системах.

Графические элементы оформляются с помощью каскадных таблиц стилей (CSS), которые могут значительно облегчить страницу.

Текстовые блоки. Используйте легко читаемые шрифты; лучше всего подходят средние шрифты, выполненные черным по белому.

Поисковые системы часто не блокируют огромные сайты и игнорируют маленькие, поэтому, если вы используете гиперссылки для перехода от одной к другой, страницы вашего сайта будут добавляться постепенно.

Включение ключевых слов в заголовки.

Ключевые слова - слова, которые наиболее полно характеризуют содержание html-страницы и являются ключевыми словами для этой страницы. Поисковые системы используют эти ключевые слова для определения релевантности страницы определенному запросу.

Метатег используется для указания служебной информации о html-странице (< META>).

Тип копии - тип документа и его кодировка. Выбор кодовой страницы для корректного отображения текста браузером. Например,

<head> <meta http-equiv="content-type" content="text/html; charset=windows-1251"> < /head>

windows-1251 - кириллический символ, кодировка, обозначающая русский язык.

content-language - указывает язык документа. Значение этого атрибута используется как поисковыми роботами, так и веб-серверами. Например,

< meta http-equiv="content-language" content="ru">

Атрибуты группы метатегов NAME:

description - Описание документа. Это один из важнейших параметров, влияющих на ранжирование страницы поисковыми системами. Описание обычно представляет собой краткое изложение информации, содержащейся на странице. В описании также должны быть использованы ключевые слова. Содержимое метатега description также отображается поисковыми системами на странице запроса сразу под содержимым метатега TITLE. Пользователь, прочитавший его, обычно решает, стоит ли ему посещать эту страницу. Пример HTML-кода.

<meta name="description" content="Изготовление и остекление: металлоконструкции, окна, двери и лестницы">.

KEYWORDS - Ключевые слова. Введите ключевые слова или фразы, которые будут использоваться на этой странице. Вводите их через запятую. Примеры HTML-кода.

<meta name="keywords" content=" Остекление окон, дверей, Остекление конструкций г. Тамбов, исторические окна, алюминиевые окна, скандинавские, немецкие, стеклопакеты, окна, двери, лестницы">.

Выбор домена.

Домен или доменное имя - это название адреса в Интернете. В настоящее время адрес каждого компьютера в Интернете (и компьютера, на котором расположен ваш сайт) описывается с помощью IP-адреса (группа из четырех чисел от 0 до 255 и запись, например, 123.45.67.89). Без доменных имен каждый сайт пришлось бы адресовать по IP-адресу, что всем кажется очень неудобным. С другой стороны, без доменных имен вы можете иметь только один сайт на одном компьютере, в то время как с доменными именами вы можете иметь много сайтов на одном сервере, то есть на одном IP-адресе.

Доменные имена регистрируются официальным регистратором доменов. В России это Региональные сетевые информационные центры.

Домены имеют разные уровни: первый, второй, третий и т. д. Каждый домен уникален, и нет двух одинаковых. Домены первого уровня: com, net, ru, info, biz, tsa, de и т. д. Под этими доменами располагаются домены второго уровня, такие как, например, google.com и rambler.ru, banner.org.ru, soft.narod.ru, shops.com.UA, banner.kiev.UA и т.д. Домены первого уровня делятся на две группы: территориальные домены (например, ru, de, us) и экстерриториальные (например, com, org, net). Домены первого уровня также называются зонами. Например, домен любого уровня в географической зоне .ru относится к России. Любой желающий может зарегистрировать домен второго уровня в домене первого уровня.

Многие хостинг-провайдеры предлагают домены второго уровня .ru, .com, .net, .org, .ws, .info и т. д. Заказ хостинга бесплатен.

Выбор имени веб-сайта (доменного имени).

Необходимо выбрать простое, легко запоминающееся имя. Это очень важный момент в процессе продвижения интернет-ресурса. Хорошо подобранное, запоминающееся имя (URL, то есть адрес, с которого посетители приходят на сайт) может служить хорошей рекламой и давать дополнительное преимущество перед сайтами конкурентов с аналогичным содержанием.

Выбор хостинга.

Хостинг - это размещение веб-сайта путем аренды дискового пространства у провайдера.

Веб-хостинг (или виртуальный хостинг) - это комплекс услуг по обслуживанию аппаратного и программного обеспечения, в рамках которого сайт клиента размещается на сервере хостинг-провайдера. В этом случае провайдер берет на себя ответственность за правильное функционирование сервера и обеспечивает своевременное обновление программного обеспечения. Услуги хостинга также обычно включают в себя службу поддержки. Хостинг делится на платный и бесплатный.

Бесплатный хостинг. Подходит для простых сайтов. При бесплатном хостинге вам предоставляется место для размещения вашего сайта на одном из крупных серверов. Обычно с каждым сайтом, размещенным на таком сервере, хостер вешает свои рекламные баннеры. Бесплатные сайты могут быть размещены на: narod.ru, mail.ru, agava.ru, holm.ru и т.д.

Платный хостинг - если вы хотите иметь серьезный проект (сайт) в интернете надолго, а не на один - два месяца, я бы сразу обошелся доменом 2-го уровня и, у некоторых хостинг-провайдеров, одним годом (который потом можно продлить) Я не советую вам думать о покупке платного хостинга за тысячу рублей (можно начать покупать за 3 или 6 месяцев).

Дисковое пространство измеряется в мегабайтах: страница текста обычно занимает 10 килобайт; 100 килобайт - это 1/10 мегабайта. Это означает, что на 1 мегабайте дискового пространства можно хранить примерно 10 фотографий или 100 страниц текста. Для простого расчета необходимо выбрать размер дискового пространства. Чем больше дискового пространства, тем лучше, так как в будущем вам придется редактировать свой сайт, добавлять информацию, ссылки с других сайтов, партнерские программы и т. д.

Продвижение сайта.

Раскрутка сайта - самый важный процесс для любого сайта в интернете. Это бесконечный процесс, который нельзя остановить даже на один день. Интернет - это огромное количество информации. Каждый день появляются новые сайты, которые конкурируют с вашим в плане продвижения. Если вы просто разместите свой сайт в Интернете и решите, что этого достаточно, ваш сайт быстро окажется на дне Интернета, где его никто не сможет найти. Чтобы о вашем сайте узнали, нужно постоянно выводить его в свет.

Продвижение в поисковых системах.

Одним из эффективных способов привлечения посетителей на ваш сайт (раскрутки) является регистрация вашего ресурса в каталоге.

Вам необходимо составить список поисковых систем и приступить к регистрации.

Выбрав конкретную поисковую систему, вы должны найти в ней пункт меню, например, «Добавить сайт» или «Предоставить ресурсы». Некоторые из этих пунктов хорошо спрятаны в каталоге и в некоторых поисковых системах, поэтому вам придется проявить изобретательность, чтобы найти их. Затем вам нужно следовать инструкциям на экране, которые немного отличаются в разных системах.

Обмен ссылками.

Это еще один очень хороший метод продвижения. Обмен ссылками - это когда вы размещаете ссылку на другом сайте, а он в ответ размещает ссылку на вашем сайте. Эти ссылки приводят к вам посетителей.

Рекомендуется обмениваться ссылками с родственными сайтами, тогда посетители будут целевыми. Ссылки с других сайтов повышают вашу «цитируемость», или индекс цитирования - CI (определение ценности). Поисковые системы отображают сайты с высоким CI в верхней части результатов поиска.

Ссылки не должны быть перенаправленными или написанными на javascript. Такие ссылки практически бесполезны. Существует множество скриптов для обмена ссылками, но следует помнить, что поисковые системы постоянно меняют свои алгоритмы.

Страницы со ссылками должны находиться в открытых для индексации местах и иметь ссылки с главной страницы сайта. Также важно контролировать ответное размещение ссылок и их эффективность.

Баннерная реклама.

Баннерная реклама - это согласованные или оплаченные графические элементы, которые появляются на странице WWW-сервера и служат ссылкой на этот сайт или страницу.

Существует несколько критериев для размера баннера, поэтому прежде чем рисовать или заказывать баннер, полезно решить, кто будет его размещать, и ознакомиться с правилами выбранной системы или сервера.

Когда баннер готов, необходимо приступить к его размещению. Есть два основных способа решения этой задачи:

1. Договориться индивидуально с конкретным сервером (платно или на условиях обмена).

2. Размещение заказа в агентстве, предлагающем размещение на нескольких сайтах.

Первый способ явно предпочтительнее для малобюджетных веб-проектов и для страниц, созданных скорее для личных нужд, чем для серьезного коммерческого использования. Во-первых, необходимо связаться по электронной почте с администраторами и продюсерами интересующих сайтов и заключить с каждым из них договор. Также необходимо ознакомиться со статистикой посещений различных страниц требуемого сервера и оценить время, которое потребуется для организации заказанного количества просмотров.

Очень полезно договориться о размещении баннеров на специализированных тематических серверах или на серверах, которые не просто близки по направлению, но где ссылки с нашего сайта будут отдельным дополнительным ресурсом, а не переадресацией на страницу конкурента. Необходимо выявить такие сайты и написать короткое, вежливое и не обязательно комплементарное письмо их владельцу или продюсеру со скромной просьбой о размещении нашего баннера.

Внесетевая реклама и формирование аудитории.

Внесетевую рекламу можно разделить на рекламу в Интернете и рекламу за его пределами. Прежде чем рекламировать свою деятельность в Интернете, необходимо ознакомиться с правилами этикета, действующими в Сети. Эти правила, известные как «нетикет» или netiquette, призваны обеспечить устойчивую работу вдали от всего, что не регулируется и не регламентируется официальными законами сетевого веба. В результате эти правила фактически обязывают провайдеров, то есть тех, кто предоставляет доступ в интернет. Если отдельный провайдер отказывается следовать правилам, другие провайдеры откажутся обмениваться с ним информацией.

Поэтому рекламу в Интернете следует размещать с осторожностью. Существует два вида интернет-технологий: конференцсвязь и электронная почта.

Действия.

- Включите адрес веб-сайта (URL) в подпись, которая появляется во всех ответах по электронной почте и на конференциях.

- Присоединяйтесь к группам по интересам с целью привлечения новых посетителей на веб-сайт.

- Заведите новую тему и представьте в ней свою страницу.

Реклама вне Интернет-сайта осуществляется в соответствии с обычными правилами рекламной кампании. Не забывайте указывать адрес сайта, а также другие сведения о вашей компании на визитных карточках, брошюрах, листовках, плакатах, рекламных объявлениях в журналах и т. д.

# 5. Организационно экономический раздел

# 5.1 Затраты на разработку и размещение сайта в Интернете

Расходы на создание веб-сайта включают в себя электроэнергию, стоимость публикации продукта в Интернете (хостинг), зарплату программистов и курьеров, а также любые другие расходы, такие как канцелярские товары и компьютерные принадлежности. Такие расходы, как аренда офиса и амортизация компьютеров и оргтехники, относятся к основному офису.

Таблица 5.1 - Расчет электроэнергии для девятичасового рабочего дня.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование | кол-во | кВт/час | кВт в сутки (примерно) | кВт в месяц |
| Компьютер | 1 | 0,17 | 1,53 | 45,9 |
| Освещение | 3 | 0,36 | 9,72 | 291,6 |
| Сплит | 1 | 0,7 | 6,3 | 189 |
| ИТОГО: |  | 1,23 |  | 526,5 |

Для предприятий 1 кВт/ч = 2,89 (руб.).

2,89\*526,5=1521,6 (рублей) в месяц.

Заработная плата программиста составляет 30000 (рублей).

Расчет ежемесячных расходов на поддержку сайта.

С 2010 года единый социальный налог будет отменен, а действующие налогоплательщики будут уплачивать страховые взносы в ПФР, ФСС, федеральные и государственные ФОМСы в соответствии с законом № 212-ФЗ от 24 июля 2009 года.

Таблица 5.2 - Ставки налогов и их распределение

|  |  |
| --- | --- |
| Направление отчисления | 1-ый предел (0-280 000 руб. включительно) |
| ПФР | 14% |
| ТФОМС | 2% |
| ФФОМС | 1,1% |
| ФСС | 2,9% |
| ИТОГО: | 26% |

Таким образом, для типичной организации максимальная налоговая нагрузканаодногосотрудникасоставляет26%.

Таблица 5.3 - Расчет ежемесячных затрат на содержание сайта

|  |  |  |
| --- | --- | --- |
| Наименование | Сумма, (руб). | НО, (руб). |
| Зарплата программиста | 30000 | 7800 |
| Зарплата курьера | 2500 | 650 |
| Транспортные расходы курьера | 1700 |  |
| Электроэнергия | 1521,6 |  |
| Хостинг | 300 |  |
| Интернет | 400 |  |
| Прочие расходы | 250 |  |
| Итого: | 36471,6 | 8450 |
| ИТОГО: | 44921,6 | |

Где НО - налоговый вычет, Rпост = 44921,60 (руб.) - постоянные ежемесячные расходы. Поскольку помещение и оборудование уже есть, рассчитаем годовую сумму амортизации.

Годовая сумма амортизации рассчитывается по следующей формуле

где *Ф* - первоначальная стоимость основных фондов по видам, руб.; *NA* - норма амортизации по видам основных фондов, в %.

Годовую сумму амортизационных отчислений отразим в таблице 5.4.

Таблица 5.4 - Расчет годовой суммы амортизационных отчислений.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Элементы основных фондов | Кол-во | Стои-мость, руб. | Сумма руб. | Норма амортиза-ции, % | Амортизацион - ные отчисления, руб. |
| Компьютер | 1 | 28000 | 28000 | 20% | 5600 |
| Сплит система | 1 | 19000 | 19000 | 20% | 3800 |
| Помещение | 13,6м2 | 5400 | 73440 | 3% | 2203,2 |
| ИТОГО: | | | | | 11603,2 |

Таким образом, годовая стоимость амортизации составляет 11603,2 (руб.). Исходя из того, что трудоемкость создания информационной системы составляет 30 дней, амортизация оборудования за этот период рассчитывается по формуле:



Теперь рассчитаем сумму амортизации для указанных групп оборудования с учетом количества календарных дней разработки площадки по формуле:

А = =953,7 (руб).

Заработная плата программиста составляет 30000 рублей. Таким образом, заработная плата, включенная в стоимость программы с учетом 30 дней работы программы, выглядит следующим образом:



где ЗПпр - заработная плата в месяц программиста, руб.;

Тфакт - число календарных дней на разработку интернет - магазина;

Д - число дней в периоде (месяц).

*ЗПпр =*  = 45000 (руб).

Отчисления на социальное страхование составят:

НО = ЗПпр \* 26%

НО = 11700 руб.

Таблица 5.5 - Расчет ежемесячных материальных затрат.

|  |  |
| --- | --- |
| Наименование | Сумма, руб/мес. |
| Электроэнергия | 1521,6 |
| Хостинг | 300 |
| Интернет | 400 |
| Прочие расходы | 250 |
| Итого: | 2471,6 |

Зм= 2471,6 (руб /мес).

Таким образом, стоимость за период разработки программного продукта рассчитывается по следующей формуле:

*Зпр* = 

где Зм - ежемесячные затраты, руб.; Тфакт - число календарных дней на разработку сайта; Д - число дней в периоде (месяц).

= 3370,4 (руб).

Рассчитаем себестоимость программного продукта по формуле:

Сст = Зпр + ЗПпр + НО + А

где, Сст - себестоимость разработки программы

Сст = 3370,4+45000+11700+953,7 = 61024,1 (руб).

Данная стоимость является оценочной и не учитывает детали, которые не оказывают существенного влияния на общую стоимость: Сст ≈ 61024 (руб.). Исходя из обычной нормы прибыли в 20%, мы можем определить цену разработанной нами программы:



где Сст - себестоимость разработки программы;

R - планируемый уровень рентабельности.

 (руб).

Поскольку площадка и оборудование уже предоставлены заказчику, стоимость установки программного продукта составит 62244,48 рубля. Предполагается, что сайт выйдет на самоокупаемость при затратах 44921,60 руб. в месяц, что увеличит оборот компании на 15 % и прибыль на 6,5 %.

П = (Тобн + Тобн\* Тоб%) \*П%

где П - предполагаемая прибыль; Тобн - базовый вариант товарооборота; Тоб% - предполагаемый процент прироста товарооборота; П% - предполагаемый прирост прибыли.

П1= (200000+200000\*15/100) \*6,5/100 = 14950 (руб).

Таким образом, полная окупаемость сайта, включая затраты на внедрение, произойдет через два месяца после того, как заказы компании увеличатся на 20 %, а прибыль - на 8 %.

П2 = (200000+200000\*20/100) \*8/100 = 19200 (руб).

Таким образом, расчеты подтверждают предположение о том, что прибыль от использования участка ожидается с третьего месяца эксплуатации.

# Выводы

Проанализировав сайт GlassArt, можно сделать выводы об экономической целесообразности и окупаемости сайта. Потенциальным клиентам предоставляется возможность ознакомиться с компанией и предлагаемым ассортиментом продукции.

ПП призван обеспечить наиболее эффективное позиционирование продукции компании в интернете, укрепить положительный имидж и узнаваемость бренда и, конечно, увеличить продажи. По оценкам специалистов, сайт самоокупается уже через три месяца, а его прибыль увеличится на 6,5 %.

# 6. Безопасность жизнедеятельности

С развитием науки и техники важную роль играет безопасное выполнение людьми своих трудовых обязанностей. В связи с этим создается и развивается наука об охране труда и жизнедеятельности человека.

Безопасность жизнедеятельности - это комплекс мероприятий, направленных на обеспечение безопасности человека в окружающей среде, сохранение здоровья, разработку способов и средств защиты путем снижения воздействия вредных и опасных факторов до допустимого уровня и разработки средств ограничения ущерба при ликвидации последствий чрезвычайных ситуаций.

Охрана здоровья работников, обеспечение безопасных условий труда, ликвидация профессиональных заболеваний и несчастных случаев на производстве - одна из главных забот человеческого общества. Обращается внимание на необходимость широкого применения прогрессивных форм научной организации труда, сведения к минимуму ручного, малоквалифицированного труда и создания условий, исключающих профессиональные заболевания и несчастные случаи на производстве.

Рабочие места должны быть оборудованы средствами защиты от возможного воздействия опасных и вредных производственных факторов. Уровень этих факторов не должен превышать предельных значений, установленных правовыми, техническими и санитарно-техническими нормами. Согласно этим нормам, на рабочих местах должны быть созданы такие условия труда, при которых воздействие опасных и вредных факторов либо полностью исключено, либо находится в допустимых пределах.

Данный раздел дипломного проекта посвящен рассмотрению следующих вопросов:

- Определение оптимальных условий труда для инженеров-программистов;

- Расчет уровней освещенности;

- Расчет уровня шума;

# 6.1 Описание условий труда программиста

Сегодня компьютерная техника широко используется во всех сферах человеческой деятельности. При работе с компьютерами люди подвергаются воздействию многих опасных и вредных производственных факторов. К ним относятся электромагнитные поля (радиочастотный диапазон: ВЧ, УВЧ, СВЧ), инфракрасное излучение, ионизирующее излучение, шум, вибрация и статическое электричество.

Работа с компьютером характеризуется высоким умственным напряжением и нервно-эмоциональной нагрузкой на оператора, высокой интенсивностью зрительной работы и значительной нагрузкой на мышцы рук при работе с клавиатурой компьютера. Большое значение имеет рациональное проектирование и расположение элементов рабочего места, что важно для поддержания оптимальной рабочей позы оператора.

При работе на компьютере необходимо соблюдать правильный режим труда и отдыха. В противном случае работники жалуются на неудовлетворенность работой, головные боли, раздражительность, нарушения сна, усталость, болезненные ощущения в глазах, спине, шее и руках, большую нагрузку на органы зрения.

# 6.2 Требования к производственным помещениям

# 6.2.1 Цветовое оформление и коэффициенты отражения

Цветовые тона помещений и мебели должны способствовать созданию благоприятных условий для зрительного восприятия и хорошего настроения.

Источники света, дающие отражения на поверхности экрана, такие как лампы и окна, значительно ухудшают точность изображения и вызывают физиологические помехи. Отражения, в том числе от вторичных источников света, должны быть сведены к минимуму. Для защиты окон от избыточного света можно использовать шторы или экраны.

В помещениях, где установлены компьютеры, необходимо обеспечить следующие уровни отражения: потолок: 60-70 %; стены: 40-50 %; пол: около 30 %. Другие поверхности и рабочая мебель: 30-40%.

# 6.2.2 Освещение

Правильно спроектированное и реализованное промышленное освещение улучшает визуальные условия труда, снижает утомляемость, повышает производительность труда, благотворно влияет на рабочую среду, оказывает положительное психологическое воздействие на работников, повышает безопасность труда и снижает травматизм.

Недостаточное освещение вызывает зрительное утомление, снижает бдительность и приводит к развитию преждевременной усталости. Слишком яркое освещение ослепляет, раздражает и вызывает усталость глаз. Неправильно направленный свет на рабочем месте может вызвать резкие тени, блики и дезориентацию работников. Все эти причины могут привести к несчастным случаям и профессиональным заболеваниям, поэтому так важен правильный расчет освещения.

Существует три типа освещения: естественное, искусственное и комбинированное (сочетание естественного и искусственного освещения).

Естественное освещение - освещение дневным светом, проникающим через проемы в наружных стенах. Естественное освещение характеризуется значительными колебаниями в зависимости от времени суток, времени года, характера местности и различных других факторов.

Искусственное освещение используется в темное или дневное время суток, когда невозможно обеспечить стандартизированные значения коэффициентов естественного дневного освещения (пасмурная погода, короткий световой день). Освещение, при котором недостаточность естественного освещения по отношению к нормативному значению компенсируется искусственным освещением, называется комбинированным.

Искусственное освещение можно разделить на рабочее, аварийное, эвакуационное и охранное. Рабочее освещение включает в себя общее и комбинированное освещение. При общем освещении лампы располагаются равномерно по всей верхней части помещения или в соответствии с расположением оборудования. Комбинированное освещение - общее освещение плюс местное освещение.

Согласно СНиП 26-05-95, в помещениях вычислительных центров должна применяться комбинированная система освещения.

Для высокоточных зрительных работ (минимальный размер объектов от 0,3 до 0,5 мм) коэффициент естественной освещенности (КЕО) должен быть менее 1,5%; для среднеточных зрительных работ (минимальный размер объектов от 0,5 до 1,0 мм) КЕО должен быть менее 1,0%. В качестве источников света для искусственного освещения обычно используются люминесцентные лампы типа ЛБ или ДРЛ, которые должны быть объединены попарно в светильники и равномерно распределены по рабочей поверхности.

Требования к освещенности в помещении, где находится компьютер, следующие: для высокоточной зрительной работы общая освещенность составляет 300 лк, а комбинированная - 750 лк; для среднеточной работы аналогичные требования составляют 200 лк и 300 лк соответственно.

Кроме того, все поле зрения должно быть хорошо и равномерно освещено. Это основное требование гигиены. Другими словами, освещенность помещения и экрана компьютера должна быть примерно одинаковой. Это связано с тем, что яркий свет в периферийном поле зрения значительно увеличивает нагрузку на глаза, что, в свою очередь, быстро ухудшает зрение.

# 6.2.3 Параметры микроклимата

Хотя параметры микроклимата могут варьироваться в широких пределах, необходимым условием жизнедеятельности человека является терморегуляция, т.е. способность организма регулировать теплообмен с окружающей средой для поддержания постоянной температуры тела. Принцип регулирования микроклимата заключается в создании оптимальных условий для теплообмена между организмом человека и окружающей средой.

Компьютерное оборудование является одним из основных источников тепловыделения, вызывая повышение температуры в помещении и снижение относительной влажности воздуха. В помещениях, где установлены компьютеры, должны соблюдаться особые параметры микроклимата. Гигиенический норматив СанПиН 2.2.4.548-96 «Гигиена труда и микроклимат помещений» устанавливает значения параметров микроклимата, создающих комфортные условия. В таблице 6.1 приведены нормы, установленные в зависимости от времени года, характера трудового процесса и характера производственного помещения.

Объем помещений, в которых размещаются работники вычислительных центров, должен быть не менее 19,5 м3/чел. с учетом максимального числа одновременно работающих в смену. Нормы подачи свежего воздуха в помещения с компьютерами приведены в табл. 6.2.

Таблица6.1 - Параметры микроклимата для помещений, где установлены компьютеры

|  |  |  |
| --- | --- | --- |
| Период года | Параметр микроклимата | Величина |
| Холодный | Температура воздуха в помещении  Относительная влажность  Скорость движения воздуха | 22…24°С  40…60%  до 0,1м/с |
| Теплый | Температура воздуха в помещении  Относительная влажность  Скорость движения воздуха | 23…25°С  40…60%  0,1…0,2м/с |

Таблица 6.2 ***-*** Нормы подачи свежего воздуха в помещения, где расположены компьютеры

|  |  |
| --- | --- |
| Характеристика помещения | Объемный расход подаваемого в помещение свежего воздуха, м3 /на одного человека в час |
| Объем до 20м3 на человека  20…40м3 на человека  Более 40м3 на человека | Не менее 30  Не менее 20  Естественная вентиляция |

Для обеспечения комфортных условий используются как организационные (рациональная организация труда в зависимости от времени года и дня недели, чередование труда и отдыха), так и технические меры (системы вентиляции, кондиционирования и отопления). В данном случае комфортные условия труда специалистов обеспечивает кондиционирование воздуха. Кондиционер - это автоматическая система вентиляции, которая поддерживает постоянный микроклимат в помещении.

# 6.2.4 Уровень шума и вибрации

Шум может оказывать вредное воздействие на организм человека и ухудшать условия труда. Люди, работающие в шумной обстановке в течение длительного времени, испытывают раздражительность, головные боли, головокружение, потерю памяти, повышенную утомляемость, потерю аппетита и боли в ушах. Такое нарушение работы многих органов и систем организма может привести к стрессовым негативным изменениям в эмоциональном состоянии человека. Под воздействием шума у человека снижается концентрация внимания, нарушаются физиологические функции, повышенные энергозатраты и нервно-психическое напряжение, приводят к утомлению, нарушается передача речи. Все это снижает работоспособность человека, производительность, качество работы и безопасность. Длительное воздействие шума высокой интенсивности [>80 дБ (А)] может привести к частичной или полной потере слуха.

Уровень шума не должен превышать 50 дБА на рабочих местах математиков, программистов и видеооператоров и 65 дБА в помещениях для обработки компьютерной информации. Для снижения уровня шума стены и потолки компьютерных залов покрывают звукопоглощающими материалами. Уровень вибрации в компьютерных центрах можно снизить, установив оборудование на специальные антивибрационные платформы.

# 6.2.5 Электромагнитное и ионизирующее излучения

В таблице 6.3 приведены допуски на параметры неионизирующего электромагнитного излучения для компьютерных мониторов.

Максимальное значение рентгеновского излучения на рабочем месте оператора компьютера обычно не превышает 10 мкбэр/ч, а интенсивность ультрафиолетового и инфракрасного излучения от экрана монитора не превышает 10. 100 мВт/м2 и менее.

Таблица 6.3 - Допустимые значения параметров неионизирующих электромагнитных излучений (согласно СанПиН 2.2.542-96).

|  |  |
| --- | --- |
| Наименование параметра | Допустимые значения |
| Напряженность электрической составляющей электромагнитного  поля на расстоянии 50см от поверхности видеомонитора | 10В/м |
| Напряженность магнитной составляющей электромагнитного  поля на расстоянии 50см от поверхности видеомонитора | 0,3А/м |
| Напряженность электростатического поля не должна превышать:  для взрослых пользователей  для детей дошкольных учреждений и учащихся  средних специальных и высших учебных заведений | 20кВ/м  15кВ/м |

Для снижения этого излучения используются мониторы с более низким уровнем излучения (MPR-II, TCO-92, TCO-99), устанавливаются защитные экраны и соблюдается регламентированная система труда и отдыха.

# 

# 6.2.6 Эргономические требования к рабочему месту

Проектирование рабочих мест с видеотерминалами является одним из ключевых вопросов эргономического проектирования в области компьютерной техники.

Рабочее место и расположение всех его элементов должны отвечать антропометрическим, физиологическим и психологическим требованиям. Характер работы также имеет решающее значение. В частности, при обустройстве рабочего места программиста должны соблюдаться следующие основные требования: оптимальное расположение оборудования на рабочем месте и достаточное рабочее пространство для всех необходимых движений и перемещений.

В частности, эргономические аспекты проектирования рабочего места видеотерминала требуют соблюдения следующих требований

- Высота верстака;

- пространство для ног;

- Требования к расположению файлов на рабочем месте (например, наличие и размер полок для файлов, различные возможные варианты расположения файлов, расстояние от глаз пользователя до экрана, файлов и клавиатуры);

- Характеристики рабочего кресла, требования к поверхности стола, возможности регулировки элементов рабочей зоны.

Основными элементами рабочего места программиста являются стол и стул. Основная рабочая поза - сидячая.

Сидячая рабочая поза минимизирует усталость программиста. Правильное планирование рабочего места позволяет упорядоченно и последовательно разместить предметы, инструменты труда и документы. Предметы, необходимые для частой работы, должны располагаться в рабочей зоне в пределах досягаемости.

Зона упражнений - пространство в рабочей зоне, где происходит движение человека.

Зона максимального разгибания руки является частью зоны движения в рабочей зоне и ограничена дугой, которую вытягивает рука при максимальном разгибании при движении в плечевом суставе.

Как показано на рисунке 6.3.1, оптимальная зона - это зона движения в рабочей зоне, определяемая дугой, проведенной, когда предплечье находится в точке опоры локтя и движется в локтевом суставе, а плечо относительно неподвижно.



а *-* зона максимальной досягаемости;

б - зона досягаемости пальцев при вытянутой руке;

в - зона легкой досягаемости ладони;

г - оптимальное пространство для грубой ручной работы;

д *-* оптимальное пространство для тонкой ручной работы

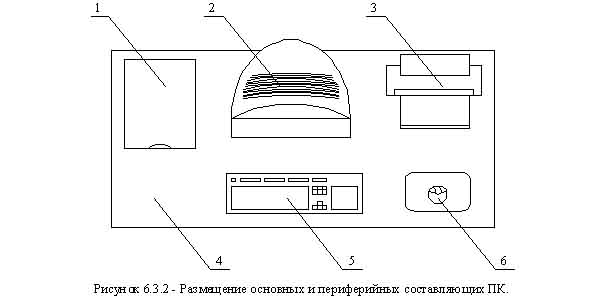
Рисунок. 6.3.1 - Зоны досягаемости рук в горизонтальной плоскости.



Оптимальное размещение рабочих элементов и документов в пределах досягаемости:

* Дисплей расположен в зоне «а» (в центре);
* Системный блок расположен в нише приставного стола;
* Клавиатура расположена в зоне «г/д»;
* Мышь расположена справа от зоны «в»;
* Сканер расположен в зоне «a/б» (слева);
* Принтер находится в зоне «a» (справа).

Документы, необходимые для работы, находятся в зоне «в»; редко используемые документы - в ящике стола.



На рисунке 6.3.2 приведен пример основных и периферийных компонентов ПК на рабочем столе программиста: 1 - сканер, 2 - монитор, 3 - принтер, 4 - рабочий стол, 5 - клавиатура, 6 - манипулятор «мышь».

Для комфортной работы верстак должен отвечать следующим требованиям

- Высота стола должна быть подобрана таким образом, чтобы пользователь мог свободно сидеть в удобной позе;

- нижняя часть стола должна быть спроектирована таким образом, чтобы программист мог удобно сидеть, не поднимая ног; и

- поверхность стола не должна мешать прямой видимости программиста;

- Поверхность стола не должна мешать обзору программиста. Стол должен иметь выдвижные ящики (не менее трех для файлов, списков и канцелярских принадлежностей).

- Рекомендуемая высота рабочего места должна составлять 680-760 мм. Высота поверхности для размещения клавиатуры должна составлять около 650 мм.

Характеристики рабочего кресла очень важны. Поэтому рекомендуемая высота кресла от пола составляет 420-550 мм. Сиденье должно быть мягким, с закругленным передним краем и регулируемым углом наклона спинки.

Кресло должно быть сконструировано таким образом, чтобы документы можно было разместить в разных положениях, например, рядом с видеотерминалом, между монитором и клавиатурой. Если качество изображения на видеотерминале низкое (например, заметное мерцание), расстояние между глазами и экраном (около 700 мм) должно быть больше, чем расстояние между глазами и документом (300-450 мм). В целом, чем выше качество изображения видеотерминала, тем больше должно быть одинаковое расстояние от глаз пользователя до экрана, документа и клавиатуры.

Положение экрана определяется следующим образом:

- Расстояние для чтения (0,6. .0,7 м).

- Угол чтения, от горизонтали 20о, т.е. направление от центра экрана, при этом экран должен быть перпендикулярен этому направлению.

Также должна быть предусмотрена возможность регулировки экрана:

- Высота +3 см;

- Наклон относительно вертикального направления, от -10о до +20 о;

- Ориентация влево и вправо.

Правильная рабочая поза пользователя также очень важна. Неудобная рабочая поза может вызвать боль в мышцах, суставах и сухожилиях; рабочая поза конечного пользователя видеосистемы выглядит следующим образом:

- Голова не должна быть наклонена более чем на 20о,

- Плечи должны быть расслаблены,

- Угол наклона локтей должен быть 80о.... ...100о, а предплечья и кисти должны быть ровными.

- Предплечья и кисти должны быть горизонтальными.

Причины неправильной осанки пользователя следующие: отсутствие хороших держателей для документов, слишком высокие клавиатуры, слишком низкие документы, некуда положить руки, не хватает места для ног.

Для преодоления этих недостатков были сделаны некоторые общие предложения: не помешала бы мобильная клавиатура; необходимо предусмотреть специальные устройства для регулировки высоты рабочего стола, клавиатуры и экрана, а также подставки для рук. Размер текста, плотность выравнивания, контрастность и соотношение яркости между текстом и фоном экрана очень важны для эффективной и качественной работы за компьютером. Если расстояние от глаз оператора до экрана дисплея составляет 60-80 сантиметров, то высота логотипа должна быть не менее 3 миллиметров, оптимальное соотношение ширины и высоты логотипа - 3:4, а расстояние между логотипами - 15-20% от высоты. Соотношение яркости фона экрана и логотипа должно составлять от 1:2 до 1:15.

При работе за компьютером врачи рекомендуют располагать монитор на расстоянии 50-60 см от глаз. Специалисты также считают, что верхняя часть видеомонитора должна находиться на уровне глаз или чуть ниже. Когда человек смотрит прямо перед собой, его глаза открываются шире, чем, когда они опущены вниз. Это может значительно увеличить площадь обзора и обезвожить глаза. Кроме того, если экран установлен высоко, а глаза широко открыты, нарушается функция моргания. Это означает, что глаза не могут полностью закрыться и омыться слезами, чтобы сохранить достаточное количество влаги, что может привести к быстрой утомляемости глаз. Создание благоприятных условий труда на производстве и соответствующее эстетическое оформление рабочих мест имеют большое значение для облегчения труда, повышения привлекательности рабочих мест и положительного влияния на производительность труда.

# 6.3 Пожарная безопасность

Пожарная безопасность на оборонных объектах.

1. Каждый объект обороны должен иметь систему пожарной безопасности.

2. Целью системы пожарной безопасности защищаемого объекта является предотвращение пожаров, обеспечение безопасности людей и защита имущества в случае пожара.

3. Система пожарной безопасности охраняемого объекта состоит из системы противопожарной защиты, системы пожаротушения и комплекса организационно-технических мероприятий по обеспечению пожарной безопасности.

4. Система обеспечения пожарной безопасности объекта защиты должна включать в себя комплекс мероприятий, направленных на исключение возможности превышения установленных настоящим Федеральным законом допустимых значений пожарного риска и на предотвращение риска причинения вреда третьим лицам в результате пожара.

Пожар может возникнуть в любом помещении. Для тушения пожара и обеспечения безопасности работников предприятия должны быть приняты определенные меры пожаротушения.

Стационарные устройства пожаротушения позволяют потушить пожар на ранней стадии без участия человека. Они устанавливаются в зданиях и сооружениях, а также используются для защиты наружного технического оборудования. В зависимости от используемого огнетушащего вещества они делятся на водяные, пенные, газовые, порошковые и паровые. Стационарные системы пожаротушения могут управляться автоматически или вручную с помощью пульта дистанционного управления.

# 6.4 Расчет уровня освещенности

Расчет освещенности рабочего места можно свести к выбору системы освещения и определению необходимого количества, типа и расположения ламп. Исходя из этого, можно рассчитать параметры искусственного освещения.

Искусственное освещение обычно обеспечивается двумя типами электрических источников света: лампами накаливания и люминесцентными лампами. Люминесцентные лампы используются в данном случае потому, что они имеют много преимуществ перед лампами накаливания:

- спектр ближе к дневному или естественному свету;

- высокий КПД (в 1,5-2 раза выше, чем у ламп накаливания);

- высокая светоотдача (в 3-4 раза выше, чем у ламп накаливания);

- длительный срок эксплуатации.

Расчет освещения основан на помещении площадью 13,6 кв. м, шириной 4,2 м, длиной 3,23 м и высотой 3,2 м. Используйте метод светового потока.

Чтобы определить количество светильников, используйте формулу светового потока, падающего на поверхность:



где *F* - рассчитываемый световой поток, Лм;

*Е* - нормированная минимальная освещенность, Лк (определяется по таблице). Работу программиста, в соответствии с этой таблицей, можно отнести к разряду точных работ, следовательно, минимальная освещенность будет Е = 300Лк;

*S* - площадь освещаемого помещения (в нашем случае S = 13,6м2);

*Z* - отношение средней освещенности к минимальной (обычно принимается равным 1,1…1,2, пусть Z= 1,1);

*К* - коэффициент запаса, учитывающий уменьшение светового потока лампы в результате загрязнения светильников в процессе эксплуатации (его значение зависит от типа помещения и характера проводимых в нем работ и в нашем случае К = 1,5);

*n* - коэффициент использования, (выражается отношением светового потока, падающего на расчетную поверхность, к суммарному потоку всех ламп и исчисляется в долях единицы; зависит от характеристик светильника, размеров помещения, окраски стен и потолка, характеризуемых коэффициентами отражения от стен (РС) и потолка (РП)), значение коэффициентов РС и РП были указаны выше: РС=40%, РП=60%. Значение n определим по таблице коэффициентов использования различных светильников. Для этого вычислим индекс помещения по формуле (СНиП 23-5-95):



где *S* - площадь помещения, S = 13,6 м2; *h* - расчетная высота подвеса, учитывая подвесной потолок h = 3 м; *A* - ширина помещения, А = 3,23 м; *В* - длина помещения, В = 4,2 м. Подставив значения получим:



Зная индекс помещения ***I***, по таблице 1 СНиП 23-05-95 находим *n*=0,25

Подставим все значения в формулу для определения светового потока *F*:

 (Лм)

Для освещения выбираем люминесцентные лампы типа ЛБ40-1, световой поток которых F = 4320 Лм

Рассчитаем необходимое количество ламп по формуле:



где *N* - определяемое число ламп;

*F* - световой поток, F = 26928 Лм;

*Fл* - световой поток лампы, Fл = 4320 Лм.

 (шт)

При выборе осветительных приборов используем светильники типа ОД. Каждый светильник комплектуется тремя лампами.

# 6.5 Оценка уровня шума

Одним из нежелательных элементов производственной среды в центре обработки данных является высокий уровень шума, создаваемый печатным оборудованием, кондиционерами и вентиляторами системы охлаждения самих компьютеров.

Для того чтобы определить необходимость и актуальность снижения уровня шума, необходимо понять, каков уровень шума на рабочем месте оператора.

Уровень шума, создаваемый несколькими источниками помех, работающими одновременно, может быть рассчитан по принципу суммы энергии, излучаемой каждым источником:



где Li - уровень звукового давления i-го источника шума;

n - количество источников шума.

Результаты расчетов сравниваются с допустимым уровнем шума на рабочем месте. Если расчетные результаты превышают допустимый уровень шума, требуются специальные меры по снижению шума. К таким мерам относятся покрытие стен и потолков цеха звукопоглощающими материалами, снижение шума у источника шума, правильное расположение оборудования и рациональная организация рабочего места оператора.

В таблице 6.5 приведены уровни звукового давления источников шума на рабочем месте оператора, которые воздействуют на оператора.

Таблица 6.5 - Уровни звукового давления различных источников.

|  |  |
| --- | --- |
| Источник шума | Уровень шума, дБ |
| Жесткий диск | 25 |
| Кулер | 29 |
| Монитор | 7 |
| Клавиатура | 10 |
| Принтер | 63 |
| Сканер | 38 |
| Кондиционер | 36 |

Системный блок на рабочем месте оператора обычно содержит такое оборудование, как жесткие диски, вентиляторы охлаждения компьютера, мониторы, клавиатуры, принтеры и сканеры.

Подставив значения уровня звукового давления для каждой единицы оборудования в уравнение, получим:

L∑=10·lg (102,5+102,9+100,7+101+106,3+103,8 + 103,6) =63,03 (дБ)

Полученное значение не превышает допустимый уровень шума 65 дБ на рабочем месте оператора (ГОСТ 12.1.030). Это значение еще ниже, если учесть, что периферийное оборудование, такое как сканеры и принтеры, не может быть использовано одновременно. Кроме того, поскольку принтер оснащен устройством автоматической подачи, оператору нет необходимости непосредственно присутствовать при работе принтера.

# Выводы

В этом разделе изложены требования к рабочему месту программиста. Необходимо обеспечить комфортные условия для работы. На основе исследовательской литературы по этому вопросу указаны оптимальные размеры рабочих столов, стульев и рабочих поверхностей, выбор систем, расчет оптимального освещения в производственном помещении и расчет уровня шума на рабочем месте. Условия, соответствующие оптимальной конфигурации рабочего места для инженеров и программистов, помогут поддерживать хорошую работоспособность в течение всего дня и повысить производительность труда программистов как количественно, так и качественно.

# Заключение

Темой дипломного проекта является "Разработка веб-сайта по предоставлению услуг по дизайнерскому остеклению для компании GlassArt".

Основной целью дипломного проекта было создание сайта для производственной компании, удобного для пользователя, информативного и не содержащего лишней и избыточной информации.

Итогом дипломного проекта стал дизайн сайта для компании «ГлассАрт». Сайт был разработан с целью наиболее эффективного позиционирования своей продукции в Интернете, создания положительного имиджа, повышения узнаваемости бренда и, конечно же, увеличения продаж.

Для работы с сайтом достаточно иметь компьютер и доступ в Интернет. Никаких специальных знаний в области информационных технологий не требуется.

Сайт написан на языке разметки HTML и языке программирования JavaScript.

HTML - основной стандарт для создания электронных документов и публикации их в Интернете.

JavaScript - объектно-ориентированный язык сценариев и программирования; JavaScript часто используется как встроенный язык для программного доступа к объектам приложения.

Проанализировав работу «GlassArt» по созданию сайта компании, можно сделать выводы об экономической целесообразности сайта и окупаемости инвестиций. Потенциальные клиенты имеют возможность ознакомиться с компанией и ассортиментом предлагаемой продукции.

PP был разработан с целью наиболее эффективного позиционирования продукции компании в Интернете, повышения положительного имиджа и узнаваемости бренда, и, конечно же, увеличения продаж. Предполагается, что сайт выйдет на самоокупаемость уже через три месяца, а прогнозируемый рост прибыли составит 6,5 %.

Во время работы над дипломным проектом использовались такие программы как:

Microsoft Word

Visual Studio Code

Adobe Photoshop

Microsoft Office Picture Manager

# Список использованных источников

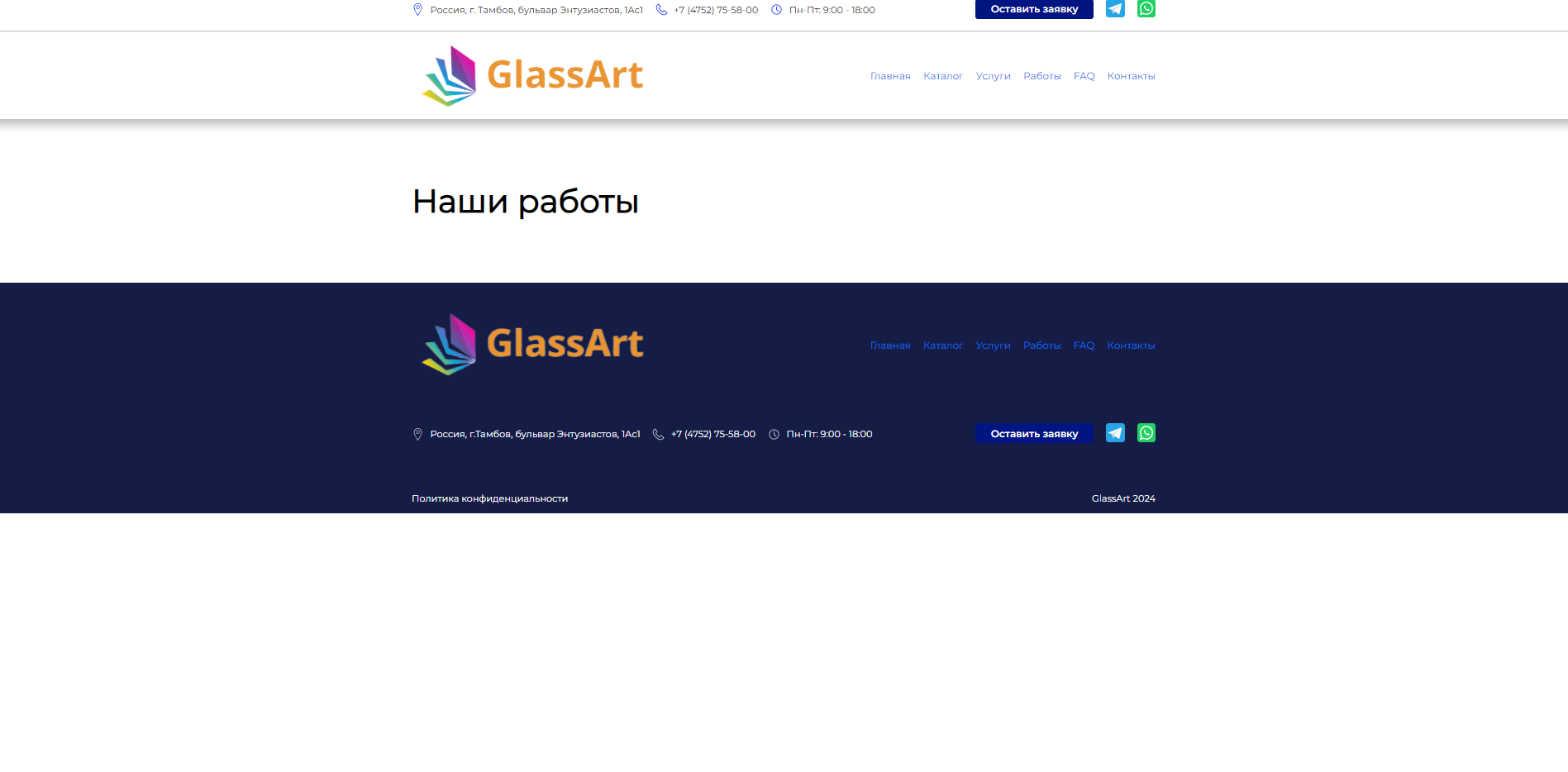
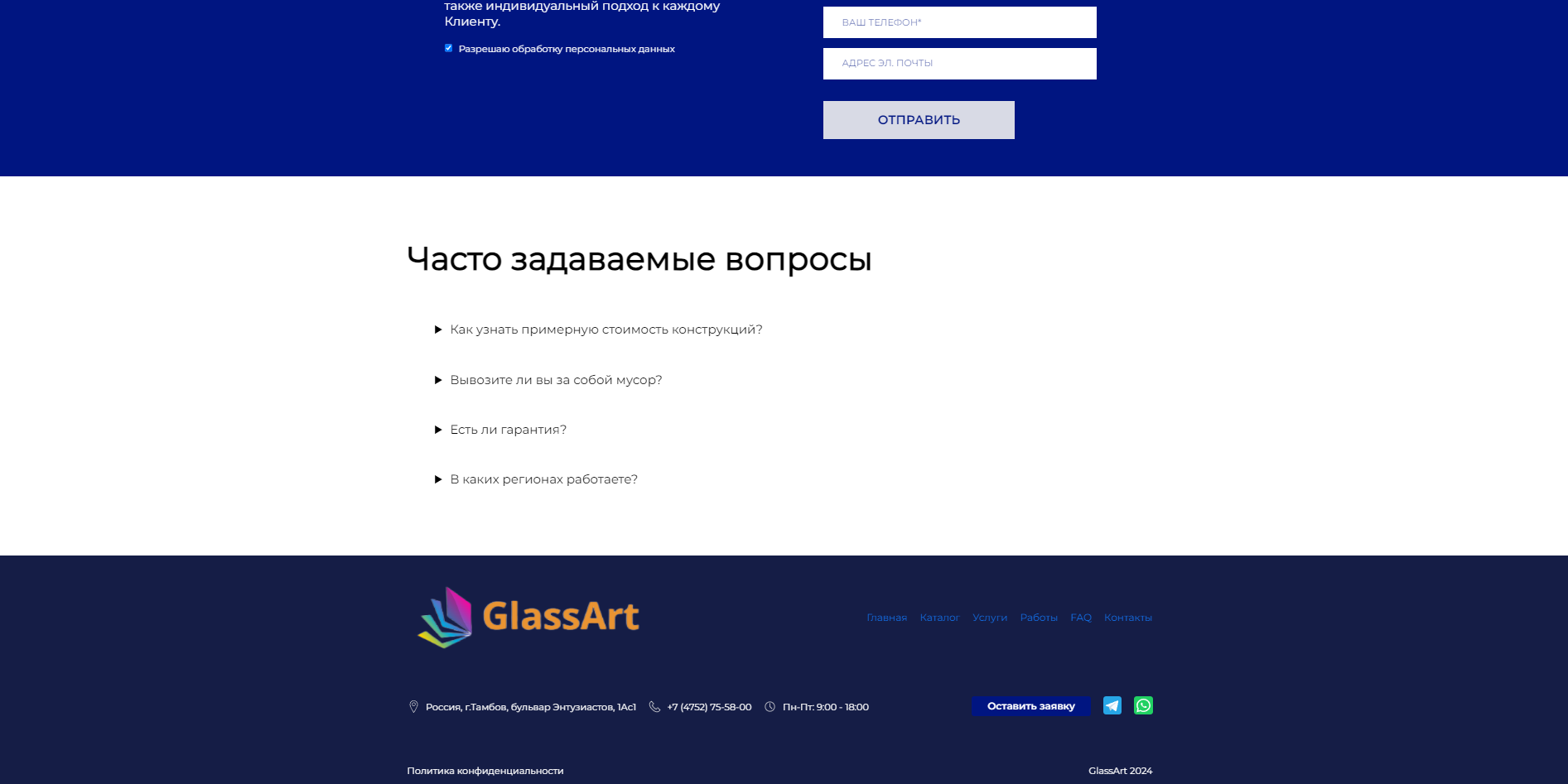
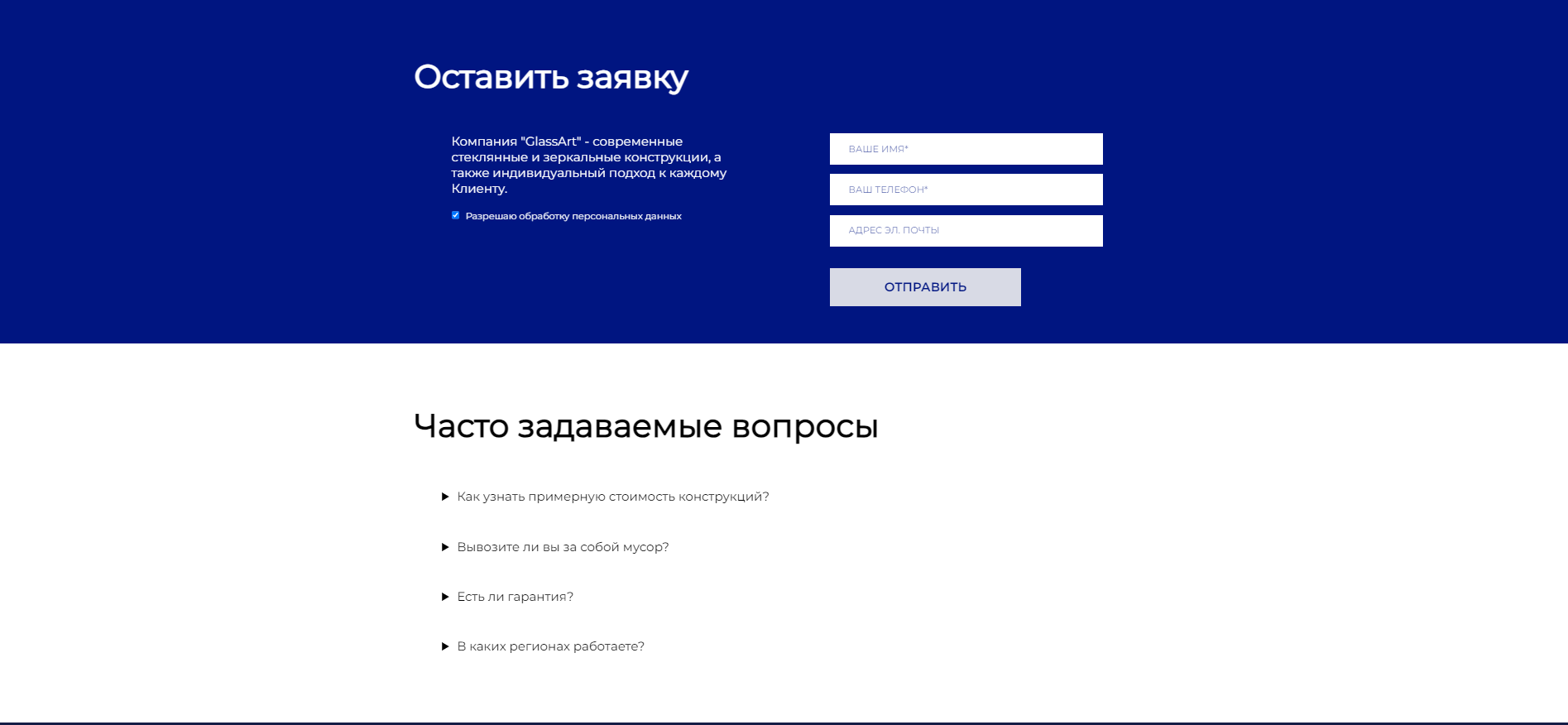
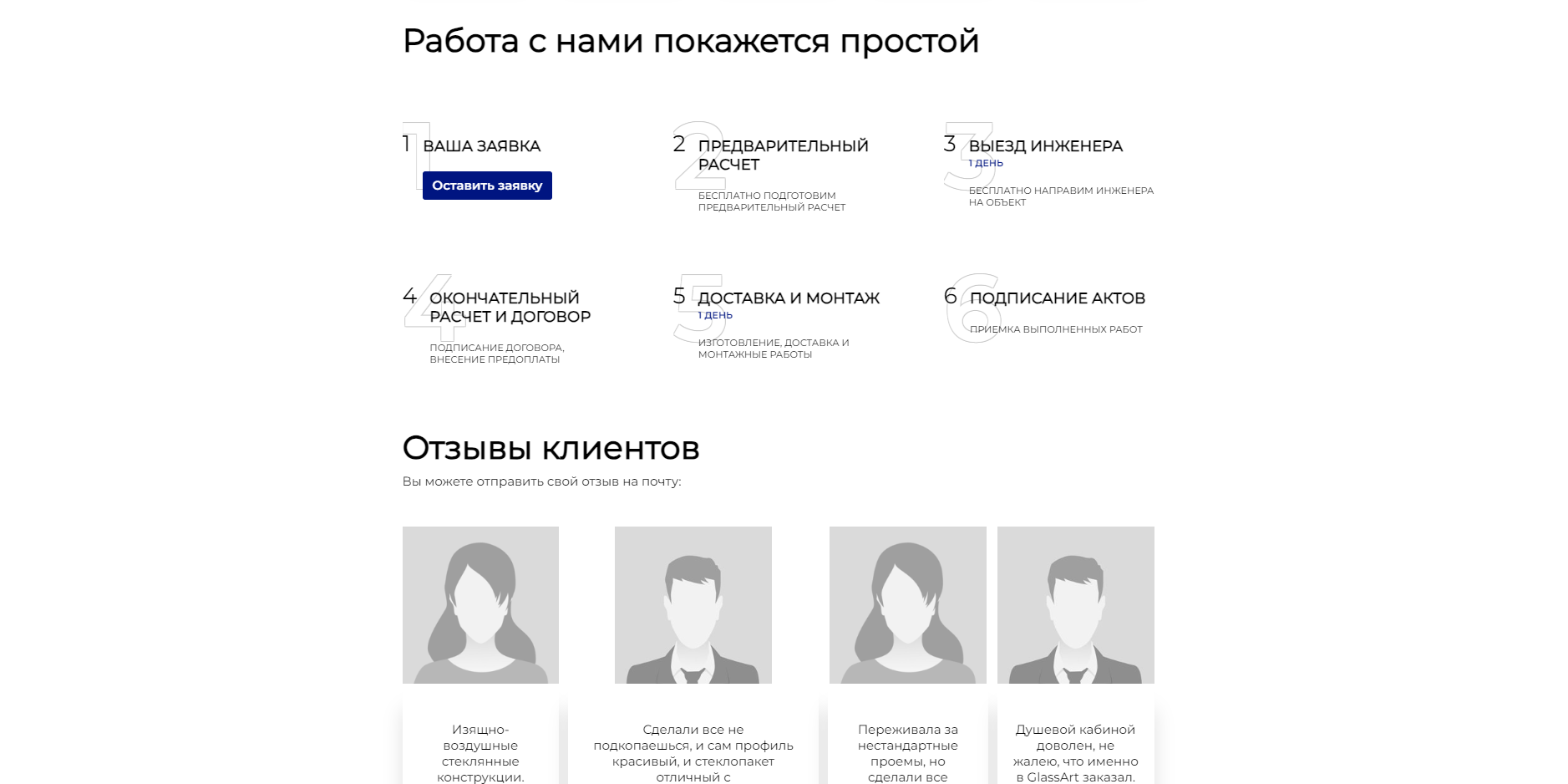
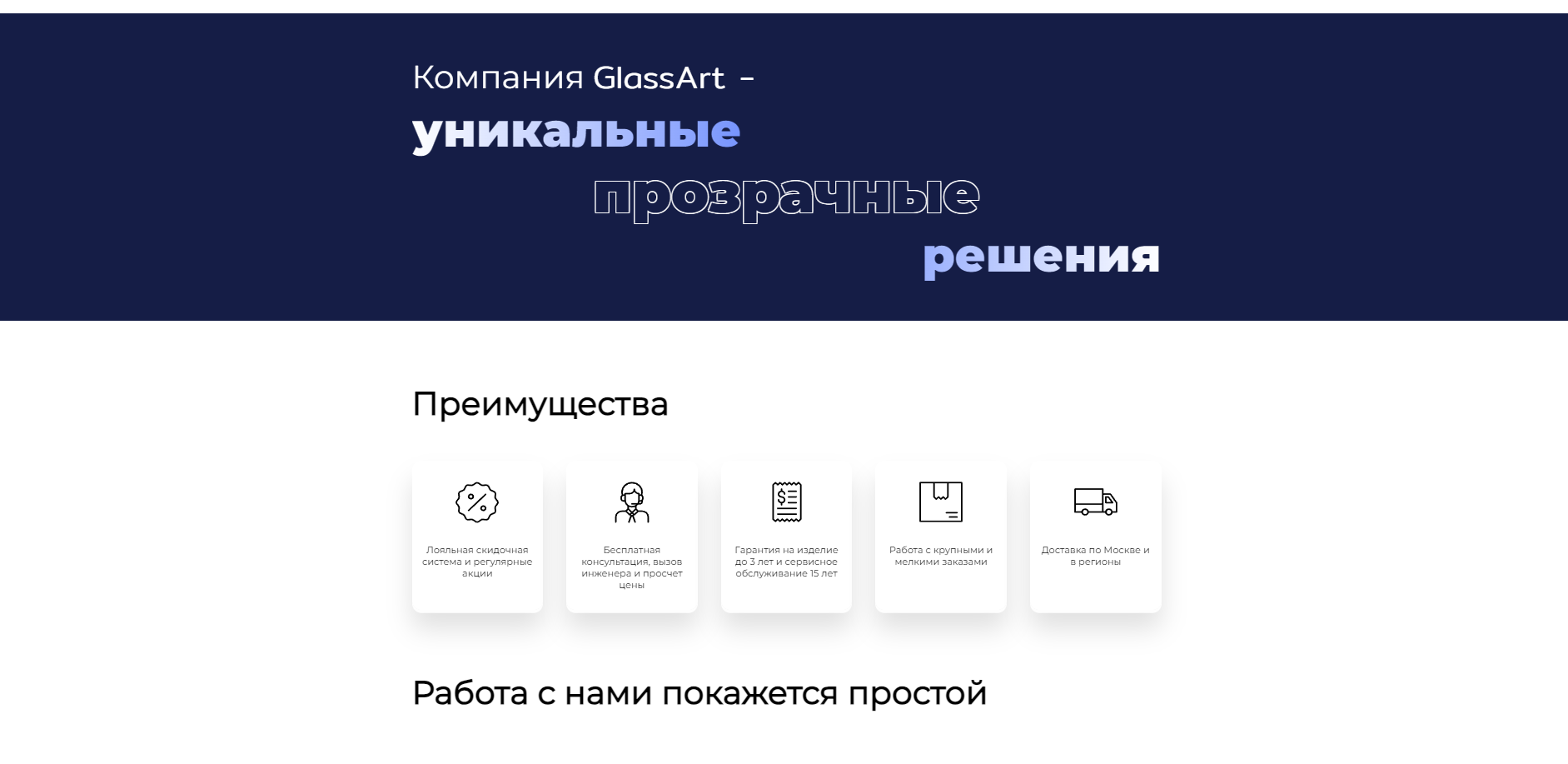
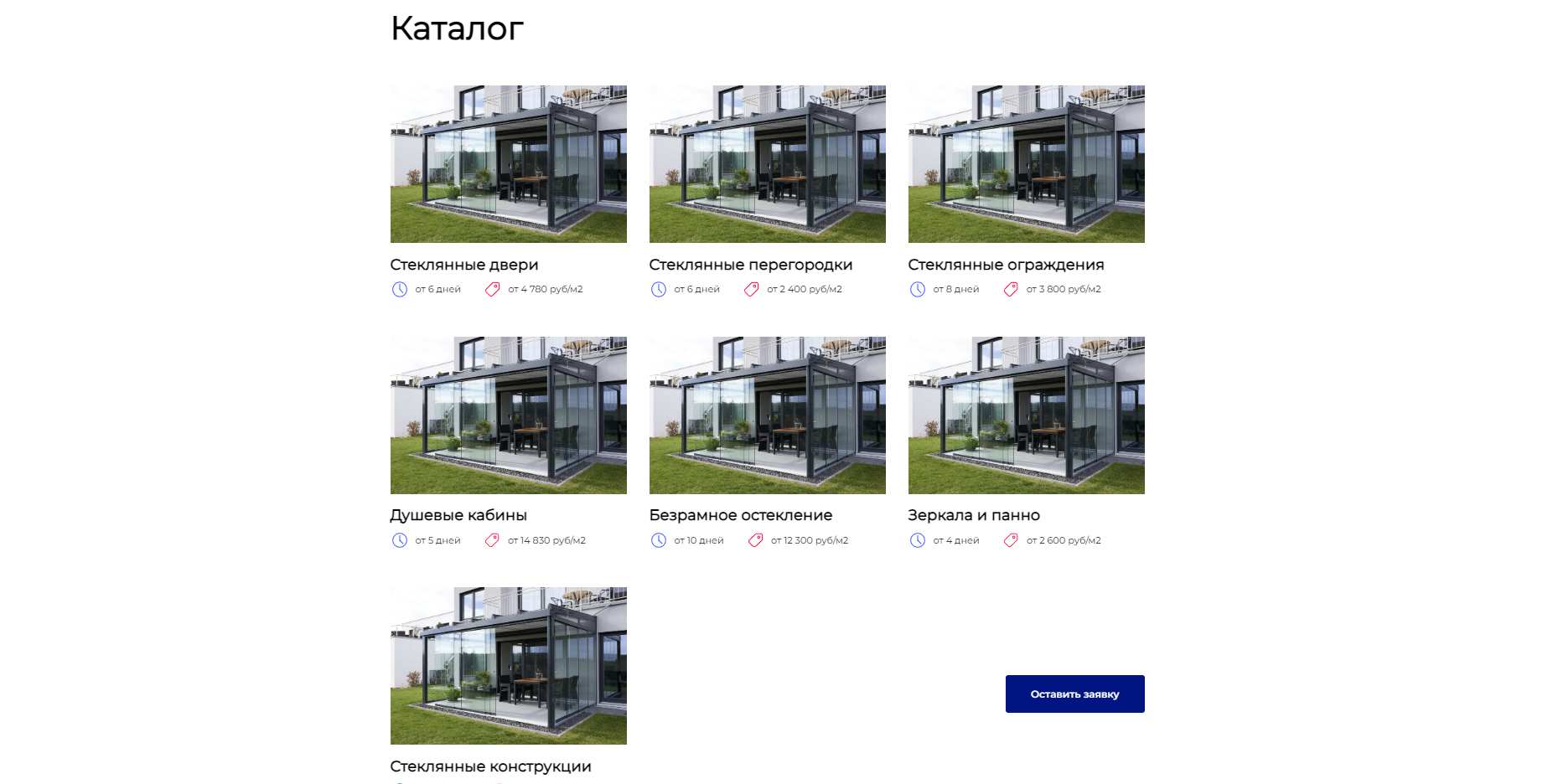
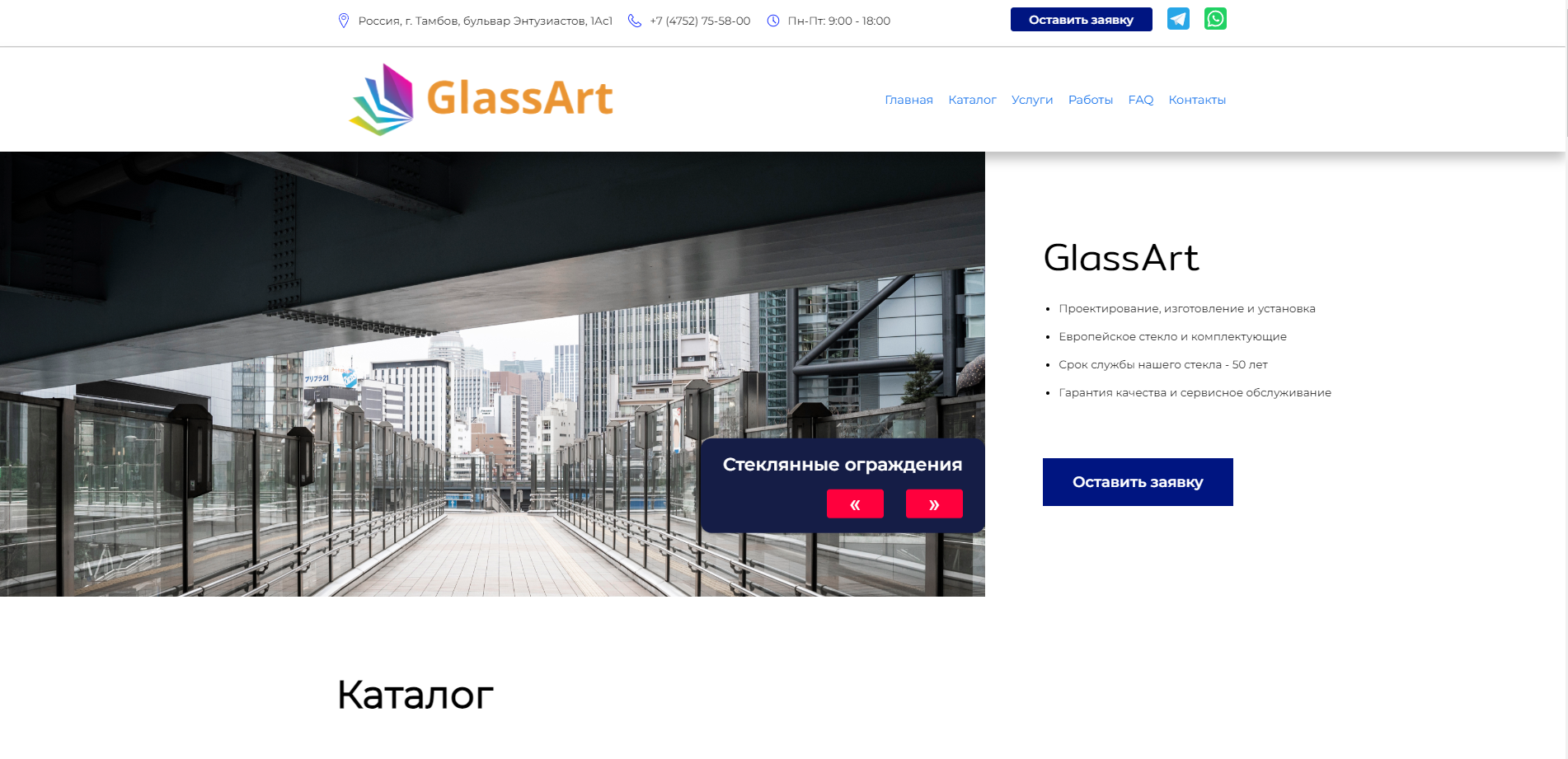
1. Ковалев А., Курдюмов И. Управление проектом по созданию интернет-сайта, М., Альпина Паблишер, 2001г.
2. Организация, планирование и управление предприятием. Составление и расчет параметров сетевой модели выполнения комплекса работ. Методические указания к выполнению курсовой работы. - СПб.: СЗПИ, 1994, - 36 с
3. Дипломное проектирование. Учебное пособие/ Под ред. д. т. н., проф.В.И. Лачина. - Ростов н/Д: изд-во "Феникс", 2003. - 352с. (Серия "Высшее образование".)
4. Нильсен Я. "Веб-дизайн. Книга Якоба Нильсена", "Символ-Плюс". 2003г.512 стр.
5. И. Ашманов, А. Иванов, "Оптимизация и продвижение сайтов в поисковых системах", Издательство "Питер". 2008г.400 стр.
6. Кантор Е.Л. Экономика предприятия: Учебник для вузов. - М.: ПИТЕР, 2003.
7. Центральное бюро нормативов по труду государственного комитета СССР по труду и социальным вопросам. Типовые нормы времени на программирование задач для ЭВМ. М.: Издательство "Экономика", 1989г.
8. Безопасность жизнедеятельности: Учебник/ Под ред. проф. Э.А. Арустамова. - 4-е изд., перераб. и доп. - М.: Издательско-торговая корпорация "Дашков и К0", 2002.
9. Федеральный закон Российской Федерации от 22 июля 2008г. № 123-ФЗ "Технический регламент о требованиях пожарной безопасности"
10. Санитарно-эпидемиологические правила и нормативы (СанПиН 2.2.2/2.4.1340-03). Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. Утверждены главным государственным санитарным врачом РФ от 30.05.03г.

# Приложения

Расположение практической части дипломной работы:

<https://github.com/Oleg-365/Diplom/tree/main/diplom_web>

1. Скриншоты сайта



1. Фрагменты кода разработанного сайта.

Файл About.js

import "./About.scss";

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

const About = () => {

  return (

    <section className="about">

      <div className="about-content">

        <h2>GlassArt</h2>

        <ul>

          <li>Проектирование, изготовление и установка</li>

          <li>Европейское стекло и комплектующие</li>

          <li>Срок службы нашего стекла - 50 лет</li>

          <li>Гарантия качества и сервисное обслуживание</li>

        </ul>

        <Link to="/#feedback">Оставить заявку</Link>

      </div>

    </section>

  );

};

export default About;

Файл About.scss

@import '../variables.modules.scss';

.about {

  margin-left: 10%;

}

.about-content{

  // width: 450px;

  h2 {

    font-family: $font-mitr;

    font-size: $font-size-52;

    font-weight: normal;

    letter-spacing: 0.5;

  }

  ul {

    margin: 10px 3% 100px;

    list-style-type: disc;

  }

  li {

    margin-top: 20px;

    font-family: $font-main;

    font-size: $font-size-15;

  }

  a {

    padding: 20px 40px;

    background-color: $color-buttons;

    color: $color-white;

    text-decoration: none;

    font-family: $font-semibold;

    font-size: $font-size-20;

    cursor: pointer;

    transition: $transition-buttons;

    &:hover {

      transition: $transition-buttons;

      opacity: 0.8;

    }

  }

}

@media screen and (max-width:870px) {

  .about {

    margin: 0;

    display: flex;

    justify-content: center;

    text-align: center;

  }

  .about-content {

    align-content: center;

    h2 {

      font-size: $font-size-52;

    }

  }

}

@media screen and (max-width: 500px) {

  .about-content {

    h2 {

      font-size: $font-size-36;

    }

  }

}

Файл Advantages.js

import './Advantages.scss';

import AdvantagesCards from './Advantages-Cards/Advantages-Cards';

const Advantages = () => {

  return (

    <section class="advantages wrap">

      <h2 class="headers-h2">Преимущества</h2>

      <AdvantagesCards/>

    </section>

  )

}

export default Advantages;

Файл Advantages.scss

@import '../variables.modules.scss';

Файл Advantages-Cards.js

import './Advantages.scss';

import AdvantagesCards from './Advantages-Cards/Advantages-Cards';

const Advantages = () => {

  return (

    <section class="advantages wrap">

      <h2 class="headers-h2">Преимущества</h2>

      <AdvantagesCards/>

    </section>

  )

}

export default Advantages;

Файл Advantages-Cards.scss

@import '../variables.modules.scss';

Файл Advantages-Card.js

import './Advantages-Card.scss';

const AdvantagesCard = ({ cardData }) => {

  const { imgPath, imgAlt, text } = cardData;

  return (

    <div class="advantages-card">

        <img src={imgPath} alt={imgAlt}/>

        <p>{text}</p>

      </div>

  )

}

export default AdvantagesCard;

Файл Advantages-Card.scss

@import '../../../variables.modules.scss';

.advantages-card {

  width: 15%;

  padding: 30px 15px;

  border-radius: 15px;

  box-shadow: 0 5px 0 0 rgba(255, 255, 255, 0.95), 0 30px 40px 0 rgba(0, 0, 0, 0.12);

  background: rgba(255, 255, 255, 0.6);

  display: flex;

  align-items: center;

  flex-direction: column;

  img {

    width: 80px;

    margin-bottom: 30px;

  }

  p{

    font-family: $font-main;

    font-size: $font-size-15;

    text-wrap: wrap;

    text-align: center;

  }

}

@media screen and (max-width: 768px) {

  .advantages-card {

    width: 60%;

  }

}

Файл Before-Header.js

import "./Before-Header.scss";

import BHeaderContact from './Before-Header-Contact/Before-Header-Contact';

import BHeaderButtons from "./Before-Header-Buttons/Before-Header-Buttons";

const BeforeHeader = () => {

  return (

  <div className="wrap before-header">

    <BHeaderContact/>

    <BHeaderButtons/>

  </div>

  )

}

export default BeforeHeader;

Файл Before-Header.scss

.before-header {

  display: flex;

  justify-content: space-between;

}

@media screen and (max-width: 1199px) {

  .before-header {

    flex-direction: column;

    align-items: center;

  }

}

Файл Before-Header-Buttons.js

import "./Before-Header-Buttons.scss";

import TelegramIcon from "assets/media/icons/header/telegram.svg";

import WhatsAppIcon from "assets/media/icons/header/whatsapp.svg";

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

const BHeaderButtons = () => {

  return (

    <div className="bheader-buttons">

      <Link to="/#feedback" className="bheader-button">

        Оставить заявку

      </Link>

      <a href="https://t.me">

        <img src={TelegramIcon} alt="" />

      </a>

      <a href="https://wa.me">

        <img src={WhatsAppIcon} alt="" />

      </a>

    </div>

  );

};

export default BHeaderButtons;

Файл Before-Header-Buttons.scss

@import '../../variables.modules.scss';

.bheader-buttons {

  display: flex;

  gap: 20px;

  align-items: flex-start;

  margin: 17px 0;

  a {

    &:hover {

      opacity: 0.8;

      transition: $transition-buttons;

    }

  }

}

.bheader-button {

  background-color: $color-buttons;

  color: $color-white;

  font-family: $font-semibold;

  padding: 6px 25px;

  border-radius: $border-radius-button;

  transition: $transition-buttons;

  text-decoration: none;

  &:hover {

    opacity: 0.8;

    transition: $transition-buttons;

  }

}

Файл Before-Header-Contact.js

import "./Before-Header-Contact.scss";

import BHeaderCElem from "./Before-Header-Contact-Elem/Before-Header-Contact-Elem";

import IconAddress from "assets/media/icons/header/header-address.svg";

import IconPhone from "assets/media/icons/header/header-phone.svg";

import IconTime from "assets/media/icons/header/header-time.svg";

const BHeaderContact = () => {

  return (

    <div className="bheader-contact">

      <BHeaderCElem

        imgPath={IconAddress}

        text="Россия, г. Тамбов, бульвар Энтузиастов, 1Ас1"

      />

      <BHeaderCElem imgPath={IconPhone} text="+7 (4752) 75-58-00" />

      <BHeaderCElem imgPath={IconTime} text="Пн-Пт: 9:00 - 18:00" />

    </div>

  );

};

export default BHeaderContact;

Файл Before-Header-Contact.scss

.bheader-contact {

  display: flex;

  gap: 20px;

  margin: 17px 0;

}

@media screen and (max-width: 768px) {

  .bheader-contact {

    flex-direction: column;

    align-items: center;

    text-align: center;

  }

}

Файл Before-Header-Contact-Elem.js

import "./Before-Header-Contact-Elem.scss";

const BHeaderCElem = (props) => {

  return (

    <div className="bheader-content-elem">

      <img src={props.imgPath} alt="" />

      <a href="#">{props.text}</a>

    </div>

  );

};

export default BHeaderCElem;

Файл Before-Header-Contact-Elem.scss

@import '../../../variables.modules.scss';

.bheader-content-elem {

  display: flex;

  gap: 10px;

  align-items: center;

  a {

    text-decoration: none;

    color: $color-black;

    font-family: $font-main;

    font-size: $font-size-15;

    text-wrap: wrap;

  }

}

Файл Before-Header-Hr.js

import './Before-Header-Hr.scss';

const BHeaderHr = () => {

  return (

    <div>

        <hr class="bheader-hr"/>

    </div>

  )

}

export default BHeaderHr;

Файл Before-Header-Hr.scss

.bheader-hr {

  border-top: 1px solid #ccc;

}

Файл Catalog.js

import "./Catalog.scss";

import CatalogCards from "./Catalog-Cards/Catalog-Cards";

const Catalog = () => {

  return (

    <section class="catalog wrap" id="catalog">

      <h2 class="headers-h2">Каталог</h2>

      <CatalogCards />

    </section>

  );

};

export default Catalog;

Файл Catalog.scss

@import '../variables.modules.scss';

Файл Catalog-Cards.js

import "./Catalog-Cards.scss";

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

import CatalogCard from "./Catalog-Card/Catalog-Card";

const CatalogCards = () => {

  const cardsData = [

    {

      id: 1,

      title: "Стеклянные двери",

      textTime: "от 6 дней",

      textPrice: "от 4 780 руб/м2",

    },

    {

      id: 2,

      title: "Стеклянные перегородки",

      textTime: "от 6 дней",

      textPrice: "от 2 400 руб/м2",

    },

    {

      id: 3,

      title: "Стеклянные ограждения",

      textTime: "от 8 дней",

      textPrice: "от 3 800 руб/м2",

    },

    {

      id: 4,

      title: "Душевые кабины",

      textTime: "от 5 дней",

      textPrice: "от 14 830 руб/м2",

    },

    {

      id: 5,

      title: "Безрамное остекление",

      textTime: "от 10 дней",

      textPrice: "от 12 300 руб/м2",

    },

    {

      id: 6,

      title: "Зеркала и панно",

      textTime: "от 4 дней",

      textPrice: "от 2 600 руб/м2",

    },

    {

      id: 7,

      title: "Стеклянные конструкции",

      textTime: "от 10 дней",

      textPrice: "от 7 700 руб/м2",

    },

  ];

  return (

    <div class="catalog-cards">

      {cardsData.map((cardData) => (

        <CatalogCard key={cardData.id} cardData={cardData} />

      ))}

      <div class="catalog-card" id="catalog-button">

        <Link to="/#feedback" class="button">

          Оставить заявку

        </Link>

      </div>

    </div>

  );

};

export default CatalogCards;

Файл Catalog-Cards.scss

@import '../../variables.modules.scss';

.catalog-cards {

  display: flex;

  column-gap: 12px;

  row-gap: 60px;

  flex-wrap: wrap;

  justify-content: space-between;

  align-items: center;

}

@media screen and (max-width: 1199px) {

  .catalog-cards {

    justify-content: space-evenly;

  }

  #catalog-button {

    display: none;

  }

}

@media screen and (max-width: 768px) {

  .catalog-cards {

    flex-direction: column;

  }

}

Файл Catalog-Card.js

import "./Catalog-Card.scss";

import CatalogCardDesc from "./Catalog-Card-Desc/Catalog-Card-Desc";

import ImgPlug1 from "assets/media/images/img5.jpg";

const CatalogCard = ({ cardData }) => {

  const { title } = cardData;

  return (

    <div className="catalog-card">

      <img src={ImgPlug1} alt="img1" />

      <h3>{title}</h3>

      <CatalogCardDesc cardData={cardData} />

    </div>

  );

};

export default CatalogCard;

Файл Catalog-Card.scss

Файл @import '../../../variables.modules.scss';

.catalog-card {

  h3 {

    font-family: $font-main;

    font-size: $font-size-24;

    margin: 15px 0 10px;

  }

}

.catalog-card > img {

  width: 100%;

}

.button {

  text-decoration: none;

  padding: 20px 40px;

  background-color: $color-buttons;

  color: $color-white;

  font-family: $font-semibold;

  border-radius: $border-radius-button;

  transition: $transition-buttons;

  &:hover {

    opacity: 0.8;

    transition: $transition-buttons;

  }

}

@media screen and (max-width: 768px) {

  .catalog-card {

    h3 {

      text-align: center;

      font-size: $font-size-20;

    }

  }

}

Файл Catalog-Card-Desc.js

import './Catalog-Card-Desc.scss';

import CatalogCardDescElem from './Catalog-Card-Desc-Elem/Catalog-Card-Desc-Elem';

import IconTime from 'assets/media/icons/catalog/catalog-time.svg';

import IconPrice from 'assets/media/icons/catalog/catalog-price.svg';

const CatalogCardDesc = ({ cardData }) => {

  const { textTime, textPrice } = cardData;

  return (

    <div class="catalog-card-desc">

      <CatalogCardDescElem imgPath={IconTime} text={textTime}/>

      <CatalogCardDescElem imgPath={IconPrice} text={textPrice}/>

    </div>

  )

}

export default CatalogCardDesc;

Файл Catalog-Card-Desc.scss

@import '../../../../variables.modules.scss';

.catalog-card-desc {

  display: flex;

  gap: 35px;

}

@media screen and (max-width: 768px) {

  .catalog-card-desc {

    justify-content: center;

    gap: 3%;

  }

}

Файл Catalog-Card-Desc-Elem.js

import './Catalog-Card-Desc-Elem.scss';

const CatalogCardDescElem = ({text, imgPath}) => {

  return (

    <div class="catalog-card-desc-elem">

      <img src={imgPath} alt=""/>

      <p>{text}</p>

    </div>

  )

}

export default CatalogCardDescElem;

Файл Catalog-Card-Desc-Elem.scss

@import '../../../../../variables.modules.scss';

.catalog-card-desc-elem {

  display: flex;

  gap: 10px;

  align-items: center;

  p {

    font-family: $font-main;

    font-size: $font-size-15;

  }

}

Файл Faq.js

import "./Faq.scss";

import FaqCard from "./FaqCard/FaqCard";

const Faq = () => {

  const faqData = [

    {

      id: 1,

      summary: "Как узнать примерную стоимость конструкций?",

      text: "Оставьте заявку на сайте, после мы договоримся о бесплатном замере и расчёте стоимости работ.",

    },

    {

      id: 2,

      summary: "Вывозите ли вы за собой мусор?",

      text: "Да, мы вывозим весь строительный  мусор с объекта.",

    },

    {

      id: 3,

      summary: "Есть ли гарантия?",

      text: "Мы предоставляем 1 год бесплатного гарантийного обслуживания.",

    },

    {

      id: 4,

      summary: "В каких регионах работаете?",

      text: "Работаем в северо-западном федеральном округе, центральном федеральном округе, южном федеральном округе.",

    },

  ];

  return (

    <section class="faq wrap" id="faq">

      <h2 class="headers-h2">Часто задаваемые вопросы</h2>

      {faqData.map((cardData) => (

        <FaqCard key={cardData.id} cardData={cardData} />

      ))}

    </section>

  );

};

export default Faq;

Файл Faq.scss

@import '../variables.modules.scss';

@media screen and (max-width: 768px) {

  .faq {

    text-align: center;

  }

}

Файл FaqCard.js

import './FaqCard.scss';

const FaqCard = ({ cardData }) => {

  const { summary, text } = cardData;

  return (

    <details>

      <summary>{summary}</summary>

      {text}

    </details>

  )

}

export default FaqCard;

Файл FaqCard.scss

@import '../../variables.modules.scss';

details {

  font-family: $font-main;

  font-size: $font-size-20;

  padding: 10px 10px 0;

  margin: 35px;

}

summary {

  margin: -10px -10px 0;

  padding: 10px;

}

details[open] {

  padding: 10px;

}

details[open] summary {

  border-bottom: 1px solid $color-buttons;

  margin-bottom: 1em;

}

@media screen and (max-width: 450px) {

  details {

    font-size: $font-size-15;

  }

}

Файл Feedback.js

import "./Feedback.scss";

const Feedback = () => {

  return (

    <section class="feedback" id="feedback">

      <div class="wrap">

        <h2 class="headers-h2">Оставить заявку</h2>

        <div class="feedback-content">

          <div class="feedback-content-col">

            <p>

              Компания "GlassArt" - современные стеклянные и зеркальные

              конструкции, а также индивидуальный подход к каждому Клиенту.

            </p>

            <input type="checkbox" id="" name="" checked />

            <label for="">Разрешаю обработку персональных данных</label>

          </div>

          <div class="feedback-content-col">

            <form>

              <input type="text" placeholder="Ваше имя\*" required />

              <input type="text" placeholder="Ваш телефон\*" required />

              <input type="email" placeholder="Адрес эл. почты" />

              <input type="submit" value="Отправить" />

            </form>

          </div>

        </div>

      </div>

    </section>

  );

};

export default Feedback;

Файл Feedback.scss

@import '../variables.modules.scss';

.feedback{

  background-color: $color-buttons;

  h2 {

    color: $color-white;

  }

}

.feedback-content {

  max-width: 90%;

  margin: 0 auto;

  padding-bottom: 60px;

  display: flex;

  gap: 80px;

}

.feedback-content-col {

  width: 50%;

  p {

    font-family: $font-main;

    font-size: $font-size-20;

    color: $color-white;

    margin-bottom: 20px;

  }

  form {

    padding: 0 30px;

    display: flex;

    flex-direction: column;

    gap: 15px;

  }

  input[type='text'], input[type='email'] {

    border: none;

    padding: 18px 30px;

    transition: $transition-border-radius;

    &:hover {

      border-radius: 5px;

      transition: $transition-border-radius;

    }

  }

  input[type='submit'] {

    width: 70%;

    border: none;

    font-family: $font-semibold;

    font-size: $font-size-20;

    color: $color-buttons;

    text-transform: uppercase;

    opacity: 0.9;

    padding: 18px;

    margin-top: 20px;

    transition: all 0.3s ease;

    &:hover {

      cursor: pointer;

      opacity: 1;

      border-radius: 5px;

      transition: all 0.3s ease;

    }

  }

  input::placeholder {

    color: $color-buttons;

    opacity: 0.7;

    font-family: $font-main;

    font-size: $font-size-15;

    text-transform: uppercase;

  }

  label {

    font-family: $font-main;

    font-size: $font-size-15;

    margin-left: 10px;

    color: $color-white;

  }

}

@media screen and (max-width: 768px) {

  .feedback-content {

    flex-direction: column;

    align-items: center;

  }

  .feedback-content-col {

    width: 80%;

    text-align: center;

    form {

      align-items: center;

    }

    input[type='text'], input[type='email'], input[type='submit'] {

      width: 80%;

    }

  }

}

@media screen and (max-width: 450px) {

  .feedback-content-col {

    p {

      font-size: $font-size-15;

    }

    input::placeholder, input[type='submit'] {

      font-size: 12px;

    }

  }

}

Файл Footer.js

import LogoWhite from "assets/media/icons/header/logo\_1.png";

import IconAddress from "assets/media/icons/header/header-address.svg";

import IconPhone from "assets/media/icons/header/header-phone.svg";

import IconTime from "assets/media/icons/header/header-time.svg";

import TelegramIcon from "assets/media/icons/header/telegram.svg";

import WhatsAppIcon from "assets/media/icons/header/whatsapp.svg";

import "./Footer.scss";

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

const Footer = () => {

  return (

    <footer id="footer">

      <div class="header-content footer-content wrap">

        <Link to="/">

          <img src={LogoWhite} alt="logo" />

        </Link>

        <nav>

          <ul>

            <li>

              <Link to={{ pathname: "/", hash: "#about" }}>Главная</Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#catalog" }}>Каталог</Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#services" }}>Услуги</Link>

            </li>

            <li>

              <Link to="/works">Работы</Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#faq" }}>FAQ</Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#footer" }}>Контакты</Link>

            </li>

          </ul>

        </nav>

      </div>

      <div class="wrap before-header footer-content">

        <div class="bheader-contact">

          <div class="bheader-content-elem">

            <img src={IconAddress} class="filter-white" alt="" />

            <a href="#">Россия, г.Тамбов, бульвар Энтузиастов, 1Ас1</a>

          </div>

          <div class="bheader-content-elem">

            <img src={IconPhone} class="filter-white" alt="" />

            <a href="#">+7 (4752) 75-58-00</a>

          </div>

          <div class="bheader-content-elem">

            <img src={IconTime} class="filter-white" alt="" />

            <a href="#">Пн-Пт: 9:00 - 18:00</a>

          </div>

        </div>

        <div class="bheader-buttons">

          <a href="#feedback" class="bheader-button">

            Оставить заявку

          </a>

          <a href="https://t.me">

            <img src={TelegramIcon} alt="" />

          </a>

          <a href="https://wa.me">

            <img src={WhatsAppIcon} alt="" />

          </a>

        </div>

      </div>

      <div class="wrap confidence">

        <p>Политика конфиденциальности</p>

        <p>GlassArt 2024</p>

      </div>

    </footer>

  );

};

export default Footer;

Файл Footer.scss

@import '../variables.modules.scss';

footer {

  margin-top: 100px;

  background-color: $color-contrast;

}

.footer-content {

  padding-top: 40px;

  a {

    color: $color-white;

  }

}

.filter-white {

  filter: invert(100%) sepia(0%) saturate(0%) hue-rotate(93deg) brightness(103%) contrast(103%);

}

.confidence {

  display: flex;

  justify-content: space-between;

  font-family: $font-main;

  font-size: $font-size-15;

  color: $color-white;

  padding: 60px 0 15px 0;

}

Файл Header.js

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

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

import "components/Header/Header.scss";

import Logo from "assets/media/icons/header/logo\_1.png";

const Header = () => {

  const [menuOpen, setMenuOpen] = useState(false);

  const toggleMenu = () => {

    setMenuOpen(!menuOpen);

  };

  const closeMenu = () => {

    setMenuOpen(false);

  };

  const handleOutsideClick = (event) => {

    if (event.target.classList.contains("overlay")) {

      closeMenu();

    }

  };

  useEffect(() => {

    if (menuOpen) {

      document.addEventListener("click", handleOutsideClick);

    } else {

      document.removeEventListener("click", handleOutsideClick);

    }

    return () => {

      document.removeEventListener("click", handleOutsideClick);

    };

  }, [menuOpen]);

  return (

    <header>

      <div className="header-content wrap">

        <Link to="/">

          <img src={Logo} alt="logo" />

        </Link>

        <button className="burger-menu" onClick={toggleMenu}>

          ☰

        </button>

        <nav className={menuOpen ? "open" : ""}>

          <button className="close-menu" onClick={closeMenu}>

            ✕

          </button>

          <ul>

            <li>

              <Link to={{ pathname: "/", hash: "#about" }} onClick={closeMenu}>

                Главная

              </Link>

            </li>

            <li>

              <Link

                to={{ pathname: "/", hash: "#catalog" }}

                onClick={closeMenu}

              >

                Каталог

              </Link>

            </li>

            <li>

              <Link

                to={{ pathname: "/", hash: "#services" }}

                onClick={closeMenu}

              >

                Услуги

              </Link>

            </li>

            <li>

              <Link to="/works" onClick={closeMenu}>

                Работы

              </Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#faq" }} onClick={closeMenu}>

                FAQ

              </Link>

            </li>

            <li>

              <Link to={{ pathname: "/", hash: "#footer" }} onClick={closeMenu}>

                Контакты

              </Link>

            </li>

          </ul>

        </nav>

      </div>

      <div

        className={menuOpen ? "overlay open" : "overlay"}

        onClick={closeMenu}

      ></div>

    </header>

  );

};

export default Header;

Файл Header.scss

@import "../variables.modules.scss";

header {

  background-color: $color-white;

  box-shadow: 0 10px 15px 0 rgba(0, 0, 0, 0.25);

  z-index: 10;

  position: relative;

}

.header-content {

  display: flex;

  justify-content: space-between;

  align-items: center;

  padding: 10px 0;

  .burger-menu {

    display: none;

    background: none;

    border: none;

    font-size: 2rem;

    cursor: pointer;

  }

  .close-menu {

    display: none;

    background: none;

    border: none;

    font-size: 2rem;

    cursor: pointer;

    position: absolute;

    top: 20px;

    right: 20px;

    color: $color-white;

  }

  nav {

    ul {

      display: flex;

      gap: 20px;

    }

    li {

      list-style-type: none;

      font-family: $font-main;

    }

    a {

      text-decoration: none;

      color: rgb(18, 112, 235);

      transition: $transition-buttons;

      &:hover {

        opacity: 0.8;

        transition: $transition-buttons;

      }

    }

  }

}

@media screen and (max-width: 900px) {

  .header-content {

    .burger-menu {

      display: block;

    }

    .close-menu {

      display: block;

    }

    nav {

      display: flex;

      flex-direction: column;

      align-items: center;

      justify-content: center;

      position: fixed;

      top: 0;

      right: 0;

      height: 100vh;

      width: 300px;

      background-color: $color-contrast;

      transform: translateX(100%);

      transition: transform 0.3s ease-in-out;

      z-index: 20;

      &.open {

        transform: translateX(0);

      }

      ul {

        display: flex;

        flex-direction: column;

        align-items: center;

        gap: 0;

        padding: 0;

        margin: 0;

        height: 100%;

        justify-content: center;

        li {

          margin: 20px 0;

        }

        a {

          color: $color-white;

          font-size: 1.5rem;

        }

      }

    }

  }

  .overlay {

    display: none;

    position: fixed;

    top: 0;

    left: 0;

    width: 100%;

    height: 100%;

    background: rgba(0, 0, 0, 0.5);

    z-index: 15;

  }

  .overlay.open {

    display: block;

  }

}

Файл Reviews.js

import './Reviews.scss';

import ReviewsCards from './Reviews-Cards/Reviews-Cards';

const Reviews = () => {

  return (

    <section class="reviews wrap">

      <h2 class="headers-h2">Отзывы клиентов</h2>

      <p class="reviews-subtitle">Вы можете отправить свой отзыв на почту: </p>

      <ReviewsCards/>

    </section>

  )

}

export default Reviews;

Файл Reviews.scss

@import '../variables.modules.scss';

.reviews {

  h2 {

    margin-bottom: 10px;

  }

}

.reviews-subtitle {

  font-family: $font-main;

  font-size: $font-size-20;

  margin-bottom: 60px;

}

@media screen and (max-width: 768px) {

  .reviews-subtitle {

    text-align: center;

  }

}

Файл Reviews-Cards.js

import "./Reviews-Cards.scss";

import Plug1 from "assets/media/icons/reviews/img\_woman.png";

import Plug2 from "assets/media/icons/reviews/img\_man.jpg";

import ReviewsCard from "./Review-Card/Review-Card";

const ReviewsCards = () => {

  const revCardsData = [

    {

      id: 1,

      imgPath: Plug1,

      text: "Изящно-воздушные стеклянные конструкции.",

    },

    {

      id: 2,

      imgPath: Plug2,

      text: "Сделали все не подкопаешься, и сам профиль красивый, и стеклопакет отличный с энергосбережением, и фурнитура европейская.",

    },

    {

      id: 3,

      imgPath: Plug1,

      text: "Переживала за нестандартные проемы, но сделали все идеально в плане размеров.",

    },

    {

      id: 4,

      imgPath: Plug2,

      text: "Душевой кабиной доволен, не жалею, что именно в GlassArt заказал.",

    },

  ];

  return (

    <div class="reviews-cards">

      {revCardsData.map((cardData) => (

        <ReviewsCard key={cardData.id} cardData={cardData} />

      ))}

    </div>

  );

};

export default ReviewsCards;

Файл Reviews-Cards.scss

@import "../../variables.modules.scss";

.reviews-cards {

  display: flex;

  justify-content: space-between;

  gap: 15px;

  margin-bottom: 100px;

}

@media screen and (max-width: 1199px) {

  .reviews-cards {

    flex-wrap: wrap;

    row-gap: 60px;

  }

}

@media screen and (max-width: 768px) {

  .reviews-cards {

    justify-content: center;

  }

}

Файл Review-Card.js

import './Review-Card.scss';

const ReviewsCard = ({ cardData }) => {

  const { imgPath, text } = cardData;

  return (

    <div class="reviews-card">

      <img src={imgPath} alt="reviews"/>

      <p>{text}</p>

    </div>

  )

}

export default ReviewsCard;

Файл Review-Card.scss

@import '../../../variables.modules.scss';

.reviews-card {

  display: flex;

  flex-direction: column;

  align-items: center;

  p {

    font-family: $font-main;

    font-size: $font-size-20;

    text-align: center;

    padding: 60px 10% 50px 10%;

    border-radius: 10px;

    backdrop-filter: blur(40px);

    box-shadow: 0 5px 0 0 rgba(255, 255, 255, 0.95), 0 40px 30px 0 rgba(0, 0, 0, 0.12);

    background: #fff;

  }

}

@media screen and (max-width: 500px) {

  .reviews-card {

    img {

      width: 60%;

    }

    width: 70%;

  }

}

Файл Services.js

import './Services.scss';

import ServicesCards from './Services-Cards/Services-Cards';

const Services = () => {

  return (

    <section class="services wrap" id="services">

      <h2 class="headers-h2">Работа с нами покажется простой</h2>

      <ServicesCards/>

    </section>

  )

}

export default Services;

Файл Services.scss

@import '../variables.modules.scss';

Файл Services-Cards.js

import './Services-Cards.scss';

import { Link } from 'react-router-dom';

import ServicesCard from './Services-Card/Services-Card';

const ServicesCards = () => {

  const servCardsData = [

    {

      id: 2,

      header: "ПРЕДВАРИТЕЛЬНЫЙ РАСЧЕТ",

      span: "",

      text: "БЕСПЛАТНО ПОДГОТОВИМ ПРЕДВАРИТЕЛЬНЫЙ РАСЧЕТ",

    },

    {

      id: 3,

      header: "ВЫЕЗД ИНЖЕНЕРА",

      span: "1 ДЕНЬ",

      text: "БЕСПЛАТНО НАПРАВИМ ИНЖЕНЕРА НА ОБЪЕКТ",

    },

    {

      id: 4,

      header: "ОКОНЧАТЕЛЬНЫЙ РАСЧЕТ И ДОГОВОР",

      span: "",

      text: "ПОДПИСАНИЕ ДОГОВОРА, ВНЕСЕНИЕ ПРЕДОПЛАТЫ",

    },

    {

      id: 5,

      header: "ДОСТАВКА И МОНТАЖ",

      span: "1 ДЕНЬ",

      text: "ИЗГОТОВЛЕНИЕ, ДОСТАВКА И МОНТАЖНЫЕ РАБОТЫ",

    },

    {

      id: 6 ,

      header: "ПОДПИСАНИЕ АКТОВ",

      span: "",

      text: "ПРИЕМКА ВЫПОЛНЕННЫХ РАБОТ",

    },

  ]

  return (

    <div class="services-cards">

      <div class="services-card">

        <p class="services-card-bg">1</p>

        <div class="services-card-content">

          <p>1</p>

          <div class="services-card-content-elem">

            <h3>ВАША ЗАЯВКА</h3>

            <div class="services-button">

              <Link to="/#feedback">Оставить заявку</Link>

            </div>

          </div>

        </div>

      </div>

      {servCardsData.map(cardData => (

        <ServicesCard key={cardData.id} cardData={cardData} />

      ))}

    </div>

  )

}

export default ServicesCards;

Файл Services-Cards.scss

@import '../../variables.modules.scss';

.services-cards {

  display: flex;

  justify-content: space-between;

  flex-wrap: wrap;

  row-gap: 60px;

}

.services-button {

  margin-top: 25px;

  padding: 10px 15px;

  background-color: $color-buttons;

  border-radius: $border-radius-button;

  transition: $transition-buttons;

  &:hover {

    opacity: 0.8;

    transition: $transition-buttons;

    cursor: pointer;

  }

  a {

    font-family: $font-semibold;

    font-size: $font-size-20;

    color: $color-white;

    text-decoration: none;

  }

}

@media screen and (max-width: 1000px) {

  .services-cards {

    align-items: center;

    row-gap: 10px;

  }

}

@media screen and (max-width: 768px) {

  .services-cards {

    flex-direction: column;

  }

  .services-button {

    margin-top: 15px;

    padding: 5px 10px;

    a {

      font-size: $font-size-15;

    }

  }

}

Файл Services-Card.js

import './Services-Card.scss';

const ServicesCard = ({ cardData }) => {

  const { id, header, span, text }  = cardData;

  return (

    <div class="services-card">

        <p class="services-card-bg">{id}</p>

        <div class="services-card-content">

          <p>{id}</p>

          <div class="services-card-content-elem">

            <h3>{header}</h3>

            <span>{span}</span>

            <p>{text}</p>

          </div>

        </div>

      </div>

  )

}

export default ServicesCard;

Файл Services-Card.scss

@import '../../../variables.modules.scss';

.services-card-bg {

  font-family: $font-semibold;

  font-size: 150px;

  color: $color-white;

  text-shadow:

  -1px -1px 0 #CBCBCB,

   1px -1px 0 #CBCBCB,

   -1px 1px 0 #CBCBCB,

    1px 1px 0 #CBCBCB;

}

.services-card {

  overflow: hidden;

  position: relative;

  width: 28%;

}

.services-card-content {

  display: flex;

  align-items: baseline;

  gap: 20px;

  position: absolute;

  top: 28%;

  left: 0;

  z-index: 1;

  p {

    font-family: $font-main;

    font-size: $font-size-36;

  }

}

.services-card-content-elem {

  h3 {

    font-family: $font-main;

    font-size: $font-size-24;

  }

  span {

    padding-top: 8px;

    font-family: $font-semibold;

    font-size: $font-size-15;

    color: $color-buttons;

  }

  p {

    margin-top: 25px;

    font-size: $font-size-15;

  }

}

@media screen and (max-width: 1199px) {

  .services-card {

    width: 28%;

    height: 30vw;

  }

  .services-card-bg {

    font-size: $font-size-90;

  }

  .services-card-content {

    p {

      font-size: $font-size-20;

    }

  }

  .services-card-content-elem {

    h3 {

      font-size: $font-size-24;

    }

  }

}

@media screen and (max-width: 1000px) {

  .services-card {

    width: 40%;

    height: 40vw;

  }

}

@media screen and (max-width: 768px) {

  .services-card{

   width: 60%;

   height: 25vw;

  }

  .services-card-content-elem {

    span {

      font-size: 10px;

    }

  }

}

@media screen and (max-width: 700px) {

  .services-card {

    width: 80%;

    height: 40vw;

  }

}

@media screen and (max-width: 460px) {

  .services-card {

    width: 85%;

    height: 55vw;

  }

  .services-card-bg {

    font-size: $font-size-75;

  }

  .services-card-content {

    p {

      margin-top: 10px;

      font-size: $font-size-15;

    }

  }

  .services-card-content-elem {

    h3 {

      font-size: $font-size-20;

    }

  }

}

@media screen and (max-width: 340px) {

  .services-card {

    width: 90%;

    height: 55vw;

  }

}

Файл Slogan.js

import "./Slogan.scss";

const Slogan = () => {

  return (

    <section class="slogan">

      <div class="slogan-content wrap">

        <p class="slogan-main">

          Компания <span>GlassArt -</span>

        </p>

        <p class="slogan-gradient" data-text="уникальные">

          уникальные

        </p>

        <p class="slogan-clear">прозрачные</p>

        <p class="slogan-gradient slogan-right-text">решения</p>

      </div>

    </section>

  );

};

export default Slogan;

Файл Slogan.scss

@import '../variables.modules.scss';

.slogan {

  margin-top: 100px;

  background-color: $color-contrast;

  color: $color-white;

  padding-top: 0;

}

.slogan-main {

  font-family: $font-main;

  font-size: $font-size-52;

  color: $color-white;

  span {

    font-family: $font-mitr;

  }

}

.slogan-gradient {

  font-family: $font-black;

  font-size: $font-size-75;

  background: radial-gradient(circle at center, #6385FD, #fff);

  -webkit-background-clip: text;

  background-clip: text;

  color: transparent;

}

.slogan-clear {

  font-family: $font-black;

  font-size: $font-size-90;

  text-align: center;

  color: $color-contrast;

  text-shadow:

  -1px -1px 0 #fff,

   1px -1px 0 #fff,

   -1px 1px 0 #fff,

    1px 1px 0 #fff;

}

.slogan-right-text {

  text-align: right;

}

.slogan-content {

  padding: 60px 0;

}

@media screen and (max-width: 768px) {

  .slogan-main {

    font-size: $font-size-36;

  }

  .slogan-gradient, .slogan-clear {

    font-size: $font-size-52;

  }

}

@media screen and (max-width: 400px) {

  .slogan-right-text, .slogan-gradient, .slogan-clear, .slogan-main {

    text-align: center;

    font-size: $font-size-36;

  }

}

Файл Swiper.js

import Swiper from 'react-id-swiper';

import React, { useRef, useState, useEffect } from 'react';

import '../../../node\_modules/swiper/swiper.scss';

import carouselImg1 from 'assets/media/images/carousel1.jpg';

import carouselImg2 from 'assets/media/images/carousel2.jpg';

import carouselImg3 from 'assets/media/images/carousel3.jpg';

import carouselImg4 from 'assets/media/images/carousel4.jpg';

import carouselImg5 from 'assets/media/images/carousel5.jpg';

import './Swiper.scss';

import About from '../About/About';

const ManipulatingComponentOutSideSwiper = () => {

  const swiperRef = useRef(null);

  const [activeIndex, setActiveIndex] = useState(0);

  const goNext = () => {

    if (swiperRef.current && swiperRef.current.swiper) {

      swiperRef.current.swiper.slideNext();

    }

  };

  const goPrev = () => {

    if (swiperRef.current && swiperRef.current.swiper) {

      swiperRef.current.swiper.slidePrev();

    }

  };

  const carouselImgData = [

    {

      id: 1,

      imgPath: carouselImg1,

      imgAlt: 'Стеклянные ограждения',

    },

    {

      id: 2,

      imgPath: carouselImg2,

      imgAlt: 'Стеклянные двери',

    },

    {

      id: 3,

      imgPath: carouselImg3,

      imgAlt: 'Стеклянные перегородки',

    },

    {

      id: 4,

      imgPath: carouselImg4,

      imgAlt: 'Душевые кабины',

    },

    {

      id: 5,

      imgPath: carouselImg5,

      imgAlt: 'Зеркала и панно',

    }

  ];

  useEffect(() => {

    const swiperInstance = swiperRef.current.swiper;

    swiperInstance.on('slideChange', () => {

      setActiveIndex(swiperInstance.activeIndex);

    });

  }, []);

  return (

    <div className="slider">

      <div className='slider-container'>

        <Swiper ref={swiperRef}>

          {carouselImgData.map((imgData, index) => (

            <div className='slider-slide' key={index}>

              <img src={imgData.imgPath} alt={imgData.imgAlt} />

            </div>

          ))}

        </Swiper>

        <div className="buttons-slider-container">

          <div>

            <p>{carouselImgData[activeIndex].imgAlt}</p>

          </div>

          <div className='buttons-slider'>

            <button onClick={goPrev}>&laquo;</button>

            <button onClick={goNext}>&raquo;</button>

          </div>

        </div>

      </div>

      <div className="white-paper">

        <About />

      </div>

    </div>

  );

};

export default ManipulatingComponentOutSideSwiper;

Файл Swiper.scss

@import '../variables.modules.scss';

.slider {

  display: flex;

  flex-direction: row;

  width: 100%;

  height: 600px;

  position: relative;

  @media screen and (max-width: 870px) {

    flex-direction: column;

    height: auto;

  }

}

.slider-container {

  position: relative;

  width: 63%;

  height: 100%;

  overflow: hidden; // Обеспечивает обрезку изображения

  img {

    width: 100%;

    height: 100%;

    object-fit: cover; // Изображение заполняет контейнер

    object-position: center; // Центровка изображения

  }

  @media screen and (max-width: 870px) {

    width: 100%;

    height: auto;

    img {

      height: auto;

      max-height: 600px;

    }

  }

}

.white-paper {

  z-index: 1;

  background-color: #fff;

  width: 37%;

  height: 100%;

  position: relative;

  right: 0;

  @media screen and (max-width: 870px) {

    width: 100%;

    max-height: 570px;

    height: auto;

  }

}

.buttons-slider-container {

  p {

    text-align: right;

    font-family: $font-semibold;

    font-size: $font-size-24;

    color: $color-white;

    @media screen and (max-width: 570px) {

      font-size: $font-size-20;

    }

  }

  padding: 20px 30px;

  position: absolute;

  top: 75%; // Центровка по вертикали

  right: 0px; // Отступ от правого края

  transform: translateY(-50%); // Центровка по вертикали

  border-radius: 15px;

  background-color: $color-contrast;

  row-gap: 20px;

  z-index: 3;

  display: flex;

  flex-direction: column;

  @media screen and (max-width: 870px) {

    top: auto;

    bottom: 0px;

    right: 50%;

    transform: translateX(50%);

    margin-right: 0;

    width: calc(100% - 40px);

    left: 50%;

    transform: translateX(-50%);

  }

}

.buttons-slider {

  display: flex;

  justify-content: flex-end;

  gap: 30px;

  button {

    padding: 2px 30px;

    font-size: 30px;

    color: $color-white;

    background-color: #FF003D;

    border-radius: $border-radius-button;

    border: none;

    transition: $transition-buttons;

    &:hover {

      transition: $transition-buttons;

      opacity: 0.8;

      cursor: pointer;

    }

  }

}

@media screen and (max-width: 1200px) {

  .slider {

    flex-direction: column;

    width: 100%;

    height: auto;

  }

  .white-paper {

    position: relative;

    width: 100%;

    height: auto;

  }

  .slider-container {

    width: 100%;

    height: auto;

    overflow: hidden; // Обеспечивает обрезку изображения

    img {

      width: 100%;

      height: 100%;

      object-fit: cover;

      object-position: center; // Центровка изображения

    }

  }

}

Файл Works-Content.js

import './Works-Content.scss';

const WorksContent = () => {

  return (

    <section className='wrap'>

      <h2 className='headers-h2'>Наши работы</h2>

    </section>

  )

};

export default WorksContent;

Файл Works-Content.scss

@import '../variables.modules.scss';

Файл variables.modules.scss

$font-main: "Montserrat-Regular", sans-serif;

$font-semibold: "Montserrat-SemiBold", sans-serif;

$font-black: "Montserrat-Black", sans-serif;

$font-mitr: "Mitr-Light", sans-serif;

$font-size-15: 15px;

$font-size-20: 20px;

$font-size-24: 24px;

$font-size-36: 36px;

$font-size-52: 52px;

$font-size-75: 75px;

$font-size-90: 90px;

$color-black: #000;

$color-white: #fff;

$color-buttons: #001581;

$color-contrast: #151D46;

$border-radius-button: 4px;

$transition-buttons: opacity 0.3s ease;

$transition-border-radius: border-radius 0.3s ease;

:export {

  color-black: $color-black;

  color-white: $color-white;

  color-buttons: $color-buttons;

  color-contrast: $color-contrast;

  font-main: $font-main;

  font-semibold: $font-semibold;

  font-black: $font-black;

  font-mitr: $font-mitr;

  font-size-15: $font-size-15;

  font-size-20: $font-size-20;

  font-size-24: $font-size-24;

  font-size-36: $font-size-36;

  font-size-52: $font-size-52;

  font-size-75: $font-size-75;

  font-size-90: $font-size-90;

  border-radius-button: $border-radius-button;

  transition-buttons: $transition-buttons;

  transition-border-radius: $transition-border-radius;

}

\* {

  margin: 0;

  padding: 0;

}

html,

body {

  scroll-behavior: smooth;

  overflow-x: hidden;

}

.wrap {

  max-width: 1200px;

  width: 100%;

  margin: 0 auto;

}

section {

  padding-top: 100px;

}

.headers-h2 {

  font-family: $font-main;

  font-size: $font-size-52;

  margin-bottom: 60px;

}

@media screen and (max-width: 1199px) {

  .wrap {

    max-width: 90%;

  }

}

@media screen and (max-width: 768px) {

  .headers-h2 {

    text-align: center;

    font-size: $font-size-36;

  }

}

Файл variables.modules.scss

$font-main: "Montserrat-Regular", sans-serif;

$font-semibold: "Montserrat-SemiBold", sans-serif;

$font-black: "Montserrat-Black", sans-serif;

$font-mitr: "Mitr-Light", sans-serif;

$font-size-15: 15px;

$font-size-20: 20px;

$font-size-24: 24px;

$font-size-36: 36px;

$font-size-52: 52px;

$font-size-75: 75px;

$font-size-90: 90px;

$color-black: #000;

$color-white: #fff;

$color-buttons: #001581;

$color-contrast: #151D46;

$border-radius-button: 4px;

$transition-buttons: opacity 0.3s ease;

$transition-border-radius: border-radius 0.3s ease;

:export {

  color-black: $color-black;

  color-white: $color-white;

  color-buttons: $color-buttons;

  color-contrast: $color-contrast;

  font-main: $font-main;

  font-semibold: $font-semibold;

  font-black: $font-black;

  font-mitr: $font-mitr;

  font-size-15: $font-size-15;

  font-size-20: $font-size-20;

  font-size-24: $font-size-24;

  font-size-36: $font-size-36;

  font-size-52: $font-size-52;

  font-size-75: $font-size-75;

  font-size-90: $font-size-90;

  border-radius-button: $border-radius-button;

  transition-buttons: $transition-buttons;

  transition-border-radius: $transition-border-radius;

}

\* {

  margin: 0;

  padding: 0;

}

html,

body {

  scroll-behavior: smooth;

  overflow-x: hidden;

}

.wrap {

  max-width: 1200px;

  width: 100%;

  margin: 0 auto;

}

section {

  padding-top: 100px;

}

.headers-h2 {

  font-family: $font-main;

  font-size: $font-size-52;

  margin-bottom: 60px;

}

@media screen and (max-width: 1199px) {

  .wrap {

    max-width: 90%;

  }

}

@media screen and (max-width: 768px) {

  .headers-h2 {

    text-align: center;

    font-size: $font-size-36;

  }

}

Файл useScrollToHash.js

import { useEffect } from 'react';

import { useLocation } from 'react-router-dom';

function useScrollToHash() {

  const location = useLocation();

  useEffect(() => {

    if (location.hash) {

      const element = document.getElementById(location.hash.substring(1));

      if (element) {

        element.scrollIntoView({ behavior: 'smooth' });

      }

    }

  }, [location]);

  return null;

}

export default useScrollToHash;

Файл App.js

import React from 'react';

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

import BeforeHeader from './components/Before-Header/Before-Header';

import BHeaderHr from './components/Before-Header-Hr/Before-Header-Hr';

import Header from "./components/Header/Header";

import Catalog from './components/Catalog/Catalog';

import Slogan from './components/Slogan/Slogan';

import Advantages from './components/Advantages/Advantages';

import Services from './components/Services/Services';

import Reviews from './components/Reviews/Reviews';

import Feedback from './components/Feedback/Feedback';

import Faq from './components/Faq/Faq';

import Footer from './components/Footer/Footer';

import ManipulatingComponentOutSideSwiper from './components/Swiper/Swiper';

import WorksContent from './components/Works/Works-Content/Works-Content';

import useScrollToHash from './hooks/useScrollToHash';

function Home() {

  useScrollToHash();

  return (

    <>

      <ManipulatingComponentOutSideSwiper />

      <Catalog />

      <Slogan />

      <Advantages />

      <Services />

      <Reviews />

      <Feedback />

      <Faq />

    </>

  );

}

function Works() {

  return (

    <>

      <WorksContent />

    </>

  );

}

function App() {

  return (

    <Router>

      <BeforeHeader />

      <BHeaderHr />

      <Header />

      <Routes>

        <Route path="/" element={<Home />} />

        <Route path="/works" element={<Works />} />

      </Routes>

      <Footer />

    </Router>

  );

}

export default App;

Файл index.css

@font-face {

  font-family: "Mitr-Light";

  src: local("Mitr-Light"),

   url("./assets/fonts/Mitr-Light.ttf") format("truetype");

  font-weight: normal;

}

@font-face {

  font-family: "Montserrat-Black";

  src: local("Montserrat-Black"),

   url("./assets/fonts/Montserrat-Black.ttf") format("truetype");

}

@font-face {

  font-family: "Montserrat-Regular";

  src: local("Montserrat-Regular"),

   url("./assets/fonts/Montserrat-Regular.ttf") format("truetype");

}

@font-face {

  font-family: "Montserrat-SemiBold";

  src: local("Montserrat-Semibold"),

   url("./assets/fonts/Montserrat-SemiBold.ttf") format("truetype");

}

body {

  margin: 0;

  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',

    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',

    sans-serif;

  -webkit-font-smoothing: antialiased;

  -moz-osx-font-smoothing: grayscale;

}

code {

  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',

    monospace;

}

Файл index.js

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import './assets/fonts/Mitr-Light.ttf';

import './assets/fonts/Montserrat-Black.ttf';

import './assets/fonts/Montserrat-Regular.ttf';

import './assets/fonts/Montserrat-SemiBold.ttf';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);